Design and Application of Scalable Evolutionary Algorithms in Electricity Distribution Network Expansion Planning

Luong, Hoang

DOI
10.4233/uuid:15818216-0eb5-4545-bd2e-27eaa38262ba

Publication date
2018

Document Version
Final published version

Citation (APA)
Luong, H. (2018). *Design and Application of Scalable Evolutionary Algorithms in Electricity Distribution Network Expansion Planning*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:15818216-0eb5-4545-bd2e-27eaa38262ba

Design and Application of Scalable Evolutionary Algorithms in Electricity Distribution Network Expansion Planning

Design and Application of Scalable Evolutionary Algorithms in Electricity Distribution Network Expansion Planning

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on
Wednesday 17 October 2018 at 15:00 o'clock

by

Ngoc Hoang LUONG

Master of Science in Electrical and Computer Engineering,
Sungkyunkwan University, South Korea
born in Ho Chi Minh City, Vietnam.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus	chairperson
Prof.dr.ir. J.A. La Poutré	Delft University of Technology, promotor
Prof.dr. P.A.N. Bosman	Delft University of Technology, promotor
Independent members:	
Prof.dr. K. Deb	Michigan State University, USA
Prof.dr. P. Palensky	Delft University of Technology
Prof.dr.ir. G. Deconinck	KU Leuven, Belgium
Prof.dr. C. Witteveen	Delft University of Technology
Other members:	
Prof.dr.ir. J.G. Slootweg	Eindhoven University of Technology

The research in this thesis was performed at the Intelligent & Autonomous Systems (IAS) group of Centrum Wiskunde & Informatica (CWI), the national research institute for mathematics and computer science in the Netherlands. The work was carried out within the project Computational Capacity Planning in Electricity Networks (COCAPLEN, number 647.000.007), funded by the Netherlands Organization for Scientific Research (NWO). The COCAPLEN project was a joint project with the Electrical Energy Systems (EES) group of the Department of Electrical Engineering at Eindhoven University of Technology, the Netherlands.

Keywords:	evolutionary algorithms, multi-objective optimization, power systems, distribution networks, expansion planning
Printed by:	Ipskamp Printing
Front cover:	GECCO Collaboration Network. Created by Gabriela Ochoa, University of Stirling, Scotland, UK.

ISBN 978-94-028-1098-1
An electronic version of this dissertation is available at
`http://repository.tudelft.nl/`.

Science is a wonderful thing
if one does not have to earn one's living at it.

Albert Einstein

Contents

Summary

In our modern daily life, many activities require electricity, for example, the usage of domestic appliances, manufacturing, communication, and transportation. It is therefore essential to maintain a reliable supply of electricity to ensure the operation of such activities. The electricity supply, in a large part, depends on the underlying electrical networks that transfer electricity from power plants to meet the demand of end users. In the past, electricity consumption has grown over time and, at some point, the electricity demand will exceed the current capacity of certain network assets, causing overloads on parts of the networks. Functioning under overload conditions reduces the reliability of the networks and also damages network assets. Network reinforcement is thus required. This incurs substantial investment costs and time-consuming activities, such as acquisitions of new assets, constructions of substations, and installations of suitable cables and other electrical devices. Network operator companies, therefore, need to properly predict the growth of electricity demand and make suitable expansion plans to enhance the capacity of their networks. In addition, the recent emergence of renewable energy sources and smart grid technologies changes electricity consumption behaviors of users, the growth of electricity demand in general, and also the directions of network flows (due to local generation). This poses additional challenges that need to be addressed by the network operators. In this dissertation, we are interested in medium-voltage distribution networks, which are electrical networks that deliver electricity from high-voltage transmission networks to low-voltage distribution networks. Medium-voltage distribution networks typically have more complicated structures than low-voltage networks and require more frequent reinforcement activities than high-voltage transmission networks. We aim to develop robust computational methods to assist distribution network operators (DNOs) in tackling network expansion planning problems.

Electricity distribution network expansion planning (DNEP) is often formulated as an optimization problem, which involves finding the optimal expansion plan according to some objective. Solving such optimization problems requires many computational challenges to be properly addressed because they involve a set of non-convex, nonlinear equations that can give rise to many local optima in the search space. Also, there is no gradient information available due to the discrete nature of the choices to be made in solving DNEP problems, such as, the choice of which electrical devices to install from a list of standardized equipment. Another source of problem hardness comes from the fact that the planning process in practice typically involves multiple objectives that conflict with each other, such as minimizing investment cost as well as energy losses. However, there exists no *utopian* solution that optimizes all such objectives at the same time. Network operators often need to consider multiple alternative expansion plans that represent different trade-offs between involved objectives before determining which one is the desired trade-off

for a specific network.

Evolutionary algorithms (EAs) are a promising type of optimization algorithm to tackle the aforementioned challenges. EAs are population-based optimizers that maintain multiple candidate solutions at the same time, which is well-suited for efficiently and effectively obtaining a set of multiple trade-off solutions in one single optimization run. EAs normally do not require problem-specific knowledge, such as gradient information, in their operations. This makes them straightforward to use for black-box optimization where domain knowledge is not available or not straightforward to be efficiently exploited. If such domain knowledge is available for exploitation, the performance of EAs can be further enhanced.

However, there exist two major challenges in applying EAs to real-world optimization tasks, namely the setting of control parameters and achieving scalability. First, the settings of EA control parameters are crucial to the success of EAs in solving a specific problem instance. For real-world optimization, proper parameter settings for an EA, which depend on the structure of the problem instance and the operators of the EA itself, are very hard to be determined before running the EA. Practitioners thus often need to perform multiple optimization runs with different parameter settings in a time-consuming trial-and-error manner. Second, EAs that are employed for solving real-world problems like DNEP should be designed with good scalability in mind to ensure being able to efficiently handle a wide range of network sizes and structures. Regarding these challenges in the design and application of scalable EAs for solving DNEP problems, the following four research questions are formulated.

1. How can we model distribution network expansion planning (DNEP) as an optimization problem such that the outcomes of solving this problem are practically relevant while the optimization models are computationally feasible?

2. How can we design scalable EAs for solving (multi-objective) real-world applications, and in particular for solving (multi-objective) DNEP problems?

3. How can we solve DNEP problems when multiple conflicting objectives need to be taken into account such that DNOs are provided with insight into the trade-off relationship between the involved objectives?

4. How can we eliminate the troublesome requirement of control parameter settings when applying (multi-objective) EAs in practice?

This dissertation presents our research on the above questions and the obtained results.

In Chapter 1, we introduce the DNEP problem and explain the computational challenges that are involved in efficiently solving DNEP problems. We look into the current practice as well as the research literature on DNEP and explain why we choose to tackle DNEP problems with EAs. We then point out the challenges of designing and applying EA methods in practice, namely the scalability issue and the setting of EA control parameters.

In Chapter 2, we describe model-based evolutionary algorithms (MBEAs) that exploit models of problem structures during the optimization process. We focus on a major class of MBEAs that build linkage models to capture the dependency structure among problem variables and use the learned models to guide EA operators during optimization. Specifically, we present how the Genetic Algorithms (GAs) can be implemented as model-based EAs. We further consider two other types of model-based EAs: the Estimation-of-Distribution Algorithms (EDAs), and the Gene-pool Optimal Mixing Evolutionary Algorithms (GOMEAs). Using the Family-Of-Subsets (FOS) concept, we show how different types of linkage models can be employed in all of these EAs to customize their search capability.

In Chapter 3, we design a novel GOMEA for multi-objective discrete optimization (MO-GOMEA) by extending the original single-objective GOMEA. Specifically, we identify components that are crucial to the scalability of multi-objective evolutionary algorithms, namely the elitist archive of trade-off solutions, clustering of candidate solutions in the population, linkage learning, and exploiting the learned linkages during optimization. We then further enhance the usability of MO-GOMEA by eliminating the need of setting parameters. Experimental results on benchmark problems demonstrate the scalability and usability of our MO-GOMEA.

In Chapter 4, we introduce a novel single-objective formulation for the static DNEP problem that is both practically relevant and computationally efficient. The model, which considers optimization of the total cost of capital investment and operational expenditure, includes operation and design constraints as well as engineering rules that are considered in DNEP practice. We employ three classes of EAs, namely GAs, EDAs, and GOMEAs, to solve the optimization model instantiated with real-world distribution network data. Experimental results exhibit the superior performance of GOMEAs in solving DNEP compared to GAs and EDAs, even when being used out-of-the-box. The performance of GOMEAs can be further enhanced when their variation operator is customized with DNEP problem-specific knowledge.

In Chapter 5, we propose a novel decomposition heuristic to efficiently handle the dynamic DNEP problem that involves determining investment moments over a planning period. Using our decomposition heuristic, a suitable installation schedule can be obtained in an efficient manner for each static plan without explicitly modeling all time-related factors. Based on this, our problem formulation proposed in Chapter 4 for the static DNEP problem can be straightforwardly used for the dynamic DNEP problem without major modifications. Experimental results with three classes of EAs, namely GAs, EDAs, and GOMEAs, confirm the effectiveness of the decomposition heuristics for solving dynamic DNEP, and again, the excellent performance of GOMEAs. Furthermore, in this chapter, we consider and model Battery Energy Storage Systems (BESS) as a smart grid investment option together with the traditional assets such as electrical cables. The results demonstrate that BESS is a viable investment option for distribution network operators to consider in combination with the traditional option of installing electrical cables.

In Chapter 6, we demonstrate that the (dynamic) DNEP problem in practice involves multiple conflicting objectives, and should thus be solved in a true multi-

objective manner, approximating the Pareto-optimal front of trade-offs between different objectives. We additionally consider and model Demand Side Management (DSM) as a smart grid investment option together with the traditional physical asset installation. We employ the MO-GOMEA from Chapter 3 for solving different multi-objective DNEP problem models that involve multiple objectives, namely the investment cost, the energy losses, and the network reliability in terms of the averaged customer minutes lost. The resulting Pareto fronts provide DNOs with useful insights into the trade-offs between the objectives and can assist DNOs in choosing the expansion plan that exhibits the desired trade-off.

Concluding, the contribution of this dissertation is twofold. First, we show how the DNEP problem with real-world constraints can be modeled effectively and efficiently. Our problem formulation is highly customizable such that it can be straightforwardly modified by network operators to suit their needs: static or dynamic planning, employing only traditional assets or also smart grid technologies, and optimizing with respect to one objective or handling multiple objectives at the same time. Second, the dissertation proposes guidelines for constructing scalable and user-friendly EAs, that do not require users to tune their control parameters. Following the guidelines, we have designed the MO-GOMEA, which is shown to be capable of efficiently and effectively tackling real-world optimization tasks, such as different versions of the DNEP problem considered in this dissertation.

Samenvatting

In ons moderne dagelijks leven hebben veel activiteiten elektriciteit nodig, zoals het gebruik van huishoudelijke apparaten, productie, communicatie en transport. Het is daarom essentieel om voor een betrouwbare levering van elektriciteit te zorgen, zodat dergelijke activiteiten uitgevoerd kunnen blijven worden. De levering van elektriciteit hangt voor een groot deel af van onderliggende elektriciteitsnetten die elektriciteit transporteren vanaf energiecentrales om aan de vraag van eindgebruikers te voldoen. In het verleden is ons elektriciteitsverbruik gegroeid en zal het op een gegeven moment de huidige capaciteit van netwerkmaterieel overschrijden, wat resulteert in overbelasting van delen van het netwerk. Wanneer een netwerk functioneert in een situatie van overbelasting, verlaagt dit de betrouwbaarheid en beschadigt dit het netwerkmaterieel. Versterking van het netwerk is dus vereist. Dit vereist aanzienlijke investeringskosten en tijdrovende activiteiten, zoals de aanschaf van nieuw materieel, de bouw van transformatorstations en installaties van geschikte kabels en ander elektrisch materieel. Netbeheerders moeten daarom de toename van het elektriciteitsgebruik goed voorspellen en geschikte uitbreidingsplannen maken om de capaciteit van hun netwerken te vergroten. Bovendien verandert de recente opkomst van hernieuwbare energiebronnen en smart grid-technologieën het gedrag van elektriciteitsgebruikers, de groei van de elektriciteitsvraag in het algemeen en ook de richtingen van netwerkstromen (als gevolg van lokale elektriciteitsopwekking). Dit resulteert in nieuwe uitdagingen die moeten worden aangepakt door de netbeheerders. In dit proefschrift zijn we geïnteresseerd in middenspanningsdistributienetten. Dit zijn elektrische netwerken die elektriciteit overdragen van hoogspanningstransmissienetten naar laagspanningsdistributienetten. Middenspanningsdistributienetten hebben doorgaans gecompliceerdere structuren dan laagspanningsnetten en hebben frequenter versterkingsactiviteiten nodig dan hoogspanningstransmissienetten. We streven ernaar robuuste rekenmethoden te ontwikkelen om distributienetbeheerders te helpen bij het aanpakken van planningsproblemen bij netwerkuitbreidingen.

Het plannen van uitbereidingen aan distributienetten, in het Engels distribution network expansion planning (DNEP), wordt vaak geformuleerd als een optimalisatieprobleem, waarbij het optimale uitbereidingsplan gevonden moet worden aan de hand van een bepaalde doelstelling. Bij het oplossen van dit optimalisatieprobleem moeten veel rekenkundige uitdagingen worden aangepakt. In de probleemformulering komen niet-convexe, niet-lineaire vergelijkingen voor die veel lokale optima in de zoekruimte kunnen vormen. Ook is er geen gradiëntinformatie beschikbaar vanwege de discrete aard van de keuzes die gemaakt moeten worden in het oplossen van DNEP-problemen, zoals de keuze welk elektrisch materieel in een lijst met gestandaardiseerde apparatuur moet worden geïnstalleerd. Wat het probleem daarnaast vermoeilijkt is het feit dat het planningsproces typisch meerdere tegenstrijdige doelstellingen heeft, zoals het minimaliseren van investeringskosten alsook

het minimaliseren van energieverliezen. Er bestaat echter geen utopische oplossing die al deze doelen tegelijkertijd optimaliseert. Netbeheerders moeten daardoor vaak rekening houden met meerdere alternatieve uitbreidingsplannen die verschillende compromissen tussen de betrokken doelstellingen vertegenwoordigen voordat ze bepalen wat de gewenste afweging is voor een specifiek net.

Om bovengenoemde uitdagingen aan te pakken, zijn evolutionaire algoritmen (EA's) een veelbelovend type optimalisatie-algoritme. In EA's wordt een populatie van meerdere kandidaat-oplossingen tegelijkertijd beheerd, wat goed geschikt is om op efficiënte en effectieve wijze een verzameling oplossingen die verschillende afwegingen representeren, te verkrijgen in één enkele optimalisatierun. In de operaties die een EA uitvoert, is doorgaans geen probleemspecifieke informatie nodig, zoals gradiëntinformatie. Hierdoor zijn deze algoritmen gemakkelijk te gebruiken in het geval van black-box optimalisatie waarbij geen probleemspecifieke kennis beschikbaar is of het niet eenvoudig is om deze informatie efficiënt te gebruiken. Als dergelijke domeinkennis wel uitgebuit kan worden, kunnen de prestaties van EA's verder worden verbeterd.

Er zijn echter twee grote uitdagingen bij het gebruiken van EA's om praktijkgebaseerde optimalisatietaken uit te voeren, namelijk het instellen van regelparameters en de behalen van schaalbaarheid. Ten eerste spelen de instellingen van EA regelparameters een cruciale rol bij het succes van EA's bij het oplossen van een specifiek probleem. Voor optimalisatie in de praktijk zijn de juiste parameterinstellingen voor een EA, die afhankelijk zijn van de structuur van de probleeminstantie en het operatoren van het EA zelf, zeer lastig te bepalen voordat het EA wordt uitgevoerd. In praktijk moeten dus vaak meerdere optimalisatieruns worden uitgeprobeerd met steeds verschillende parameterinstellingen, wat tijdrovend is. Ten tweede moeten EA's die worden gebruikt voor het oplossen van praktijk-gebaseerde problemen zoals DNEP worden ontworpen met schaalbaarheid in het achterhoofd, om een breed scala aan netwerkgroottes en -structuren op efficiënte wijze aan te kunnen. Met betrekking tot deze uitdagingen bij het ontwerpen en toepassen van schaalbare EA's voor het oplossen van DNEP-problemen, zijn de volgende vier onderzoeksvragen geformuleerd.

1. Hoe kunnen we distribution network expansion planning (DNEP) zodanig als optimalisatieprobleem modelleren dat de uitkomsten van het oplossen van dit probleem praktisch relevant zijn terwijl de optimalisatiemodellen rekenkundig haalbaar zijn?

2. Hoe kunnen we schaalbare EA's ontwerpen voor (meerdoelige, ofwel met meerdere doelstellingen) praktijk-gebaseerde toepassingen en in het bijzonder voor het oplossen van (meerdoelige) DNEP-problemen?

3. Hoe kunnen we DNEP-problemen oplossen wanneer rekening gehouden moet worden met meerdere conflicterende doelen, zodanig dat netbeheerders inzicht krijgen in de wisselwerking tussen de betrokken doelen?

4. Hoe kunnen we het instellen van regelparameters elimineren bij het toepassen van (meerdoelige) EA's in de praktijk?

Dit proefschrift presenteert ons onderzoek naar de bovenstaande vragen en de verkregen resultaten.

In Hoofdstuk 1 introduceren we het DNEP probleem en leggen we uit wat de rekenkundige uitdagingen zijn bij het efficiënt oplossen hiervan. We onderzoeken de huidige praktijk alsook de onderzoeksliteratuur van DNEP en leggen uit waarom we ervoor kiezen om de DNEP problemen met EA methoden aan te pakken. Vervolgens wijzen we op de uitdagingen van het ontwerpen en toepassen van EA methoden in de praktijk, namelijk het schaalbaarheidsprobleem en het instellen van EA regelparameters.

In Hoofdstuk 2 beschrijven we modelgebaseerde evolutionaire algoritmen (MBEA's) die modellen van probleemstructuren uitbuiten tijdens het optimalisatieproces. We concentreren ons op een grote klasse MBEA's die modellen maken om de afhankelijkheidsstructuur tussen probleemvariabelen vast te leggen en om de geleerde modellen vervolgens te gebruiken om EA operatoren te leiden tijdens optimalisatie. In het bijzonder presenteren we hoe Genetic Algorithms (GA's) geïmplementeerd kunnen worden als model-gebaseerde EA's. We nemen voorts nog twee andere type model-gebaseerde EA's in ogenschouw: de Estimation-of-Distribution Algorithms (EDA's), en de Gene-pool Optimimal Mixing Evolutionary Algorithms (GOMEA's). Met het Family-Of-Subsets-concept (FOS) laten we zien hoe verschillende typen afhankelijkheidsmodellen in al deze EA's kunnen worden gebruikt om de zoekcapaciteit van deze EA's toe te snijden.

In Hoofdstuk 3 ontwerpen we een nieuwe GOMEA voor meerdoelige discrete optimalisatie (MO-GOMEA) door de originele enkeldoelige GOMEA uit te breiden. Specifiek identificeren we hierbij componenten die cruciaal zijn voor de schaalbaarheid van meerdoelige EA's, namelijk het elitist archive waarin oplossingen met verschillende afwegingen van de hoogste kwaliteit worden bewaard, het klusteren van kandidaat-oplossingen in de populatie, en het leren van afhankelijkheden en deze benutten tijdens optimalisatie. We verbeteren vervolgens de bruikbaarheid van MO-GOMEA verder door de noodzaak van het instellen van regelparameters te elimineren. Experimentele resultaten van benchmarkproblemen tonen de schaalbaarheid en bruikbaarheid van onze MO-GOMEA aan.

In Hoofdstuk 4 introduceren we een nieuwe enkeldoelige formulering voor het statische DNEP-probleem dat zowel praktisch relevant als rekenkundig efficiënt is. Het model beschrijft de optimalisatie van de gezamenlijke totale kosten van kapitaalinvesteringen en operationele uitgaven en neemt randvoorwaarden mee met betrekking tot de werking en het ontwerp van het netwerk zoals die ook in de praktijk gebruikt worden. We maken gebruik van drie klassen van EA's, namelijk GA's, EDA's en GOMEA's om het optimalisatiemodel, geïnstantieerd met data van distributienetten uit de praktijk, op te lossen. Experimentele resultaten laten de uitstekende prestaties van GOMEA's zien bij het oplossen van DNEP in vergelijking met GA's en EDA's, zelfs wanneer deze worden gebruikt zonder aanpassingen aan dit specifieke probleem. Door gebruik te maken van specifieke kennis van het DNEP-probleem in de variatie-operatoren van GOMEA's, kunnen de prestaties nog verder worden verbeterd.

In Hoofdstuk 5 stellen we een nieuwe decompositie-heuristiek voor om efficiënt

om te gaan met het dynamische DNEP-probleem, waarbij investeringsmomenten gedurende een planningsperiode bepaald worden. Met behulp van onze decompositieheuristiek kan voor elk statisch plan op een efficiënte manier een geschikt installatieschema worden verkregen zonder expliciet alle tijdgerelateerde factoren te modelleren. Op basis hiervan kan onze probleemformulering, voorgesteld in Hoofdstuk 4 voor het statische DNEP-probleem, eenvoudig worden gebruikt voor het dynamische DNEP-probleem zonder grote wijzigingen. Experimentele resultaten met drie klassen van EA's, namelijk GA's, EDA's en GOMEA's, bevestigen de effectiviteit van de decompositie-heuristieken voor het oplossen van het dynamische DNEP-probleem, en de wederom uitstekende prestaties van GOMEA's. Verder beschouwen en modelleren we in dit hoofdstuk Battery Energy Storage Systems (BESS) als een smart-grid-investeringsoptie samen met de traditionele materialen zoals elektrische kabels. De resultaten tonen aan dat BESS een haalbare investeringsoptie is voor netbeheerders om in overweging te nemen in combinatie met de traditionele optie om elektrische kabels te installeren.

In Hoofdstuk 6 laten we zien dat het (dynamische) DNEP-probleem meerdere conflicterende doelen omvat en dus op een meerdoelige manier moet worden opgelost, door het benaderen van het Pareto-optimale front van afwegingen tussen verschillende doelstellingen. We beschouwen en modelleren daarnaast Demand Side Management (DSM) als een smart grid investeringsoptie samen met de traditionele fysieke materiaalinstallatie. We gebruiken de in Hoofdstuk 3 ontworpen MO-GOMEA voor het oplossen van verschillende meerdoelige DNEP-probleemmodellen, waarbij de doelstellingen investeringskosten, energieverliezen en netwerkbetrouwbaarheid, uitgedrukt in verloren gemiddelde klantminuten, zijn. De resulterende Pareto-fronten bieden netbeheerders nuttige inzichten in de afwegingen tussen de doelstellingen die kunnen helpen bij het kiezen van het uitbreidingsplan met de gewenste afweging.

Concluderend is de bijdrage van dit proefschrift tweeledig. Eerst laten we zien hoe het DNEP-probleem met praktijk-gebaseerde randvoorwaarden effectief en efficiënt kan worden gemodelleerd. Onze probleemformulering is in hoge mate aanpasbaar, zodat deze eenvoudig kan worden toegesneden op de behoeften van netbeheerders: statische of dynamische planning, enkel traditioneel materieel of ook smart grid-technologieën, en optimaliseren met betrekking tot één doel of meerdere doelen tegelijkertijd. Daarnaast stelt het proefschrift richtlijnen ten behoeve van het ontwerpen van schaalbare en gebruiksvriendelijke EA's voor, waarbij gebruikers geen regelparameters hoeven af te stemmen. Aan de hand van deze richtlijnen hebben we MO-GOMEA ontworpen, waarvan wordt aangetoond dat deze in staat is om praktijk-gebaseerde optimalisatietaken met meerdere doelen op efficiënte en effectieve wijze aan te pakken, zoals de verschillende versies van de DNEP-problemen die in dit proefschrift aan bod zijn gekomen.

1

Introduction

't Begin van alle dingh is swaer,
Maer 't wert veel lichter achter naer.

The beginning of everything is hard,
But it becomes much easier later.

De Brune

1.1. Distribution network expansion planning

1.1.1. Electricity, power systems, and expansion planning

Electricity has a central position in our modern daily life. Electric power enables the operation of household appliances, the manufacturing activities at industrial sites, the control of transportation systems, the transmission of information in communication networks, and also the functioning of numerous other industries. Such omnipresence of electricity is brought about by the underlying power systems. A traditional power system [1] starts from power plants, often at remote areas, where primary energy sources like coal, oil, gas, nuclear fuels, or hydro, are converted into electricity. To efficiently transport electricity over long distances, step-up transformers are used to increase the voltage before transferring the generated electricity through the high-voltage (HV) transmission networks to demand centers [2]. Step-down transformers are then used to decrease the voltage before electricity is transferred to residential areas or industrial customers through the medium-voltage (MV) distribution networks. Step-down transformers again decrease the voltage before electricity is distributed to household consumers through the low-voltage (LV) distribution networks. Transformers are normally located in *substations*, where other electrical equipment, such as circuit breakers, switches, or regulators, can also be found. Full-scale power systems therefore can be seen as complex networks that are made of interconnected devices to generate, transport, and distribute electricity.

Electrical networks (i.e., transmission and distribution networks) are essential to the delivery of electricity generated at power plants to end-consumers. Failure of a network component (e.g., lines/cables, transformers) can lead to cascading failures, causing massive power outages. The switching off of the 380 kV line over the river Ems in Germany triggered a big cascade of line failures across Europe in 2006, disrupting the power supply of more than 15 million European households [3]. The contact of the overloaded Stuart-Atlanta 345 kV line with an overgrown tree was part of the causes for the widespread blackout in the United States' Midwest and Northeast regions and Canada's Ontario province in 2003, affecting about 50 millions people [4]. While transmission network failures often hit the media's headlines because of their massive scopes of effect, they seldom happen due to the N-1 or N-2 redundancy of HV networks (i.e., the network can still operate normally when one or two transmission lines fail). Outages of MV distribution networks, in fact, account for the largest contribution to the total system interruption (i.e., the total System Average Interruption Duration Index SAIDI) [5]. There are many more outages on LV parts of the network, but each of those affects only a small number of customers compared to less-frequent but larger-scaled MV outages [5].

To ensure normal and secure operation, the magnitude of power flows (i.e., electricity currents) from MV transmission substations (or HV/MV transformer substations) to electric power consumers should stay within the capacity of distribution network components (e.g., cables and transformers). Electricity consumption increases gradually over time, and the peak power demand, at a certain point in the future, will exceed the currently available network capacity, causing bottlenecks and thereby overloads. Long-time overloads on network assets are detrimental to power supply, network security, and assets' lifetime. Distribution network operators (DNOs) often draw out expansion plans to prevent such bottlenecks on the networks by reconfiguring the networks, replacing old and low-capacity network assets, installing new cable connections, or building new transformer substations. The new network configuration needs to have enough capacity to handle the forecasted growth of the peak power demand and also to satisfy other network constraints as well. For example, in the Netherlands, the network configuration should maintain a radial topology due to protection reasons. Urban distribution networks might also require the capability to reconfigure the network when network failures occur. Replacing only potentially overloaded assets might not be sufficient because the power demands can then be satisfied but the network reconfigurability is compromised. Therefore, feasible expansion plans might require more than locally solving network bottlenecks. The procedure that DNOs perform to design feasible expansion plans can be termed *distribution network expansion planning* (DNEP), which is the focus of this thesis.

For a given network, there are many possible expansion plans that satisfy the growth in power demand, but not all of them are equally good. For example, the simplest, and probably the most expensive, solution is upgrading all assets in the whole network, which also results in overcapacity and inefficiency of network facility usage. In practice, it is important to know the optimal expansion plan with respect to some objective. Finding the best solution out of all possible solutions

(i.e., the search space) is the goal of *optimization*. DNEP can be formulated as an optimization problem. The most common objective in DNEP is arguably to minimize the investment cost or the total cost (e.g., the combination of investment and operation cost). Solving what is known as the static DNEP then involves finding the most economical expansion plan that answers the questions *where*, on the grid, network capacity should be enhanced, and *what* kinds of network assets should be installed there to satisfy network constraints regarding the peak power demand predicted for the planning year. Dynamic DNEP problems also involve the question *when* enhancement activities should be carried out during the planning period. The difficulty of solving DNEP increases steeply in the dynamic case because potential expansions are considered in every year during the planning period, instead of only in the final horizon year.

1.1.2. Smart grid technologies

The European Commission's energy strategy aims to reduce greenhouse gas emissions by 80-95% of the 1990 level by 2050 [6]. Such transition toward low-carbon energy consumption will involve the increasing presence of renewable energy generation, like wind farms or photovoltaics systems, and the participation of electric vehicles (EVs). Besides growths in power demands, these new technologies impose additional challenges for DNOs. Distributed generation (DG) systems such as photovoltaics (PV) are connected to distribution networks, and during noon times, high PV penetration can exceed local demands, causing reverse power-flow and voltage-rise problems [7]. Uncontrolled EV charging might substantially raise peak loads on network assets such that overloads would happen sooner, resulting in more network component replacements [8]. Consequently, DNOs might need to continuously enhance the network capacity by installing more cables and building new substations to catch up with the development of peak loads. Such excessive network expansion is definitely undesirable and uneconomical because the duration of sharp peak loads is normally much shorter than that of lower base loads. It would be more efficient in use of existing capacity if parts of peak electricity consumption are shifted to off-peak hours, flattening the load profile.

Demand side management (DSM) and battery energy storage system (BESS) are smart grid technologies that can achieve such a peak-shaving effect. DSM involves incentivizing electricity consumers to schedule some of their energy consumption activities, e.g., heating, cooling, dish/cloth washing, or tumble drying, out of peak power demand periods [9]. BESS absorbs energy during off-peak hours and injects the stored energy into distribution networks during peak load hours [10]. The energy injection of BESS needs to be properly performed to satisfy local energy consumption without causing overloads on the network. BESS can also be used to store the surplus of PV penetration and to mitigate other network impacts due to sudden changes in PV output [7]. In this thesis, we assume that DNOs are allowed to employ DSM and BESS as alternatives to traditional assets like cables and transformers. It is of interest to investigate if smart grid technology can help DNOs to maintain the magnitude of peak loads within the current network capacity, and thereby, to postpone costly network expansions. Also, it could be that mixtures of both smart

grid technologies and traditional asset reinforcements are the better choices.

1.1.3. Multiple conflicting criteria

Besides minimizing the investment cost, DNOs often need to take other criteria into account when performing DNEP such as reducing energy losses or increasing network reliability. These objectives often conflict with each other, e.g., expansion plans that have low investment costs normally install thin electric cables, resulting in more energy losses, and vice versa. It is not expected that a single expansion plan exists that has the smallest cost, the least energy losses, and the highest reliability all at the same time. Usually, all these objectives are considered in the optimization process by capitalizing the non-financial objectives like energy losses and reliability and then combining them with the investment cost to form a single total cost objective function. Solving this single-objectivized problem yields the expansion plan with the least lump sum cost. This expansion plan, however, is not the overall optimal solution because a *utopian* solution that optimizes all objectives at the same time does not normally exist. Instead, the obtained expansion plan is only the best possible solution in the case that a DNEP practitioner considers only the total financial picture. If, for example, another DNEP practitioner finds network reliability more important, (s)he would expect a different expansion plan. Considering all possible preferences, we get a collection of equally-good expansion plans that are all optimal in the sense that any improvement in one objective would deteriorate other objectives. Such a collection of so-called *Pareto non-dominated solutions* is termed the *Pareto-optimal set*, which is the optimum of a *multi-objective optimization problem* (MOOP) [11]. Solutions not belonging to the Pareto-optimal set are called *dominated solutions*, which are not desirable because there exist other solutions that are strictly better than them in at least one objective and are not worse than them in all the remaining objectives. The goal of multi-objective optimization (MOO) is then to find the Pareto-optimal set out of all possible sets of solutions.

Instead of having a single objective value as in single-objective optimization, the quality of an MOOP candidate solution is indicated by a vector of multiple objective values. For an m-objective optimization problem with n decision variables, in addition to the n-dimensional decision variable space, we have the so-called m-dimensional *objective space* that contains the objective-value vector of all MOOP candidate solutions. The image of the Pareto-optimal set in the objective space is termed the *Pareto-optimal front*. Figure 1.1 shows an example of a Pareto-optimal front of Pareto non-dominated solutions for a multi-objective DNEP aiming to minimize investment cost and energy loss at the same time. Traversing the solutions along the Pareto-optimal front of a multi-objective DNEP problem informs DNOs about the most efficient ways to improve one objective at the cost of other objectives. For example, having the Pareto-optimal front can answer the question what are the best possible expansion plans to reduce energy losses if DNOs are willing to invest more in network enhancement, or how much network reliability and energy efficiency must be compromised if DNOs pursue less costly solutions. The results of solving DNEP as a multi-objective optimization problem are, therefore, much more

informative than those of solving DNEP as a single-objective problem. Because there potentially exist numerous non-dominated solutions on the Pareto-optimal front, obtaining exactly the whole front is prohibitively time-consuming. Instead, it is often expected to obtain an approximation set of solutions that is representative of the Pareto-optimal front in the sense that the range of the whole front should be covered, the approximate set should be as close as possible to the front, and the non-dominated solutions in the approximation set should be as diverse as possible (i.e., evenly distant from each other) [12]. Note that diversity here is defined in the objective space because the interesting trade-offs between all objectives under concern are expressed in the objective space.

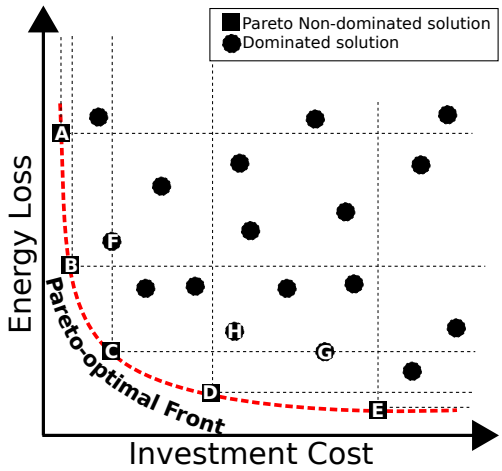

Figure 1.1: An example of a multi-objective DNEP with two objectives: minimizing investment cost and minimizing energy loss. Each solution corresponds to an expansion plan. Solution F costs as much as solution C but solution C has less energy loss → solution C *dominates* solution F. Solution G has the same amount of energy loss as solution C but solution C is more economical → solution G is *dominated* by solution C. Solution F is dominated by solution C, but is not dominated by solution D. Solution H is dominated by both solutions C and D. Solutions A, B, C, D, and E are not dominated by any solution, and are thus *Pareto non-dominated solutions*. If there are no other Pareto non-dominated solutions, then the set {A,B,C,D,E} is the *Pareto-optimal set* and its image in the objective space is the *Pareto-optimal front*.

Having presented the challenges in DNEP, we will discuss the algorithms that can be used to solve DNEP problems in the following section.

1.2. Optimization methods

DNEP is often regarded as being a hard problem. Basically, this classification comes from two sources: the complicated power flow constraint and objective functions, and the discrete nature of DNEP. First, the power flow constraint involves a set of non-linear power flow equations [13], and the formulations of energy losses on cables and transformers (considered an operation cost) have some quadratic terms [14]. Non-linearity can form many local optima in the search space, i.e., it can result in

non-convex search space. Second, all network asset types that can be considered for installation must be chosen from a collection of standardized devices. It is impossible to install a cable of an arbitrary thickness or a transformer of any fractional amount of capacity. Choices of network assets are then discrete (i.e., integer values), for which derivatives do not exist to guide the search. DNEP normally has a vast search space that grows exponentially with the network size. For example, suppose we have to decide whether or not to upgrade each existing cable in a network of n cable connections, we then have to find the optimal solution from a total of 2^n possible options. There currently exists no efficient algorithm to find the optimal expansion plan in general. It was proved in [15] that transmission network expansion planning (TNEP) is NP-hard. Although we have not learned about a dedicated complexity proof for DNEP yet and this thesis does not focus on complexity analysis either, it is conceivable that the complexity result of TNEP also applies to DNEP since the two problems have many features in common. In the following sections, we will introduce two classes of optimization algorithms that are commonly used to solve DNEP problems.

1.2.1. Mathematical programming methods

Mathematical optimization methods have a long history of research and real-world applications. Some of the most popular methods in discrete optimization are, e.g., branch and bound, cutting plane algorithms, and interior point methods [16]. Due to their long-established developments, they have been implemented in many professional/commercial optimization software packages, which can be employed to solve a wide range of real-world problems, including DNEP. Users normally are not involved in the operation of the solvers but need to write a mathematical program of the original problem that fits the the solvers' capability and the computation budget (e.g., time and computing resources). Writing proper mathematical programs, however, is not a trivial task. For example, the non-linear power flow equations need to be linearized so that DNEP is then modeled as a mixed-integer linear programming (MILP) problem, which can be solved by an MILP solver like CPLEX [17]. Similarly, if the objective function involves energy losses, quadratic terms in the formulation of the cost associated with energy losses need converting to linear terms by piecewise linear approximation since CPLEX is not efficient in handling quadratic objectives [14]. While it is possible to use a non-linear solver, e.g., a Sequential Quadratic Programming (SQP) solver, in combination with a branch and bound method to handle the discrete decision variables [18], such an approach in general still requires a convex approximation of the original non-convex problem for the sake of efficiency [19]. It can happen that the solutions obtained from solving models of linearized (or approximated) constraints might not be feasible if evaluated against the original non-linear constraints. The solutions must then be enhanced to attain feasibility, possibly resulting in poor solutions. DNEP can be modeled as a mixed-integer non-linear programming (MINLP) problem without any linearization as in [20], but it is then not guaranteed to find the globally optimal solution in an efficient manner [14] due to the non-convex search space. Writing proper mathematical models requires practitioners to have a certain degree of optimization and mathematics expertise.

Furthermore, if the problem is extended to include an additional constraint or to consider a new type of network component then the model needs to be reformulated. It is not always easy to formulate some distribution network constraints in mathematical forms that can be handled by the solver, e.g., the radiality constraint [21] or the reconfigurability constraint. Also, some new types of network component, such as battery energy storage systems, are currently on-going research and it would take time to reach a certain maturity before their structures in DNEP (e.g., interactions between peak power demand and battery charging/discharging) can be clearly understood and properly modeled to fit the solvers' capability.

To find the approximation set of non-dominated solutions for multi-objective optimization problems by mathematical programming methods, the weighted sum approach and the ϵ-constrained approach are frequently used [11, 22]. The weighted sum approach combines all (conflicting) objectives of interest into a single objective function by assigning each objective a coefficient. For example, a practitioner can define the a set of coefficients weighting the relative importance of energy losses and network reliability [23]. The original multi-objective problem is then transformed into a single-objective problem, which can be solved by available solvers to obtain a single non-dominated solution. To find other non-dominated solutions, different sets of coefficients need to be proposed and tried out. This weighted sum approach cannot obtain solutions on the non-convex parts of the Pareto-optimal front if such parts do exist [24]. The ϵ-constrained approach keeps one objective as the master objective and converts other objectives into problem constraints bounded by ϵ values, i.e., an objective function f_i which is not chosen as the master objective is converted into a problem constraint $f_i(\boldsymbol{x}) \leq \epsilon_i$ for the minimization case [11, 22]. Similarly, this single-objectively reformulated problem is then solved by available solvers to obtain a single solution, and different sets of ϵ values must be proposed to find other solutions. Both weighted sum and ϵ-constrained approaches require running the solver multiple times, each time with a different set of input parameters (i.e., a set of weight coefficients or ϵ values), which is time-consuming [22]. It is also not straightforward to generate these sets of input parameters in such a way that the set of finally obtained non-dominated solutions well-approximates the Pareto-optimal front. In other words, equally-distant coefficient sets might not correspond to evenly-spread non-dominated solutions.

1.2.2. Metaheuristics and evolutionary algorithms

Compared to mathematical programming approaches, metaheuristics offer practitioners a greater flexibility to handle, in general, the complexity of real-world optimization problems, and in particular, the uncertainty of future power system designs with the participation of many upcoming smart-grid technologies. Metaheuristics are loosely defined as algorithms that have some notable features as follows. First, metaheuristics require little or no problem-specific information (e.g., derivatives, continuity, convexity) in their operation apart from the solution evaluation function, which is naturally available for solving problems in a quantitative manner. Therefore, metaheuristics can be used to tackle a wide range of problems, and especially in black-box optimization, where problem structures are not available or

cannot be efficiently exploited. Second, if problem-specific information is known, it should be used to improve the solving performance, and the implementations of metaheuristics usually offer great flexibility to incorporate such domain knowledge effectively. Metaheuristics can then take the role of high-level frameworks to integrate expert knowledge, heuristics, and local search into the optimization process. Third, metaheuristics do not require the original problems to be oversimplified (linearized or approximated), and the obtained solutions are thus much more likely to be real-world feasible. The exact model of DNEP can be straightforwardly handled by metaheuristics [13]. For example, the original MINLP formulation of DNEP in [20] was solved by Genetic Algorithm (i.e., a metaheuristic algorithm) in [25]. Fourth, instead of processing a single solution, many metaheuristics employ a population of solutions. The simultaneously processing of multiple solutions induces an implicit parallel operation, approaching different local optima at the same time, which increases the probability that the finally obtained solution is the global optimum [11]. Furthermore, the population-based operation is well-suited for multi-objective optimization in terms of obtaining a set of different non-dominated solutions in one single optimization run. Based on the discussion here, we choose to employ metaheuristics for developing optimization engines to solve DNEP problems.

Metaheuristics have been widely applied in different optimization tasks for electric distribution networks [26]. Some of the most popular methods are, e.g., evolutionary algorithms (EAs [27]) in [28–32], Particle Swarm Optimization (PSO [33]) in [34, 35], Ant Colony Optimization (ACO [36]) in [37, 38], Differential Evolution (DE [39]) in [40, 41]. Many metaheuristics are inspired by phenomena in biology and nature, and biological terminologies are often borrowed to intuitively convey their operation mechanisms. For example, the Genetic Algorithm (GA [27]), i.e., an evolutionary algorithm, is often described using the terminology of heredity and natural selection. In essence, a GA starts from a collection of initial "guesses" about some possible solutions. The collection is referred to as *population* and the candidate solutions are termed *individuals*. Candidate solutions are represented as strings (or arrays) of values that encode the potential values for the problem decision variables. Solution strings are thus often called *chromosomes* and their constituent elements are termed *genes*. Each chromosome is associated with a *fitness* value, which corresponds to the optimization function value yielded by the candidate solution that chromosome encodes, quantitatively indicating how good the candidate solution is. A GA then operates in a sequence of generations until a predefined computing budget is used up or some other termination criteria are satisfied. Each generation, a number of promising solutions, often termed *parent* solutions, which have higher fitness values than others, are selected. New candidate solutions, often termed *offspring* solutions, are created then by performing some variation on the selected solutions, replacing the unselected ones. It is assumed that promising solutions contain good *building blocks* (i.e., partial solutions) and if variation is effective, better solutions can be synthesized by juxtaposing building blocks existing in the selected solutions. While the operation details of population-based metaheuristics widely differ from each other, their overall behaviors share some resemblance in the idea that the quality of solutions in subsequent iterations (generations) is gradually

improved by exchanging information/contents of promising solutions in previous iterations (generations). A simple GA implementation for DNEP can be outlined as in Figure 1.2.

A Simple Genetic Algorithm for DNEP	
1. [**Initialization**]	Randomly generate a population of n chromosomes, *i.e., n initial candidate solutions (network configurations).* Evaluate the fitness values of the initial chromosomes, *i.e., compute objective values and constraint violations.*
2. [**Selection**]	Based on fitness values, select the top $\Gamma\%$ chromosomes, *i.e., identify the best promising candidate networks.*
3. [**Crossover**]	With some crossover rate P_{cr}, recombine the selected (parent) chromosomes by exchanging their gene values to generate $n \times (1 - \Gamma/100)$ new (offspring) chromosomes, *i.e., exchange network assets between promising candidate networks to create new network configurations.*
4. [**Mutation**]	With some small mutation rate P_{mu}, alter the gene values of offspring chromosomes, *i.e., induce small changes in new candidate networks.*
5. [**Evaluation**]	Evaluate the fitness values of the offspring chromosomes, *i.e., compute objective values and constraint violations.*
6. [**Replacement**]	Replace the unselected chromosomes with the offspring, *i.e., replace low-quality solutions with new candidates.*
7. [**Termination**]	Go back to step 2 if computing budget still remains; otherwise, stop and report the best found solution.

Figure 1.2: A simple Genetic Algorithm solving DNEP

In this thesis, we particularly focus on evolutionary algorithms (EAs) but the methodologies presented here can be straightforwardly customized and incorporated with other metaheuristics. This is due to the facts that many other metaheuristics, e.g., PSO, ACO, or DE, have a population-based operation mechanism, and their variation operators, that alter currently existing solutions to create new candidate solutions, bear some resemblance to each other such that it is often possible to modify the ideas developed for one metaheuristic to work with other metaheuristics. Not aiming to cover the entire EAs corpus, we specifically state that the scope of EAs includes, but is not limited to, Genetic Algorithm (GA [27]), Estimation-of-Distribution Algorithm (EDA [42]), and Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA [43]), all of which target specifically at discrete variables. Note that each of these algorithms actually has different variants/implementations, which can be considered as different methods in their own respect. While having many advantages, there exist several challenges in the development and application of EAs and metaheuristics in industrial optimization, and particularly in DNEP, that need to be addressed.

1.3. Challenges in design and application of EAs

1.3.1. Parameter settings

While metaheuristics have been designed such that they can be straightforwardly employed by practitioners to solve their problems at hand, the current applications of metaheuristics in practice usually require users to spend a certain amount of time to do parameter tuning. For example, GA users need to set several parameters, such as the population size (i.e., the number of candidate solutions in each generation), the crossover rate (i.e., how often parent solutions are recombined to generate offspring), or the mutation rate (i.e., the probability that a decision variable can randomly change its value) [44] as shown in Figure 1.2. A PSO also needs a proper initialization for its control parameters: the swarm size, the acceleration coefficients, the inertia weight, the velocity clamping [45]. A typical ACO requires a number of parameters to be determined by practitioners: the number of ants, the initial pheromone values, the evaporation rate, the weights controlling the relative importance between the pheromone and heuristic information [36]. Similarly, a basic DE needs some user-defined parameters: the population size, the differential weight, and the crossover probability [39].

Proper control parameter setting is essential for the efficient performance of EAs and metaheuristics; otherwise the solvers operate inefficiently, resulting in slow convergence, or ineffectively, resulting in converging to subpar solutions and even leading to divergent or cyclic behavior in some cases. However, parameter setting is difficult because there exists no method to determine the optimal values of control parameters in general. Their suitable values depend on the mechanism of the solver being employed and also on the structure and the size of the specific problem instance under concern. In practice, users often need to manually try out different parameter values before obtaining acceptable results. To overcome this troublesome parameter setting problem, there exists a line of research into parameter adaptations for EAs [46]. Parameter adaptation schemes normally initialize control parameters with some random values or by following some guidelines, and then gradually change the parameters' values during the optimization process regarding the current status of the solver. The users are thus exempt from the requirement of parameter settings. In this thesis, we focus on eliminating the parameter setting for the class of evolutionary algorithms (EAs), and in particular, for GA, EDA, and GOMEA. These three EAs have a parameter in common, i.e., the population size, which can be considered as one, if not the most, crucial parameter of population-based EAs. If the population size is too small, there are not enough building blocks to synthesize the optimal solution, and the solver might suffer from premature convergence to some suboptimal solutions. If the population size is too large, the solver might overly diversify its search effort into many directions, and consequently, within a limited computing budget, promising regions of the search space cannot be reached due to lack of exploitation. Knowledge about proper population size settings for real-world applications is difficult to transfer from one problem instance to another because the same EA requires different population sizes for different problem instances. It is also inappropriate to employ the optimal population size of one EA solver for another because different EAs have different population size requirements even for

the same problem instance. This fact makes it difficult for EA users to reuse legacy results and troublesome for EA designers to compare the performance of algorithms in industrial optimization. In this thesis, besides handling other control parameters based on established research for each EA, we focus on a common scheme to get rid of this notoriously-difficult-to-set population size parameter. Our goal is twofold. Firstly, we aim to design parameter-less EAs that can be straightforwardly employed by non-EA-expert users to solve their optimization tasks. Secondly, by employing a common scheme for different EAs, we can put them on an equal footing, and thereby, conduct performance comparison in a fair manner to understand which solver is best-suited to DNEP problems.

1.3.2. Scalability

Most research on real-world applications of EAs focuses on the effectiveness of EAs to solve a specific problem while scalability issue is often omitted. Scalability, in essence, requires the employed solvers to maintain their effectiveness and efficiency when the problem size enlarges. For example, a non-scalable algorithm might be able to solve the DNEP problem for a small network, but it would need an exponentially larger amount of computing time even if the network size is linearly increased. Similar to the case of control-parameter settings, non-scalable solvers cannot be straightforwardly reused to tackle problem instances of larger sizes, and modifications in the solvers' design will thus be required. For combinatorial optimization as in the case of DNEP, the scalability of EAs typically depends on their capability for preserving and recombining good partial solutions (also known as *building blocks*) in the population to create new candidate solutions. Building blocks, however, are prone to disruption due to the stochastic nature of classic variation operators (e.g., crossover and mutation operators of the simple GA). For example, random crossovers of connected network configurations usually yield new network configurations of unconnected topologies, which are infeasible solutions. If the crossover operator takes account of the connectivity knowledge during recombination, network cables can be exchanged in such a way that the resulting networks are kept connected. Variation operators of EAs, therefore, are rarely used out-of-the-box in practice but are firstly customized with as much domain knowledge as possible so that they exploit problem-specific structures as much as possible. In this thesis, we will show how DNEP domain knowledge can be employed to modify the variation operators. The methodology then serves as a guideline for practitioners to adapt EAs to their specific needs.

While the specialization of EAs as aforementioned is often done by domain experts, from the perspective of (general-purpose) algorithm designers, or especially, black-box optimization, problem-specific knowledge is unavailable or difficult to be straightforwardly exploited. In such cases, a viable option is to infer problem structures from the candidate solutions in the working population of EAs. Usually, the structure of interest is the dependency structure that indicates which problem variables are dependent on each other and should thus be treated together when performing variation. Dependency structures can be obtained by employing statistical/machine learning techniques to fit a linkage model to the current population

in every generation. The learned linkage model is then used to guide the variation operators in creating new candidate solutions. Different types of linkage models can capture different types of dependency structures. Different EAs also have different capability in exploiting the learned model to efficiently generate new solutions. In this thesis, we consider three linkage models, namely the univariate model that assumes all problem variables are independent, the marginal product model that assumes problem variables form non-overlapping groups of dependency, and the linkage tree model that captures hierarchical dependencies. We will investigate which combination of linkage model-EA is best-suited to DNEP problems.

Designing scalable multi-objective evolutionary algorithms (MOEAs) has additional challenges to overcome. Instead of a single best solution, the optimum of a multi-objective optimization problem is the Pareto-optimal set, which might contain numerous (or even an infinite number of) trade-off alternatives. Then goal then is to find an approximation set that is representative of the Pareto-optimal front. Considering the limited size of the working populations of EAs (and of physical computer memory), many trade-off solutions must be discarded during the optimization process and we thus need a procedure to ensure that the remaining solutions are the best representatives. Also, the Pareto-optimal front might have a very wide range, and the search effort therefore must be equally divided into each direction so that the approximation set is not skewed toward some particular regions and the entire front can be evenly approached. Furthermore, the trade-off solutions located in various parts of the Pareto-optimal front might have different characteristics, and they can only be efficiently obtained if these local structures are learned and exploited in a dedicated manner. For example, it can be seen that the expansion plans which minimize investment costs often differ a lot from the plans that aim to decrease energy losses or increase network reliability (e.g., economical expansion plans typically favor the installations of small-diameter cables, which have lower acquisition costs and higher energy losses than cables of larger diameters). Considering the size of DNEP problems (i.e., the number of cable connections in large networks and the number of locations where smart grid technologies can be employed) and its complicated structure (i.e., the interactions between electric cables, network capacity, power demand, and peak-shaving effects of smart grid technologies), we need a scalable multi-objective optimization algorithm. In this thesis, we will construct a scalable parameter-less MOEA, which can be conveniently used by practitioners to solve multi-objective DNEP problems.

1.4. Research questions

In this section, we present the research questions that are addressed by this thesis and their related literature.

1. *How can we model distribution network expansion planning (DNEP) as an optimization problem such that the outcomes of solving this problem are practically relevant while the optimization models are computationally feasible?*

   The AC power flow model, that describes the operation of a distribution network, involves a system of non-linear equations characterizing the nodal power

balance [1]. These non-linear terms are often linearized, resulting in DC-like models, so that DNEP can be handled by mathematical programming solvers [47]. The quality and feasibility of solution expansion plans obtained from solving such simplified models are not guaranteed when being evaluated against the original non-linear AC power flow model. Infeasible solutions must then be repaired to attain the feasibility, which is a non-trivial task, regarding the fact that DNEP is a combinatorial problem where gradient information is not available to be exploited. Therefore, state-of-the-art DNEP research employs metaheuristic methods, e.g., EAs [48], PSO [49], or ACO [37], as the optimization solvers so that the AC power flow model can be directly taken into account and the feasibility of the solution expansion plans can be guaranteed. Nevertheless, the important reconfigurability constraint, which requires that the network can be re-configured to resume normal operation when failure occurs on a network cable [50], is not available in current optimization models [37, 48, 49]. The evaluation of the reconfigurability constraint is especially computationally expensive because the network operation must be checked against the failure of each network cable, which each corresponds to solving an AC power flow model. Reconfigurability is an essential requirement of distribution networks in urban areas, which typically supply electricity to a larger number of customers, and should thus be taken into account when solving DNEP.

The search space of DNEP, i.e., the set of all expansion plans, is prohibitively large, regarding the numerous possibilities of cable connections in combination with transformer upgrade at substations. Furthermore, the inclusion of smart grid technologies in DNEP increases the number of decision variables [34, 51], which enlarges the optimization model. To enhance the search efficiency, impractical or undesirable expansion options should be excluded from optimization models, e.g., substations that are far away from each other should not be connected, existing cables should not be replaced by new cables of lower capacities, or not all substations are suitable for installing storage systems.

The difficulties in modeling and solving DNEP are considerably increased when the investment/installation times of required assets during the planning period need to be determined (i.e., the dynamic DNEP), which is the real-world planning task that DNEP practitioners have to tackle. Traditionally, dynamic DNEP is solved by a two-phase approach [52, 53]: 1) solve the static DNEP problem to obtain a good (or near-optimal) expansion plan regarding the peak loads at the final year in the planning period, then 2) find a good (or near-optimal) installation schedule for the solution plan obtained in phase 1. Because of the facts that the cost of a dynamic plan (in which the required assets are installed along the planning period) is more economical than its corresponding static plan (that assumes all required assets to be installed at the same time) and that the time factors are ignored in phase 1, the result of this two-phase approach is typically far from optimal [54]. Recent DNEP research tackle directly the dynamic planning problem by encoding the installation schedule in the optimization model [31, 51]. The most common

approach is to represent the status of a network component in each year as a decision variable. Such overdetailed approach, however, suffers from poor scalability, e.g., DNEP over a period of 10 years for a network of 100 possible cable connections (i.e., both existing and potential cable connections) results in an optimization model of 1000 decision variables. For long-term DNEP (with planning periods of 10-30 years), proper modeling methods need to be developed. In this thesis, we will propose optimization models for both static and dynamic DNEP problems such that the models are computationally feasible without being oversimplified in order to obtain solution expansion plans that are practically relevant.

2. *How can we design scalable EAs for solving (multi-objective) real-world applications, and in particular for solving (multi-objective) DNEP problems?*

   While EAs can operate as black-box optimization algorithms that require little or no problem-specific knowledge, the capability in exploiting problem structures of the problem under concern is crucial to the scalability of an EA. EA solvers employed for solving DNEP are thus often customized with expert knowledge so that the search can be performed in an efficient manner [30, 55]. However, these customizations are not out-of-the-box features of EAs because they are specific to the DNO or the networks under concern. DNEP practitioners thus need to implement these customizations, and while EA solvers are typically highly customizable, modifying their mechanisms to incorporate expert knowledge is non-trivial (e.g., variation operators need re-designing). Therefore, in order to build a general EA framework for real-world applications like DNEP, it is highly beneficial to have a foundation EA that is capable of automatic recognition of problem structures during the EA run and then employs these learned structures to guide the search. Such an EA can be used out-of-the-box and can reasonably function in the general case, where problem-specific knowledge is not available or not straightforward to be exploited (e.g., like in the black-box optimization context).

   Linkage learning EAs (LLEAs) are a class of EAs that learn linkages (i.e., dependencies) between problem decision variables and then exploit these learned linkage information to inform the variation operators [56–58]. More specifically, problem variables that exhibit dependency on each other to some degree (i.e., a linkage set) should be treated together when existing solutions are varied to create offspring solutions (e.g., during solution recombination or mutation). Good partial solutions are thereby not disrupted too frequently and can be juxtaposed together to create high-quality solutions [43, 59]. While it has been shown that different classes of problems can be scalably solved if linkage information is properly exploited [58, 60, 61], applications of LLEAs in DNEP have been scarce so far. It is therefore worthwhile to investigate the applicability of LLEAs in DNEP, especially their out-of-the-box performance. Besides, LLEAs can also be customized with expert knowledge, which further enhances the efficiency of the search. Exploiting both linkage information and problem-specific knowledge is a promising methodology to address the

scalability issue in employing EAs for real-world optimization.

Elitism, which relates to maintaining the best solutions so far during an EA run, is essential to a steady convergence of the employed EA solver [62]. Instead of finding a single solution, the multi-objective optimization aims to obtain a set of multiple solutions that approximates the Pareto-optimal front exhibiting the trade-off relationship between the involved objectives [12]. While the size of the working population is limited, the number of trade-off solutions, which are equally good, can be numerous or even infinite. The implementation of elitism for multi-objective optimization is thus non-trivial [63]. A common and efficient elitism method is to employ a so-called *elitist archive* [64], which is an external population, to maintain an overview of non-dominated solutions obtained so far during the run.

Niching, which relates to maintaining diversity within the population in order to find multiple optima [65], is essential to obtaining trade-off solutions on different parts of the Pareto-optimal fronts [66]. Solutions on different parts of the Pareto-optimal fronts are typically different from each other, e.g., an expansion plan that has lower investment cost has different network topology and installed assets compared to an expansion plan that yields low energy losses. Therefore, it might not be beneficial to recombine such different solutions during variation. Similarly, a single linkage model is often not enough to sufficiently capture relevant dependencies pertaining to solutions in different parts of the fronts. Clustering-based operation is a straightforward implementation of niching and is often employed by state-of-the-art MOEAs [67, 68]. The working population is partitioned into multiple equal-sized clusters (in the objective space), and solution variation can then be tailored to each cluster [68] (e.g., linkage learning is performed per cluster, recombination is applied to solutions belonging to the same cluster). Each part of the Pareto-optimal front are thereby approached by an equal amount of search effort in a dedicated manner.

We pinpoint the components that are crucial to the scalability of (multi-objective) EAs, namely the exploitation of linkage information and problem-specific knowledge, elitism, and niching. In this thesis, we will show how to integrate these components for building scalable (multi-objective) EAs for real-world optimization, and for (multi-objective) DNEP in particular.

3. *How can we solve DNEP problems when multiple conflicting objectives need to be taken into account such that DNOs are provided with insight into the trade-off relationship between the involved objectives?*

   DNEP typically involves multiple planning criteria that conflict with each other, e.g., investment cost, energy losses, and network reliability. The most common approach is to capitalize and aggregate all those criteria into a single optimization function, for which the optimal solution corresponds with the most economical expansion plan with respect to the chosen capitalization parameters [31, 69, 70]. This approach requires DNEP practitioners to specify the prices of non-monetary criteria (e.g., how to capitalize energy losses or

network reliability). These user-input parameters, however, are not always available or might be subject to uncertainties. Because only the financial aspect is optimized and only the most economical expansion plan is obtained, trade-off relationships between the involved planning criteria, which typically provides DNOs with better insights into the network under concern, cannot be achieved with this aggregation approach. A generalization of this aggregation approach is the weighted sum method, in which each (capitalized) criterion is multiplied with a weighting factor and all the weighted criteria are then summed into an optimization function. Each vector of weighting factors thus yields a single-objective optimization problem that the optimal solution is also a Pareto-optimal solution. To approximate the Pareto-optimal front, multiple weight vectors are generated, often equally-spaced, to create multiple corresponding single-objective optimization problems, and each of these is then solved to obtain the associated Pareto-optimal solution. Several multi-objective DNEP problems have been tackled by this weighted sum method [71, 72]. However, obtaining a good approximation of the Pareto-optimal front by the weight sum method can be very time-consuming because separately solving each single-objective DNEP problem is still a hard optimization task due to its non-linearity, non-convexity, and non-differentiability [49]. Furthermore, equally-spaced weight vectors are not guaranteed to yield a well-spread approximation set, and if the Pareto-optimal front contains non-convex parts, solutions on such parts cannot be obtained by the weighted sum method [11].

The state-of-the-art methodology to expansion planning with conflicting criteria is to treat the involved criteria as separate optimization objectives and then to solve such a multi-objective optimization problem with true multi-objective optimization algorithms [22, 73]. Due to their advantages, multi-objective evolutionary algorithms (MOEAs) are often employed to solve multi-objective DNEP problems [30, 73], obtaining a set of multiple candidate expansion plans that approximate the trade-off relationship between the involved objectives, from which DNOs can investigate and select the plan corresponding with their desired trade-off. However, most applications of MOEAs in DNEP are performed in an ad hoc manner and the published literature overlooks many systematic issues as follows. A specific MOEA, normally some well-known MOEA like NSGA-II, is typically chosen to solve the problem instances at hand while the advantages and disadvantages of the employed MOEA in DNEP (compared with other alternative MOEAs) are often not discussed. Besides, the control parameters of MOEAs are often determined specifically for the networks under concern [30, 74] and are not guaranteed to yield efficient performance when being applied to other networks of different characteristics. Furthermore, MOEAs employed in DNEP literature are not the out-of-the-box variants but have been customized with expert/problem knowledge that might be specific to the involved DNO or the networks under concern [30, 73]. The efficiency enhancements of these customizations are often not reported, so that it is difficult to understand the performance of the original algorithm. Such information is important for building an EA framework for solving DNEP because

DNO-specific customizations would not be available in the more general case. In this thesis, we propose a systematic approach such that our methodology can serve as a general framework for tackling DNEP with multiple conflicting planning criteria.

4. *How can we eliminate the troublesome requirement of control parameter settings when applying (multi-objective) EAs in practice?*

   In general, there exist two methodologies to handle the setting of EA parameters in real-world applications: *parameter tuning* [75] and *parameter control* [44, 46]. Parameter tuning aims to obtain a good setting of the parameters of the employed EA by running multiple experiments with different combinations of parameter values on test problem instances, for which the optimal, or high-quality, solutions are known. The combination of parameter values that yields the best performance (among all the considered combinations) will then be employed for the actual optimization runs. Parameter tuning is commonly used in practice due to its straightforward implementation. However, parameter tuning is time-consuming because a sufficient number of experiments must be carried out regarding that the number of possible combinations of parameter values are numerous, or even infinite. In the context of DNEP, because the evaluation of a candidate expansion plan is computationally expensive, parameter tuning can take a prohibitive amount of time. Besides, if the distribution networks in the actual expansion planning have different network characteristics or larger sizes than the ones used for parameter tuning, the obtained parameter value results might not yield the desired performance.

   Instead of trying to find a good setting of EA parameters beforehand and using these obtained parameter values for the actual optimization run (where the parameter values remain fixed), parameter control methods start the optimization process with some initial parameter values and then adapt the parameter values during the run regarding the feedback from the search [44, 46]. Because the parameter values are adapted during the optimization run, practitioners are not required to configure EAs with good parameter values for the problem instance at hand before the run. Implementing parameter control methods, however, is non-trivial because original EA solvers often need to be re-designed to incorporate a parameter adaptation scheme. Besides, a parameter control method developed for a specific EA might not be compatible with other EAs because each EA has different operation mechanism and different parameters.

   A population-sizing-free scheme has been developed to remove the requirement of setting the population size parameter, which is a crucial parameter to the performance of all EAs [76, 77]. The scheme operates multiple populations of different sizes in an interleaved fashion such that each population is run for a number of generations before the population of the next larger size is run for one generation. A population is terminated when the average quality of its individuals is worse than that of a population having larger size. The scheme was firstly employed to remove the population size parameter setting for the simple GA [76], and later for the two EDAs Extended Compact Genetic Algorithm

(ECGA) [78] and hierarchical Bayesian Optimization Algorithm (hBOA) [79]. Compared to parameter tuning, this population-sizing-free scheme does not require a time-consuming overhead for determining a good population size before the actual optimization run. Compared to parameter control methods, this scheme has a straightforward implementation, which can be employed for any population-based EAs, and does not require EA solvers to be re-designed. The scheme was employed to remove the parameter settings of the simple GA and the ECGA solving a hypothetical DNEP problem variant, in which completely new distribution networks needed to be developed for regions that had no electricity previously [80]. However, the scheme was developed for single-objective EAs and thus needs certain modifications in order to be employed in the multi-objective optimization context, e.g., it is uninformative to compute the average quality of a multi-objective population. In this thesis, we will modify and investigate the practicality of this scheme in eliminating the requirement of control parameter settings when applying MOEAs to tackling the real-world multi-objective DNEP problem.

1.5. Outline of the thesis

The remainder of this thesis is organized as follows.

Chapter 2 introduces three typical EAs: Genetic Algorithm (GA), Estimation-of-Distribution Algorithm (EDA), and Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA). We then present three linkage models, namely the univariate model, the marginal product model, and the linkage tree model, that can be employed by EA solvers to match different types of dependency structures between problem variables. We show that each combination of EA and linkage model corresponds to an EA variant with some specific capability. Chapter 2 also introduces a population-sizing-free scheme that can be used with all population-based EAs, eliminating the requirement of tuning the population size parameter. Chapter 2 thus lays the foundation for addressing the linkage learning issue of the research question 2 and the parameter setting issue of the research question 4 for single-objective EAs.

Chapter 3 presents how to design a scalable multi-objective evolutionary algorithm (MOEA). Based on established research studies, we pinpoint the essential features of a robust MOEA, namely elitist archiving, population clustering, linkage learning, and exploiting the learned linkage to efficiently generate higher-quality candidate solutions. Following the guideline, we construct the Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm (MO-GOMEA). Experimental results on a wide range of benchmark problems suggest that MO-GOMEA has a superior scalability, compared to other available MO solvers. We then make MO-GOMEA a parameter-less MOEA so that MO-GOMEA can be straightforwardly employed to solve hard problems without the need of parameter tuning. Chapter 3 addresses the multi-objective optimization aspect of the research questions 2 and 4.

Chapter 4 formulates the static DNEP as an optimization problem: the objective, the constraints, and the decision variables. The vast combinatorial search space of the DNEP problem can be narrowed by considering expert knowledge to disregard impractical solutions. A random network generator is used to initialize the

population of EA solvers with good initial candidate solutions, and the efficiency of EAs in solving DNEP is thereby considerably improved. We propose how the general-purpose variation operators of EA solvers presented in Chapter 2 can be customized with domain knowledge to obtain different problem-specific operators. We then perform experiments for different benchmark networks, using all the proposed EA solvers with both out-of-the-box variants and customized variants. Chapter 4 addresses the modeling of realistic DNEP constraints issue stated in the research question 1 and also the exploitation of linkage information and expert knowledge issue stated in the research question 2.

Chapter 5 formulates the dynamic DNEP as an optimization problem. We propose a decomposition heuristic that can find an asset installation schedule for any feasible static expansion plan. Consequently, the dynamic DNEP can be solved by EA solvers available for the static DNEP, without compromising on their original performance. We then present how a smart-grid technology, i.e., battery energy storage systems (BESS), can be considered and modeled as network reinforcement options for DNOs, along with traditional electric cables. Experimental results show that BESS can be used to postpone costly cable expansions, and under certain conditions, mixtures of both electric cables and BESS are more economical than solely installing cables. In this chapter, we also briefly discuss the intrinsic limitations of single-objective optimization when having to deal with multiple conflicting criteria at the same time in DNEP. Chapter 5 addresses the dynamic DNEP with smart grid technologies stated by the research question 1 and partly indicates why the multi-criteria DNEP presented by the research question 3 must be properly answered.

Chapter 6 formulates the dynamic DNEP as a multi-objective optimization problem. We assume that DNOs can (financially) contribute to the demand side management (DSM) as a means to achieve the peak-shaving effect. Therefore, besides physical network asset expansion options, DSM is a policy option for DNOs to handle the growth of peak power demand. We then present some typical objectives that are of interest to DNOs: investment cost, DSM cost, energy losses, and network reliability (quantified as customer minutes lost). We solve the multi-objective DNEP for several benchmark networks with the proposed MO-GOMEA as the solver. For each case, the obtained result is a diverse set of equally good alternatives, informing DNOs about possible trade-offs between different objectives under concern. Experimental results show that DSM policies can be effective in delaying expensive physical network reinforcements and that MO-GOMEA can be used by DNOs to investigate the best possible options in various scenarios. Chapter 6 presents the answer to the research question 3.

Chapter 7 concludes the thesis. We recap the modeling and computational challenges in DNEP problems. We summarize our solutions to each of those challenges. We also mention several interesting (remaining) problems for future research. Finally, we present our insights about the design and application of evolutionary algorithms in general.

1.6. Publications

Parts of Chapters 2, 4, and 5 appeared in:
N.H. Luong, H. La Poutré, P.A.N. Bosman (2018). Exploiting Linkage Information and Problem-Specific Knowledge in Evolutionary Distribution Network Expansion Planning. *Evolutionary Computation (journal)*, vol. 26, no. 3. Publisher: MIT Press.

Parts of Chapter 3 appeared in:
N.H. Luong, P.A.N. Bosman (2012). Elitist Archiving for Multi-objective Evolutionary Algorithms: To Adapt or Not to Adapt. In: *Proceedings of the 12th International Conference on Parallel Problem Solving from Nature (PPSN '12)*. Publisher: Springer. pp. 72-81.
N.H. Luong, H. La Poutré, P.A.N. Bosman (2014). Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithms. In: *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*. Publisher: ACM. pp. 357-364.
N.H. Luong, H. La Poutré, P.A.N. Bosman (2018). Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm with the Interleaved Multi-start Scheme. *Swarm and Evolutionary Computation (journal)*, vol. 40. Publisher: Elsevier. pp. 238-254.

Parts of Chapters 4 and 5 appeared in:
N.H. Luong, M.O.W. Grond, P.A.N. Bosman, and H. La Poutré (2013). Medium-Voltage Distribution Network Expansion Planning with Gene-pool Optimal Mixing Evolutionary Algorithms. In: *Artificial Evolution 2013*. Publisher: Springer. pp. 93-105.
N.H. Luong, M.O.W. Grond, H. La Poutré, P.A.N. Bosman (2014). Efficiency Enhancements for Evolutionary Capacity Planning in Distribution Grids. In: *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*. Publisher: ACM. pp. 1189-1196.
M.O.W. Grond, **N.H. Luong**, J. Morren, P.A.N. Bosman, J.G. Slootweg, and H. La Poutré (2014). Practice-oriented Optimization of Distribution Network Planning using Metaheuristic Algorithms. In: *Power Systems Computation Conference (PSCC '14)*. Publisher: IEEE. pp. 1-8.
N.H. Luong, H. La Poutré, P.A.N. Bosman (2015). Exploiting Linkage Information and Problem-Specific Knowledge in Evolutionary Distribution Network Expansion Planning. In: *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15)*. Publisher: ACM. pp. 1231-1238.

Parts of Chapter 6 appeared in:
N.H. Luong, M.O.W. Grond, H. La Poutré, P.A.N. Bosman (2015). Scalable and Practical Multi-objective Distribution Network Expansion Planning. In: *Proceedings of Power & Energy Society General Meeting (PES-GM '15)*. Publisher: IEEE. pp. 1-5.

N.H. Luong, P.A.N. Bosman, M.O.W. Grond, H. La Poutré (2018). Evolutionary Multi-objective Dynamic Distribution Network Expansion Planning with Demand Side Management. In Book: *Application of Modern Heuristic Optimization Methods in Power and Energy Systems.* Publisher: IEEE/Wiley (accepted).

References

[1] J. J. Grainger and W. D. Stevenson, *Power system analysis* (McGraw-Hill, 1994).

[2] X.-P. Zhang, *Restructured Electric Power Systems: Analysis of Electricity Markets with Equilibrium Models* (John Wiley & Sons, 2010) p. 307.

[3] G. A. Maas, M. Bial, J. Fijalkowski, and Ucte, *UCTE Final report - System disturbance on 4 November 2006*, Tech. Rep. November (The Union for the Co-ordination of Transmission of Electricity, 2007).

[4] S. Abraham, H. Dhaliwal, R. J. Efford, L. J. Keen, A. McLellan, J. Manley, K. Vollman, N. J. Diaz, T. Ridge, and Others, *Final report on the August 14, 2003 blackout in the United states and Canada: causes and recommendations*, Tech. Rep. April (U.S.-Canada Power System Outage Task Force, 2004).

[5] J. G. Slootweg and P. Van Oirsouw, *Incorporating reliability calculations in routine network planning: Theory and practice,* in *18th International Conference and Exhibition on Electricity Distribution, CIRED 2005.* (IET, 2005) pp. 1–5.

[6] European Commission and Directorate-General for Energy, *Energy Roadmap 2050*, Tech. Rep. April (European Commission's communication, 2012) ISBN 978-92-79-21798-2 .

[7] M. J. E. Alam, K. M. Muttaqi, and D. Sutanto, *Mitigation of rooftop solar PV impacts and evening peak support by managing available capacity of distributed energy storage systems,* IEEE Transactions on Power Systems **28**, 3874 (2013).

[8] R. A. Verzijlbergh, M. O. W. Grond, Z. Lukszo, J. G. Slootweg, and M. D. Ilic, *Network impacts and cost savings of controlled EV charging,* IEEE Transactions on Smart Grid **3**, 1203 (2012).

[9] I. Laicane, D. Blumberga, A. Blumberga, and M. Rosa, *Reducing household electricity consumption through demand side management: The role of home appliance scheduling and peak load reduction,* Energy Procedia **72**, 222 (2015).

[10] E. Reihani, S. Sepasi, L. R. Roose, and M. Matsuura, *Energy management at the distribution grid using a Battery Energy Storage System (BESS),* International Journal of Electrical Power & Energy Systems **77**, 337 (2016).

[11] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms* (John Wiley & Sons, Inc., 2001).

[12] P. A. N. Bosman and D. Thierens, *The balance between proximity and diversity in multiobjective evolutionary algorithms,* IEEE Transactions on Evolutionary Computation **7**, 174 (2003).

[13] R. A. Jabr, *Polyhedral formulations and loop elimination constraints for distribution network expansion planning,* IEEE Transactions on Power Systems **28**, 1888 (2013).

[14] G. Munoz-Delgado, J. Contreras, and J. M. Arroyo, *Joint expansion planning of distributed generation and distribution networks,* IEEE Transactions on Power Systems **30**, 2579 (2015).

[15] D. Oertel and R. Ravi, *Complexity of transmission network expansion planning,* Energy Systems **5**, 179 (2013).

[16] L. A. Wolsey, *Integer programming* (Wiley, 1998).

[17] R. R. Gonçalves, J. F. Franco, and M. J. Rider, *Short-term expansion planning of radial electrical distribution systems using mixed-integer linear programming,* IET Generation, Transmission & Distribution **9**, 256 (2015).

[18] S. Leyffer, *Integrating SQP and branch-and-bound for mixed integer nonlinear programming,* Computational Optimization and Applications **18**, 295 (2001).

[19] S. Burer and A. N. Letchford, *Non-convex mixed-integer nonlinear programming: A survey,* Surveys in Operations Research and Management Science **17**, 97 (2012).

[20] W. El-Khattam, Y. Hegazy, and M. Salama, *An integrated distributed generation optimization model for distribution system planning,* IEEE Transactions on Power Systems **20**, 1158 (2005).

[21] M. Lavorato, J. F. Franco, M. J. Rider, and R. Romero, *Imposing radiality constraints in distribution system optimization problems,* IEEE Transactions on Power Systems **27**, 172 (2012).

[22] A. Alarcon-Rodriguez, G. Ault, and S. Galloway, *Multi-objective planning of distributed energy resources: A review of the state-of-the-art,* Renewable and Sustainable Energy Reviews **14**, 1353 (2010).

[23] P. Zhang, W. Li, and S. Wang, *Reliability-oriented distribution network reconfiguration considering uncertainties of data by interval analysis,* International Journal of Electrical Power & Energy Systems **34**, 138 (2012).

[24] I. Das and J. E. Dennis, *A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems,* Structural Optimization **14**, 63 (1997).

[25] E. Naderi, H. Seifi, and M. S. Sepasian, *A dynamic approach for distribution system planning considering distributed generation,* IEEE Transactions on Power Delivery **27**, 1313 (2012).

[26] A. R. Jordehi, *Optimisation of electric distribution systems: A review,* Renewable and Sustainable Energy Reviews **51**, 1088 (2015).

[27] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning* (Addison-Wesley Longman Publishing Co., Inc., 1989).

[28] V. Miranda, J. Ranito, and L. Proenca, *Genetic algorithms in optimal multistage distribution network planning,* IEEE Transactions on Power Systems **9**, 1927 (1994).

[29] I. J. Ramirez-Rosado and J. L. Bernal-Agustin, *Genetic algorithms applied to the design of large power distribution systems,* Power Systems, IEEE Transactions on **13**, 696 (1998).

[30] E. Carrano, L. Soares, R. Takahashi, R. Saldanha, and O. Neto, *Electric distribution network multiobjective design using a problem-specific genetic algorithm,* IEEE Transactions on Power Delivery **21**, 995 (2006).

[31] E. Carrano, R. Cardoso, R. Takahashi, C. Fonseca, and O. Neto, *Power distribution network expansion scheduling using dynamic programming genetic algorithm,* IET Generation, Transmission & Distribution **2**, 444 (2008).

[32] C. L. T. Borges and V. F. Martins, *Multistage expansion planning for active distribution networks under demand and distributed generation uncertainties,* International Journal of Electrical Power & Energy Systems **36**, 107 (2012).

[33] J. Kennedy and R. C. Eberhart, *Swarm Intelligence* (Morgan Kaufmann Publishers Inc., 2001).

[34] M. Sedghi, M. Aliakbar-Golkar, and M.-R. Haghifam, *Distribution network expansion considering distributed generation and storage units using modified PSO algorithm,* International Journal of Electrical Power & Energy Systems **52**, 221 (2013).

[35] J. Aghaei, K. M. Muttaqi, A. Azizivahed, and M. Gitizadeh, *Distribution expansion planning considering reliability and security of energy using modified PSO (Particle Swarm Optimization) algorithm,* Energy **65**, 398 (2014).

[36] M. Dorigo, M. Birattari, and T. Stützle, *Ant colony optimization,* IEEE Computational Intelligence Magazine **1**, 28 (2006).

[37] J. Gomez, H. Khodr, P. DeOliveira, L. Ocque, J. Yusta, R. Villasana, and A. Urdaneta, *Ant colony system algorithm for the planning of primary distribution circuits,* IEEE Transactions on Power Systems **19**, 996 (2004).

[38] C. Singh, *Reliability-constrained optimum placement of reclosers and distributed generators in distribution networks using an ant colony system algorithm,* IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **38**, 757 (2008).

[39] R. Storn and K. Price, *Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces,* Journal of Global Optimization **11**, 341 (1997).

[40] J.-P. Chiou, C.-F. Chang, and C.-T. Su, *Variable scaling hybrid differential evolution for solving network reconfiguration of distribution systems,* IEEE Transactions on Power Systems **20**, 668 (2005).

[41] L. Arya, S. Choube, and R. Arya, *Differential evolution applied for reliability optimization of radial distribution systems,* International Journal of Electrical Power & Energy Systems **33**, 271 (2011).

[42] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation* (Springer Science & Business Media, 2002) p. 382.

[43] D. Thierens and P. A. N. Bosman, *Optimal mixing evolutionary algorithms,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2011* (ACM Press, New York, New York, USA, 2011) pp. 617–624.

[44] A. Eiben, R. Hinterding, and Z. Michalewicz, *Parameter control in evolutionary algorithms,* IEEE Transactions on Evolutionary Computation **3**, 124 (1999).

[45] F. van den Bergh and A.P. Engelbrecht, *A study of particle swarm optimization particle trajectories,* Information Sciences **176**, 937 (2006).

[46] G. Karafotias, M. Hoogendoorn, and A. E. Eiben, *Parameter control in evolutionary algorithms: Trends and challenges,* IEEE Transactions on Evolutionary Computation **19**, 167 (2015).

[47] R. C. Lotero and J. Contreras, *Distribution system planning with reliability,* IEEE Transactions on Power Delivery **26**, 2552 (2011).

[48] S. M. Mazhari, H. Monsef, and R. Romero, *A hybrid heuristic and evolutionary algorithm for distribution substation planning,* IEEE Systems Journal **9**, 1396 (2015).

[49] S. Hasanvand, M. Nayeripour, and H. Fallahzadeh-Abarghouei, *A new distribution power system planning approach for distributed generations with respect to reliability assessment,* Journal of Renewable and Sustainable Energy **8**, 045501 (2016).

[50] H. L. Willis, *Power distribution planning reference book* (Marcel Dekker, 2004) p. 1217.

[51] H. Saboori, R. Hemmati, and V. Abbasi, *Multistage distribution network expansion planning considering the emerging energy storage systems,* Energy Conversion and Management **105**, 938 (2015).

[52] I. Ramirez-Rosado and T. Gonen, *Pseudodynamic planning for expansion of power distribution systems,* IEEE Transactions on Power Systems **6**, 245 (1991).

[53] M. Haghifam and M. Shahabi, *Optimal location and sizing of HV/MV substations in uncertainty load environment using genetic algorithm,* Electric Power Systems Research **63**, 37 (2002).

[54] M. G. Ippolito, G. Morana, E. R. Sanseverino, and F. Vuinovich, *Ant colony search algorithm for optimal strategical planning of electrical distribution systems expansion,* Applied Intelligence **23**, 139 (2005).

[55] E. Diaz-Dorado, J. Cidras, and E. Miguez, *Application of evolutionary algorithms for the planning of urban distribution networks of medium voltage,* Power Systems, IEEE Transactions on **17**, 879 (2002).

[56] P. A. N. Bosman and D. Thierens, *Linkage information processing in distribution estimation algorithms,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 1999* (1999) pp. 60–67.

[57] Y.-p. Chen, *Linkage Learning Genetic Algorithm,* in *Extending the Scalability of Linkage Learning Genetic Algorithms* (Springer-Verlag, Berlin/Heidelberg, 2005) pp. 35–43.

[58] M. Pelikan, M. W. Hauschild, and F. G. Lobo, *Estimation of Distribution Algorithms,* in *Springer Handbook of Computational Intelligence* (Springer Berlin Heidelberg, Berlin, Heidelberg, 2015) pp. 899–928.

[59] D. Thierens and D. E. Goldberg, *Mixing in genetic algorithms.* in *5th International Conference on Genetic Algorithms, ICGA 1993* (1993) pp. 38–47.

[60] P. A. N. Bosman and D. Thierens, *More concise and robust linkage learning by filtering and combining linkage hierarchies,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2013* (ACM Press, New York, New York, USA, 2013) pp. 359–366.

[61] K. L. Sadowski, P. A. Bosman, and D. Thierens, *On the usefulness of linkage processing for solving MAX-SAT,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2013* (ACM Press, New York, New York, USA, 2013) p. 853.

[62] K. A. De Jong, *Evolutionary computation: A unified approach* (MIT Press, 2006) p. 256.

[63] C. A. Coello Coello, G. Pulido, and E. Montes, *Current and future research trends in evolutionary multiobjective optimization,* in *Information Processing with Evolutionary Algorithms*, Advanced Information and Knowledge Processing (Springer London, 2005) pp. 213–231.

[64] J. Knowles and D. Corne, *Properties of an adaptive archiving algorithm for storing nondominated vectors,* IEEE Transactions on Evolutionary Computation **7**, 100 (2003).

[65] O. M. Shir, *Niching in evolutionary algorithms,* in *Handbook of Natural Computing* (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012) pp. 1035–1069.

[66] K. Sastry, M. Pelikan, and D. E. Goldberg, *Decomposable problems, niching, and scalability of multiobjective estimation of distribution algorithms. IlliGAL Report No. 2005004*, Tech. Rep. (University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2005).

[67] M. Pelikan, K. Sastry, and D. E. Goldberg, *Multiobjective hBOA, clustering, and scalability,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2015* (ACM Press, New York, New York, USA, 2005) pp. 663–670.

[68] P. A. N. Bosman, *The anticipated mean shift and cluster registration in mixture-based EDAs for multi-objective optimization,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2010* (ACM Press, New York, New York, USA, 2010) pp. 351–358.

[69] J. Salehi and M.-R. Haghifam, *Long term distribution network planning considering urbanity uncertainties,* International Journal of Electrical Power & Energy Systems **42**, 321 (2012).

[70] M. Ahmadigorji, N. Amjady, and S. Dehghan, *A novel two-stage evolutionary optimization method for multiyear expansion planning of distribution systems in presence of distributed generation,* Applied Soft Computing **52**, 1098 (2017).

[71] S. Ganguly, N. Sahoo, and D. Das, *Multi-objective planning of electrical distribution systems using dynamic programming,* International Journal of Electrical Power & Energy Systems **46**, 65 (2013).

[72] D. Kumar and S. Samantaray, *Design of an advanced electric power distribution systems using seeker optimization algorithm,* International Journal of Electrical Power & Energy Systems **63**, 196 (2014).

[73] S. M. Mazhari, H. Monsef, and R. Romero, *A multi-objective distribution system expansion planning incorporating customer choices on reliability,* IEEE Transactions on Power Systems **31**, 1330 (2016).

[74] F. Mendoza, J. Bernal-Agustin, and J. Dominguez-Navarro, *NSGA and SPEA applied to multiobjective design of power distribution systems,* IEEE Transactions on Power Systems **21**, 1938 (2006).

[75] A. Eiben and S. Smit, *Parameter tuning for configuring and analyzing evolutionary algorithms,* Swarm and Evolutionary Computation **1**, 19 (2011).

[76] G. R. Harik and F. G. Lobo, *A parameter-less genetic algorithm,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 1999* (Morgan Kaufmann, 1999) pp. 258–265.

[77] J. C. Pereira and F. G. Lobo, *A Java implementation of parameter-less evolutionary algorithms.* CoRR **abs/1506.0** (2015).

[78] C. F. Lima and F. G. Lobo, *Parameter-less optimization with the Extended Compact Genetic Algorithm and Iterated Local Search,* (Springer, Berlin, Heidelberg, 2004) pp. 1328–1339.

[79] M. Pelikan, A. Hartmann, and T.-K. Lin, *Parameter-less hierarchical Bayesian optimization algorithm,* in *Parameter Setting in Evolutionary Algorithms. Studies in Computational Intelligence.*, Vol. 54 (Springer Berlin Heidelberg, 2007) pp. 225–239.

[80] F. G. Lobo and D. E. Goldberg, *The parameter-less genetic algorithm in practice,* Information Sciences **167**, 217 (2004).

2

Model-Based Evolutionary Algorithms

Wie A zegt, moet ook B zeggen.
Who says A must say B.

Dutch proverb

Model-based evolutionary algorithms (MBEAs) differ from classical EAs in the usage of models to exploit problem structures during the optimization process. In this chapter, we focus on a major class of MBEAs that build models to capture the linkages (i.e., dependency structures) among problem variables and use the learned linkage models to inform variation operators in generating new candidate solutions. Linkage learning addresses the problem that stochastic variation operators of classical EAs are prone to disrupt building blocks (i.e., good partial solutions) due to their lack of linkage knowledge. Exploiting linkage information is therefore one of the biggest issues in EAs and is crucial to the scalability of EAs, especially when solving problems that exhibit efficiently exploitable linkage structures. We describe three types of linkage models, namely the univariate model, the marginal product model, and the linkage tree model, which have different capabilities of matching the problem variable dependency structure. We then outline the implementation of three MBEAs: the Genetic Algorithm, the Estimation-of-Distribution Algorithm, and the Gene-pool Optimal Mixing Evolutionary Algorithm. We show how different EA variants can be constructed by customizing an EA with various linkage models. Lastly, we describe a population-sizing-free scheme that can be employed by any population-based EAs so that practitioners are exempt from the requirement of setting the population size parameter.

Sections 2.2, 2.3, and 2.4 of this chapter have been published in [1].

2.1. Introduction

While evolutionary algorithms (EAs) have found numerous applications in various optimization tasks for electric distribution networks [2], it might not be easy for (non-EA) practitioners to choose a particular EA to tackle their specific problem at hand. The reason is that EAs are a very broad class of diverse solvers with different operation mechanisms, ranging from the simple Genetic Algorithm (GA [3]) with biologically-inspired operators to the more complex Estimation-of-Distribution Algorithms (EDAs [4]) that build and sample probabilistic graphical models for generating offspring [5]. Choosing suitable EAs is important since a problem instance can only be (efficiently) solved if the employed solver can (efficiently) exploit the problem structure. While GA is arguably the most popular EA in real-world applications, it is rarely used out-of-the-box without any modification. Instead, the simple GA is usually hybridized with problem-specific local search techniques or customized with domain expert knowledge (e.g., see [6, 7]). Such an adapted GA can be seen as a specialized solver for a specific problem. However, it is not always straightforward to exploit domain knowledge for solver customization, for example, if the problem involves complicated inter-dependent factors, like the operation problem of smart distribution grids that takes into account residential loads, power injections of distributed generation, electric vehicles [8]. If problem-specific knowledge is not available or cannot be modeled by the users, we say the problem is a black-box problem. When solving such black-box optimization problems, it is highly beneficial if problem structures can be detected along the optimization process. In this thesis, we advocate for the model-based evolutionary algorithms (MBEAs [9]), which differ from classical EAs in the explicit use of models that capture problem structures to guide the (stochastic) operators of EAs.

The advantages of MBEAs are manifold. First, because the model is explicitly defined, it is easier to see if the model of an EA solver can match the problem structure, and thereby, assess if that solver is suitable for solving the problem. Second, practitioners can directly design the model by using problem-specific knowledge so that the solver is adapted to the problem structure. Third, if domain knowledge is not available, the model can be learned from the working population of EAs by statistical methods or machine learning techniques. Furthermore, it is possible to replace the model type of an MBEA, creating a new EA variant that has different assumptions about the problem structure but the working mechanism of the original EA solver is still retained. While there exist many problem-specific types of structure, we focus on the (general-purpose) *linkage* structure (i.e., dependency) among decision variables, that indicates which variables have some degrees of dependency and should thus be treated together when generating new candidate solutions. Learning and exploiting linkage information benefit the search efficiency of EA solvers by protecting *building blocks* (i.e., good partial solutions) from being too frequently disrupted by EAs' stochastic operators, and thereby enhancing the juxtaposition of building blocks to form new (and potentially better) candidate solutions [10, 11]. To encode the linkage information that can be learned during the optimization process, we consider the general linkage model Family of Subsets (FOS [12]) and three specific types of FOS, namely univariate, marginal product,

and linkage tree. We then describe three MBEAs: GA, EDA, and Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA [12]). We will demonstrate how different EA variants can be created by mixing and matching an EA with different types of linkage model.

Since EAs are population-based optimization algorithms, the population size is one, if not the most, essential parameter. If the population size is too small, there might not be enough information (e.g., good partial solutions) in the population to synthesize the optimal solution. Also, small populations generally have low diversity, and due to the selection pressure, EA solvers will then prematurely converge to suboptimal solutions. On the other hand, if the population size is too large, EA solvers might overly explore the search space, which leads to inefficient performance. Consequently, the allowed computing time budget might be used up before any good solutions are found. Efficiency is of great importance, especially in real-world applications like DNEP, where the evaluations of candidate solutions are (computationally) expensive. Proper population size setting is, therefore, a necessary condition for EAs to achieve good performance. However, the optimal population size depends on the size of the specific problem instance at hand, on the problem structure, and also, on the particular components of the EA solver being employed. For example, in [12], experimental results showed that the minimally required population size for a solver to reliably solve a problem instance varies across problem sizes, problem structures, and working mechanisms of different EA variants. Therefore, in practice, it is nearly impossible to determine the optimal population size before actually solving the problem. Although there exist several *guidelines* of parameter settings for each EA, their effectiveness is not guaranteed when carried from laboratory benchmarks to industrial optimization tasks, or from one real-world problem to another. Practitioners often have to run their chosen EA solver multiple times with different population sizes before (accidentally) getting a value that yields acceptable results. In this thesis, we will consider a population-sizing-free scheme that was originally proposed for the parameter-less GA [13]. This scheme has been shown to be an effective method to eliminate the requirement of tuning the population size for different population-based EAs. We also present how to use the scheme as a framework for running experiments and providing a fair comparison of the performance of different EAs in solving real-world problems, where neither the optimal solutions nor the optimal population sizes are known.

The remainder of this chapter is organized as follows. Section 2.2 describes three linkage models and the type of dependency structure that each can capture. Section 2.3 outlines three EA solvers and how they can be customized with different linkage models to create multiple EA variants. Section 2.4 presents the population-sizing-free scheme that can be used to make all EA variants proposed in this chapter parameter-less. Finally, Section 2.5 concludes the chapter.

2.2. Problem structures and linkage models

A key part that characterizes an evolutionary algorithm (EA) is its variation operators (VOs), i.e., how new solutions are generated. VOs can be model-based, meaning that the way variation is performed is governed by a (learnable) model.

Models of particular interest are linkage models, which are used to encode the linkage information of the optimization problem instance at hand. A linkage model often contains information about groups of inter-dependent decision variables, also known as linkage sets. Variables in the same linkage set are dependent on each other in the sense that when being considered together, they have a significant contribution to the quality of a solution [12]. These variables should thus be jointly considered when performing variation. Variation operators can, for instance, make use of the information in linkage models to juxtapose partial solutions from existing solutions to generate new solutions. A linkage model that matches correctly with the problem dependency structure is crucial for variation operators to efficiently mix and preserve good partial solutions (i.e., building blocks) in the population in order to efficiently create high-quality solutions. Problem-specific knowledge (PSK), if available, can be used to directly construct the linkage model of the problem. If such valuable PSK is not available, or difficult to transcribe into linkage information, linkage information can be inferred from the working population of EAs by linkage learning (LL) procedures [12], in which problem variables having some degree of dependency are identified. In the following we give a general definition of a linkage model, called the Family of Subset (FOS) model, and subsequently describe three variants that can be used in practice.

2.2.1. Family of subset linkage model

Let $L = \{1, 2, \ldots, l\}$ be the set of all l decision variables of the problem instance at hand. We use the Family of Subsets (FOS [12]) concept to encode different linkage models. A FOS, denoted $\mathcal{F}$, consists of subsets of the set L, i.e., $\mathcal{F} = \{\boldsymbol{F}^1, \boldsymbol{F}^2, \ldots, \boldsymbol{F}^{|\mathcal{F}|}\}$ where $\boldsymbol{F}^i \subseteq L,\ i \in \{1, 2, \ldots |\mathcal{F}|\}$. Thus, $\mathcal{F} \subseteq \mathcal{P}(L)$, i.e., $\mathcal{F}$ is a subset of the power set of L. Each subset $\boldsymbol{F}^i$ is a linkage set containing decision variables that exhibit some degree of dependency and should thus be jointly treated when performing variation. A FOS $\mathcal{F}$ is said to be complete if every problem variable is contained in at least one linkage set, i.e., $\forall i \in L,\ \exists j \in \{1, 2, \ldots, |\mathcal{F}|\} :\ i \in \boldsymbol{F}^j$. The completeness property ensures that all problem variables are considered when performing variation following linkage sets in $\mathcal{F}$. We here consider three complete FOS models: univariate, marginal product, and linkage tree.

2.2.2. Univariate model

The univariate factorization (UF) model contains only singleton sets. It therefore expresses the assumption that all problem variables are independent from each other, i.e., $\boldsymbol{F}^i = \{i\},\ i \in L = \{1, 2, \ldots, l\}$. Because there is only one possible configuration, the univariate model does not require any linkage learning.

2.2.3. Marginal product model

The marginal product (MP) model is a partitioning of the set L of all problem variables, i.e., $\bigcup_{\boldsymbol{F}^i \in \mathcal{F}} \boldsymbol{F}^i = L$ and $\forall \boldsymbol{F}^i, \boldsymbol{F}^j \in \mathcal{F}, \boldsymbol{F}^i \neq \boldsymbol{F}^j :\ \boldsymbol{F}^i \cap \boldsymbol{F}^j = \emptyset$. Variables in the same linkage set have some degree of dependency while variables in different linkage sets are considered to be independent [12]. The well-known Extended Compact Genetic Algorithm (ECGA [14]) employs the MP as its linkage model.

The MP model is often learned from a population $\mathcal{P}$ of n individuals using a greedy algorithm that optimizes a model scoring metric as follows. First, the FOS $\mathcal{F}$ assumes the univariate structure and is then scored by the chosen metric. Next, all possible merges of two linkage sets $\boldsymbol{F}^i$ and $\boldsymbol{F}^j$ are tried out and the merge that improves the scoring metric the most is chosen. The merged linkage set is added into $\mathcal{F}$, replacing the two constituent linkage sets, i.e., $\mathcal{F} \leftarrow (\mathcal{F} \setminus \{\boldsymbol{F}^i, \boldsymbol{F}^j\}) \cup \{\boldsymbol{F}^i \cup \boldsymbol{F}^j\}$. This procedure continues until no merging event can improve the scoring metric anymore. The scoring metric employed by ECGA is named the Combined Complexity Criterion (CCC), which is the sum of the Compressed Population Complexity and the Model Complexity [14]. CCC needs to be minimized. The Compressed Population Complexity (CPC) is the cost of representing the whole population of n individuals with FOS model $\mathcal{F}$ and is calculated as

$$CPC = n \sum_{i=1}^{|\mathcal{F}|} H(X_{\boldsymbol{F}^i}) \tag{2.1}$$

where $X_{\boldsymbol{F}^i}$ is a set of random variables, in which each random variable X_k corresponds with a problem variable x_k in the linkage set $\boldsymbol{F}^i$. The entropy of a random variable measures the uncertainty associated with that random variable or, quantitatively speaking, the number of bits required to describe that random variable [15]. $H(X_{\boldsymbol{F}^i})$ is the joint entropy of the marginal distribution of $X_{\boldsymbol{F}^i}$. We have

$$H(X_{\boldsymbol{F}^i}) = -\sum_{\boldsymbol{x} \in \Omega(X_{\boldsymbol{F}^i})} P(X_{\boldsymbol{F}^i} = \boldsymbol{x}) \log_2 P(X_{\boldsymbol{F}^i} = \boldsymbol{x}) \tag{2.2}$$

where $\Omega(X_{\boldsymbol{F}^i})$ is the the sample space for random variables $X_{\boldsymbol{F}^i}$, i.e., all possible string values of the problem variables in $\boldsymbol{F}^i$ when being jointly considered. The probability $P(X_{\boldsymbol{F}^i} = \boldsymbol{x})$ can be computed by counting the frequency of $\boldsymbol{x}$ in the population $\mathcal{P}$. Because the entropy of a collection of random variables is less than or equal to the sum of individual entropies [15], minimizing CPC favors FOS models that contain large linkage sets. In contrast, minimizing the Model Complexity (MC) favors simpler FOS models that contain smaller linkage sets [16]. The MC is the cost of representing or storing FOS model $\mathcal{F}$ and is calculated as

$$MC = \log_2(n+1) \sum_{i=1}^{|\mathcal{F}|} (|\Omega(X_{\boldsymbol{F}^i})| - 1) \tag{2.3}$$

The greedy model building algorithm, which has a worst-case runtime $\mathcal{O}(nl^3)$, needs to minimize the CCC metric. Instead of computing this scoring metric for the whole FOS $\mathcal{F}$ at every merging trial, the CCC metric improvement (i.e., CCC metric decrease) can be computed more efficiently considering only problem variables in the two involved linkage sets $\boldsymbol{F}^i$ and $\boldsymbol{F}^j$ [12] as

$$\begin{aligned} CCC(\boldsymbol{F}^i, \boldsymbol{F}^j) = n[H(X_{\boldsymbol{F}^i}) + H(X_{\boldsymbol{F}^j}) - H(X_{\boldsymbol{F}^i \cup \boldsymbol{F}^j})] + \\ \log_2(n+1)[(|\Omega(X_{\boldsymbol{F}^i})| - 1) + (|\Omega(X_{\boldsymbol{F}^j})| - 1) - (|\Omega(X_{\boldsymbol{F}^i \cup \boldsymbol{F}^j})| - 1)] \end{aligned} \tag{2.4}$$

2.2.4. Linkage tree model

Because each configuration of the MP model is a partitioning of the set L, every problem variable can exist in only one linkage set. Therefore, any two variables are either dependent (if they are in the same linkage set) or independent (if they are in different linkage sets). In the LT model, a problem variable can exist in multiple linkage sets. The LT model is thus more expressive than the MP model in the sense that any two variables can be both dependent and independent. While the MP model has a flat structure, the LT model arranges its linkage sets in a hierarchical tree structure. The lowest level (i.e., leaf nodes) contains univariate linkage sets of each variable separately, i.e., $\boldsymbol{F}^i = \{i\}, i \in L = \{1, 2, \ldots, l\}$. Higher levels contain multivariate linkage sets, in which each linkage set $\boldsymbol{F}^i$ is formed by merging two mutually exclusive linkage sets $\boldsymbol{F}^j$ and $\boldsymbol{F}^k$ at lower levels, i.e., $\boldsymbol{F}^j \cap \boldsymbol{F}^k = \emptyset, |\boldsymbol{F}^j| < |\boldsymbol{F}^i|, |\boldsymbol{F}^k| < |\boldsymbol{F}^i|$ and $\boldsymbol{F}^j \cup \boldsymbol{F}^k = \boldsymbol{F}^i$. The highest level (i.e., the root node) contains the set L itself. Thus, an LT can encode different levels of dependency, from the totally independent state in the leaf nodes to the all-dependent state in the root node [17]. An LT over a set of problem variables $\{1, 2, 3, 4, 5, 6, 7, 8\}$ can be, e.g., $\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{1, 3\}, \{2, 5\}, \{4, 6\}, \{1, 3, 7\}, \{4, 6, 8\}, \{2, 4, 5, 6, 8\}, \{1, 2, 3, 4, 5, 6, 7, 8\}\}$, which is visualized as a binary tree with the root node in Figure 2.1. An LT model configuration for a set L of l problem variables is thus a FOS $\mathcal{F}$ of exactly $2l - 1$ linkage sets.

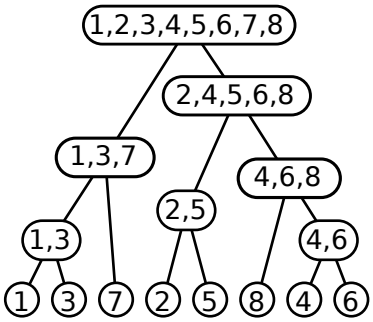

Figure 2.1: An example linkage tree with the root node for a problem of eight decision variables.

The LT model can be learned from the population of n individuals by a hierarchical clustering procedure named the Unweighted Pair Group Method with Arithmetic Mean (UPGMA). The FOS $\mathcal{F}$ is built in a bottom-up manner. First, $\mathcal{F}$ is initialized with l univariate linkage sets $\boldsymbol{F}^i = \{i\}, i \in L = \{1, 2, \ldots, l\}$. Then, UPGMA iteratively merges the two linkage sets $\boldsymbol{F}^i$ and $\boldsymbol{F}^j$ that are the most similar. The newly created linkage set $\boldsymbol{F}^i \cup \boldsymbol{F}^j$ is added to the $\mathcal{F}$. The two constituents linkage sets $\boldsymbol{F}^i$ and $\boldsymbol{F}^j$ are still kept in the LT $\boldsymbol{F}$ but they are not considered for merging anymore. The merging operations continue until the root node (i.e., the set L itself) is created. To calculate the similarity between two linkage sets $\boldsymbol{F}^i$ and $\boldsymbol{F}^j$, we take the average of the mutual information (MI) over all pairs of problem variables (X, Y) where $X \in \boldsymbol{F}^i$ and $Y \in \boldsymbol{F}^j$ [17] as follows

$$MI^{UPGMA}(\boldsymbol{F}^i, \boldsymbol{F}^j) = \frac{1}{|\boldsymbol{F}^i||\boldsymbol{F}^j|} \sum_{X \in \boldsymbol{F}^i} \sum_{Y \in \boldsymbol{F}^j} MI(X, Y) \tag{2.5}$$

where $MI(X,Y) = H(X) + H(Y) - H(X,Y)$. $MI(X,Y)$ measures the mutual dependence between the two random variables X and Y [18]. UPGMA can be optimally implemented by using the reciprocal nearest-neighbor chain technique such that an LT can be built in $\mathcal{O}(nl^2)$ time [19].

2.3. Evolutionary algorithms

2.3.1. Genetic Algorithm

```
GA //population size n
 1 for i ∈ {1,2,...,n} do
 2     P_i ← CreateRandomSolution()
 3     EvaluateFitness(P_i)
 4 while ¬TerminationCriteriaSatisfied() do
 5     F ← LearnModelFromPopulation(P)
 6     π ← RandomPermutation({1,2,...,n})
 7     k ← 0
 8     for i ∈ {1,2,...,n} do
 9         O_i ← Recombine(P_{π_k}, P_{π_{k+1}})
10         EvaluateFitness(O_i)
11         k ← k + 2
12         if k ≥ n − 1 then
13             k ← 0
14             π ← RandomPermutation({1,2,...,n})
15     P ← TournamentSelection(P + O, n, 4)
```

```
Recombine(p^0, p^1)
 1 for i ∈ {1,2,...,|F|} do
 2     if Random01() < 0.5 then
 3         o_{F^i} ← p^0_{F^i}
 4     else
 5         o_{F^i} ← p^1_{F^i}
 6 Return(o)
```

Figure 2.2: Genetic Algorithm

The Genetic Algorithm (GA) is started with a population $\mathcal{P}$ of n randomly generated candidate solutions. Next, for every generation, a FOS $\mathcal{F}$ is learned from the current population $\mathcal{P}$. An offspring population $\mathcal{O}$ of n new solutions is created from $\mathcal{P}$ by performing recombination (i.e., crossover), guided by the linkage information in $\mathcal{F}$, n times on two parent solutions that are randomly picked each time, giving one offspring solution. Then, both the parent population $\mathcal{P}$ and the offspring population $\mathcal{O}$ are combined into a selection pool $\mathcal{P} + \mathcal{O}$ of $2n$ solutions in total. Tournament selection with tournament size 4 is performed on this pool to

select n survivor solutions, which form the new parent population $\mathcal{P}$ for the next generation, ensuring convergence by logistic growth of the best solution [12]. If the univariate factorization (UF) model is chosen for linkage learning, we automatically have the simple GA with uniform crossover, in which all dependencies among problem variables are disregarded. If the marginal product (MP) model is employed, we have a GA variant with a recombination operator that can exchange blocks of values at the positions indicated by the linkage sets in $\mathcal{F}$. We do not combine the LT model with GA because the simple crossover operator is not compatible with the hierarchical structure of an LT FOS. Figure 2.2 outlines the pseudo-code for our GA implementation. Even though GA does not originally have a model, Figure 2.2 shows that the GA can easily be represented as a model-based EA (MBEA). In particular, we here can create two specific MBEA variants: GA-UF and GA-MP.

Note that a different implementation of GA exists, in which two offspring solutions are created from two parent solutions in every recombination event. In this thesis, recombining two parent solutions results in one offspring. We use this implementation so that GA can be presented in sync with both EDA and GOMEA in the sense that one offspring solution is constructed each time. There is no significant difference in performance if two offspring were to be generated in recombination.

2.3.2. Estimation-of-Distribution Algorithm

The Estimation-of-Distribution Algorithm (EDA) starts with a population $\mathcal{P}$ of n randomly generated candidate solutions. Every generation, a FOS $\mathcal{F}$ is learned from the current population $\mathcal{P}$. The EDA also derives a probability distribution from the population $\mathcal{P}$ following the obtained structure $\mathcal{F}$. In the case of the UF model or the MP model, all linkage sets of $\mathcal{F}$ are mutually exclusive, so the probability distribution can be formulated as $P_{\mathcal{F}}(\boldsymbol{X}) = \prod_{i=1}^{|\mathcal{F}|} P_{\mathcal{F}}(X_{\boldsymbol{F}^i})$ where a random variable X_i corresponds with each problem variable x_i. Each linkage set $\boldsymbol{F}^i$ corresponds to a marginal $X_{\boldsymbol{F}^i}$ whose distribution $P_{\mathcal{F}}(X_{\boldsymbol{F}^i})$ can be estimated by counting frequencies of all possible value strings of the variables in $\boldsymbol{F}^i$ in the population $\mathcal{P}$. Note that for the LT model such a joint distribution $P_{\mathcal{F}}(\boldsymbol{X})$ cannot be defined directly in terms of $P_{\mathcal{F}}(X_{\boldsymbol{F}^i})$. The EDA does not use the recombination operator (i.e., crossover) like the GA to create offspring from existing solutions. Instead, the offspring population $\mathcal{O}$ of n new solutions is generated by sampling the estimated probability distribution. The selection pool $\mathcal{P} + \mathcal{O}$ of $2n$ solutions is again created by combining $\mathcal{P}$ and $\mathcal{O}$. A tournament selection with tournament size 4 is performed on $\mathcal{P} + \mathcal{O}$ to select n survivors, forming the new population $\mathcal{P}$ for the next generation. If the UF model is employed, we have an EDA-UF that corresponds with the Univariate Marginal Distribution Algorithm (UMDA) [20]. If the MP model is used, we have an EDA-MP, which can be considered to be similar to the ECGA [14]. Figure 2.3 shows pseudo-code for the EDA.

2.3.3. Gene-pool Optimal Mixing Evolutionary Algorithm

The Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) starts with a population $\mathcal{P}$ of n randomly generated candidate solutions. Every generation, the linkage model building procedure is performed on $\mathcal{P}$ to construct a FOS $\mathcal{F}$. Using

```
EDA //population size n
1 for i ∈ {1, 2, ..., n} do
2     P_i ← CreateRandomSolution()
3     EvaluateFitness(P_i)
4 while ¬TerminationCriteriaSatisfied() do
5     F, P_F(X) ← LearnDistributionFromPopulation(P)
6     for i ∈ {1, 2, ..., n} do
7         O_i ← SampleDistribution()
8         EvaluateFitness(O_i)
9     P ← TournamentSelection(P + O, n, 4)
```

```
SampleDistribution
1 for i ∈ {1, 2, ..., |F|} do
2     o_{F^i} ← SampleSubsetDistribution(F^i, P_F(X_{F^i}))
3 Return(o)
```

Figure 2.3: Estimation-of-Distribution Algorithm

the obtained FOS $\mathcal{F}$, GOMEA transforms each existing parent solution $\boldsymbol{p} \in \mathcal{P}$ into a new offspring solution $\boldsymbol{o} \in \mathcal{O}$ whose fitness value is equal to or better than the fitness value of $\boldsymbol{p}$. The offspring population $\mathcal{O}$ completely replaces $\mathcal{P}$ and becomes the new parent population $\mathcal{P}$ for the next generation. Figure 2.4 shows pseudo-code for GOMEA.

Instead of fully creating new solutions and then evaluating them like in GA, the variation operator of GOMEA, called Gene-pool Optimal Mixing (GOM) [12], uses the learned FOS to evolve each existing parent $\boldsymbol{p}$ into a new offspring $\boldsymbol{o}$ in an iterative manner. First, $\boldsymbol{o}$ and a backup $\boldsymbol{b}$ are cloned directly from $\boldsymbol{p}$. Then, each linkage set in the FOS $\mathcal{F}$ is traversed iteratively in a random order. For each linkage set, a donor $\boldsymbol{d}$ is randomly selected from the current population $\mathcal{P}$. If the values of the donor $\boldsymbol{d}$ for the variables indicated by the linkage set differ from those in $\boldsymbol{o}$ in at least one position, these values are copied from $\boldsymbol{d}$ into $\boldsymbol{o}$. This partially-altered solution $\boldsymbol{o}$ is evaluated and compared against its backup $\boldsymbol{b}$. If $\boldsymbol{o}$ is equally good or better than $\boldsymbol{b}$ (i.e., $fitness[\boldsymbol{o}] \geq fitness[\boldsymbol{b}]$), the changes are accepted (i.e., the values are copied from $\boldsymbol{d}$) and updated into $\boldsymbol{b}$ as well. Otherwise, the changes are undone and $\boldsymbol{o}$ reverts to its backup state $\boldsymbol{b}$. Note that the acceptance of solutions having equal fitness can be beneficial to move across a fitness plateau [17]. It can be seen that each linkage set corresponds with a mixing event, in which the current solution is recombined with a random donor solution and the variables in the same linkage set are treated together, preserving the BB structure (insofar correctly represented by the FOS). After traversing the whole FOS, an offspring $\boldsymbol{o}$ is then fully constructed, replacing the original parent $\boldsymbol{p}$ in the next generation. Note that GOMEA does not need to perform selection over the combined pool $\mathcal{P} + \mathcal{O}$ like GA or EDA because the GOM operator ensures that the quality of the offspring solution is better or at least equal to the parent solution.

It can happen that GOM cannot improve the current parent solution $\boldsymbol{p}$ or that, because of a significant plateau, GOM keeps transforming back and forth solu-

tions of different genotypes but with the same fitness value. To overcome this, if GOM cannot yield a new offspring or when the number of subsequent generations that the best solution at the end of generation t $\boldsymbol{x}^{best}(t)$ does not change, i.e., the no-improvement stretch (NIS), exceeds a certain threshold, we invoke the Forced Improvement (FI) procedure [17]. In essence, FI is similar to GOM but we always use $\boldsymbol{x}^{elitist}$ as the only donor solution. $\boldsymbol{x}^{elitist}$ is the best-found-so-far solution, which is constantly checked for possible updates every time a fitness evaluation is performed. FI only accepts the mixing event that results in a strict improvement (i.e., $fitness[\boldsymbol{o}] > fitness[\boldsymbol{b}]$) and FI stops as soon as such mixing event occurs Previous research on GOMEA suggests a threshold for NIS of $1 + \lfloor \log_{10}(n) \rfloor$ [17]. If FI does not succeed in improving $\boldsymbol{p}$, $\boldsymbol{x}^{elitist}$ is returned as the new offspring.

The GOMEA-LT with the linkage tree model is the most popular variant of GOMEA, and is also known as the Linkage Tree Genetic Algorithm (LTGA) [12]. The concept of constructing an offspring in a step-wise manner by iteratively improving a parent solution is compatible with the hierarchical structure of linkage trees. Note that the linkage set associated with the root node of the linkage tree (i.e., the set of all problem variable indices) can be disregarded because it assumes that all variables should be jointly copied when performing variation, which means no new solution is created. GOMEA can also be straightforwardly combined with the UF model (which we will refer to as GOMEA-UF) or the MP model (which we will refer to as GOMEA-MP). Note that the local search-like mechanism of the GOM operator causes GOMEA to use more fitness evaluations per generation compared to the GA or the EDA. However, it has been shown that, for problems that are efficiently solvable with correctly detected linkage structures, GOMEA has population sizing requirements that are much smaller than those of GA and EDA, resulting in far fewer necessary generations and a more efficient overall performance [12].

2.4. Parameter-less evolutionary algorithms

EAs are population-based solvers and, in most cases, the population size is an essential parameter that needs to be set properly in order to obtain good performance. Moreover, population size often has a strong impact on the performance of an EA. Therefore, it could be argued that to ensure fair comparisons of the performance of different EAs in solving a problem, the optimal population size of each solver should be determined and employed. A suitable population size setting depends on the structure of the specific problem instance at hand and also on the EA being used. For a real-world application with complicated structure, unknown optimal solutions, and a computationally expensive evaluation function, bisection methods to find the minimally-required population size are not practically feasible. Practitioners often need to experiment with different population sizes in an *ad hoc* manner to find a setting that works well, given the amount of time available to solve the problem. Parameter tuning for each solver and every problem instance is inefficient and its results are non-transferable to other cases. Here, we therefore employ a scheme that can eliminate the requirement of setting a fixed population size for population-based EAs.

```
GOMEA //population size n
 1 for i ∈ {1, 2, ..., n} do
 2     P_i ← CreateRandomSolution()
 3     EvaluateFitness(P_i)
 4 x^best(0) ← arg min_{x∈P}{fitness[x]}; t ← 0; t^NIS ← 0
 5 while ¬TerminationCriteriaSatisfied() do
 6     F ← LearnModelFromPopulation(P)
 7     for i ∈ {1, 2, ..., n} do
 8         O_i ← GenepoolOptimalMixing(P_i)
 9     P ← O
10     t ← t + 1
11     x^best(t) ← arg min_{x∈P}{fitness[x]}
12     if fitness[x^best(t)] > fitness[x^best(t − 1)] then
13         t^NIS ← 0
14     else
15         t^NIS ← t^NIS + 1
```

```
GenepoolOptimalMixing(p)
 1 b ← o ← p; fitness[b] ← fitness[o] ← fitness[p];
 2 changed ← false
 3 for i ∈ {1, 2, ..., |F|} in a random order do
 4     d ← Random({P_1, P_2, ..., P_n})
 5     o ← CopyValues(o, d, F^i)
 6     if o ≠ b then
 7         EvaluateFitness(o)
 8         if fitness[o] ≥ fitness[b] then
 9             b ← o; fitness[b] ← fitness[o]; changed ← true
10         else
11             o ← b; fitness[o] ← fitness[b]
12 if ¬changed or t^NIS > 1 + ⌊log_10(n)⌋ then
13     changed ← false
14     for i ∈ {1, 2, ..., |F|} in a random order do
15         o ← CopyValues(o, x^elitist, F^i)
16         if o ≠ b then
17             EvaluateFitness(o)
18             if fitness[o] > fitness[b] then
19                 b ← o; fitness[b] ← fitness[o]; changed ← true
20             else
21                 o ← b; fitness[o] ← fitness[b]
22         if changed then breakfor
23 if ¬changed then
24     o ← x^elitist; fitness[o] ← fitness[x^elitist]
25 Return(o)
```

```
CopyValues(x, d, F^i)
1 o ← x
2 for k ∈ F^i do
3     o_k ← d_k
4 Return(o)
```

Figure 2.4: Gene-pool Optimal Mixing Evolutionary Algorithm

The parameter-less GA (P-GA) with a population-sizing-free scheme was firstly proposed in [13]. The scheme was then revised and employed with the hierarchical Bayesian Optimization Algorithm [21]. Recently, the revised implementation of the scheme has been used as a framework for several parameter-less EAs [22]. We refer to this population-sizing-free mechanism as the Harik-Lobo scheme in reference to the original authors that first proposed the idea. In essence, we run multiple instances of the EA in parallel. Each instance has a different population size but larger populations have a slower generational cycle. We start with the first population P_1 of some small size n_1. Then, by doubling the population size, the next population P_i is twice as large as the previous one, i.e., $n_i = 2n_{i-1}$ for $i > 1$. All the populations are scheduled with the principle that for every b generations of population P_i, one generation of population P_{i+1} is run. If P_{i+1} does not exist yet, it will be initialized before running its first iteration. Having no maximum population size, the EA runs and grows its populations until the computing time budget is used up. The pseudo-code for the Harik-Lobo scheme is given in Figure 2.5.

```
POPULATION-SIZING-FREE FRAMEWORK
 1  P_1 ← INITIALIZENEWPOPULATION(n_1)
 2  generation[1] ← 0
 3  max_population_index ← 1
 4  i ← 1
 5  while ¬TERMINATIONCRITERIASATISFIED() do
 6      EXECUTEONEGENERATION(P_i)
 7      generation[i] ← generation[i] + 1
 8      if generation[i] mod b = 0 then
 9          i ← i + 1
10          if i > max_population_index then
11              P_i ← INITIALIZENEWPOPULATION(2n_{i-1})
12              generation[i] ← 0
13              max_population_index ← i
14      else
15          i ← 1
```

Figure 2.5: Harik-Lobo Population-Sizing-Free Framework [22]

The generation base $b = 4$ is recommended, which indicates that smaller populations are allocated more solution evaluations than larger populations [13, 22]. In [21, 23], a different generation base $b = 2$ is used, which ensures that all populations are given the same number of evaluations. Because the parameter-less GA did not implement the mutation operator, any converged population, whose individuals become identical, will be terminated as no new offspring can be generated. Also, a population P_i can be considered inefficient if its average fitness is worse than that of a larger population P_j, for $j > i$. Once a smaller population is "overtaken" by a larger one as such, the smaller population should thus be terminated. The reason behind this is that even though P_j is started after P_i, because P_j has a larger size, with likely more diversity and thus better odds at finding high-quality solutions than P_i, we don't need to run P_i anymore. Therefore, P_i and all other smaller populations

P_k, for $k < i$, will be terminated. It can be seen that all population-based EAs can be straightforwardly put into the Harik-Lobo scheme without any major modifications to their implementation. Consequently, when combined with the Harik-Lobo scheme, all EA variants presented in this chapter become parameter-less EAs since, apart from the population size, we have well-tuned other parameters by considering established research.

2.5. Conclusions

In this chapter, we used the Family of Subset (FOS) concept as the basis to construct model-based evolutionary algorithms (MBEAs). An MBEA can be systematically composed by addressing two questions: 1) what kind of model the solver employs to match the problem structure; 2) how the solver exploits the model to generate new candidate solutions to bias the search toward promising regions in the search space. We described three linkage models that can be used by EAs to match the dependency structure of the problem at hand as follows. The univariate factorization (UF) model assumes that all variables are independent from each other. The marginal product (MP) model assumes that problem variables can form non-overlapping linkage sets where dependencies exist among constituent variables of each group. The linkage tree (LT) model assumes hierarchical dependencies in which any two variables are either dependent or independent according to different levels of linkage sets. These linkage models can be learned from the working population of EAs to derive a specific configuration in each generation. We then presented the model-based implementation of three EAs, namely the Genetic Algorithm (GA), the Estimation-of-Distribution Algorithm (EDA), and the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA). We showed how their typical variation operators can be guided by the learned linkage model to create offspring solutions: GA with the classical crossover, EDA with the model sampling, and GOMEA with the Gene-pool Optimal Mixing. EA solvers of different capabilities can be synthesized by combing an EA with a linkage model. Particularly, we created seven EA solvers: GA-UF, GA-MP, EDA-UF, EDA-MP, GOMEA-UF, GOMEA-MP, and GOMEA-LT.

Lastly, we discussed parameter-tuning problems of EAs, especially the problem of finding an appropriate population size for a specific EA to efficiently solve a specific problem instance at hand. We described the Harik-Lobo scheme, which has previously been shown to be an effective method to eliminate the notoriously-difficult-to-set population size parameter. We argued that EAs should be designed parameter-less so that EAs can be straightforwardly employed by practitioners without the need of parameter tuning.

References

[1] N. H. Luong, H. La Poutré, and P. A. N. Bosman, *Exploiting linkage information and problem-specific knowledge in evolutionary distribution network expansion planning,* Evolutionary Computation **26** (2018).

[2] A. R. Jordehi, *Optimisation of electric distribution systems: A review,* Renewable and Sustainable Energy Reviews **51**, 1088 (2015).

[3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning* (Addison-Wesley Longman Publishing Co., Inc., 1989).

[4] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation* (Springer Science & Business Media, 2002) p. 382.

[5] P. Larrañaga, H. Karshenas, C. Bielza, and R. Santana, *A review on probabilistic graphical models in evolutionary computation,* Journal of Heuristics **18**, 795 (2012).

[6] C. Wang and Y. Gao, *Determination of power distribution network configuration using non-revisiting genetic algorithm,* IEEE Transactions on Power Systems **28**, 3638 (2013).

[7] E. Carrano, R. Takahashi, E. Cardoso, R. Saldanha, and O. Neto, *Optimal substation location and energy distribution network design using a hybrid GA-BFGS algorithm,* IEE Proceedings - Generation, Transmission and Distribution **152**, 919 (2005).

[8] A. Zakariazadeh, S. Jadid, and P. Siano, *Integrated operation of electric vehicles and renewable generation in a smart distribution system,* Energy Conversion and Management **89**, 99 (2015).

[9] D. Thierens and P. A. Bosman, *Model-based evolutionary algorithms,* in *Proceedings Companion of the Genetic and Evolutionary Computation Conference - GECCO 2015* (ACM Press, New York, New York, USA, 2015) pp. 93–120.

[10] D. Thierens and D. E. Goldberg, *Mixing in genetic algorithms.* in *5th International Conference on Genetic Algorithms, ICGA 1993* (1993) pp. 38–47.

[11] D. Thierens, *The linkage tree genetic algorithm,* in *Parallel Problem Solving from Nature, PPSN XI* (2010) pp. 264–273.

[12] D. Thierens and P. A. N. Bosman, *Optimal mixing evolutionary algorithms,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2011* (ACM Press, New York, New York, USA, 2011) pp. 617–624.

[13] G. R. Harik and F. G. Lobo, *A parameter-less genetic algorithm,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 1999* (Morgan Kaufmann, 1999) pp. 258–265.

[14] G. R. Harik, F. G. Lobo, and K. Sastry, *Linkage learning via probabilistic modeling in the Extended Compact Genetic Algorithm (ECGA),* in *Scalable Optimization via Probabilistic Modeling*, Studies in Computational Intelligence, Vol. 33, edited by M. Pelikan, K. Sastry, and E. Cantú-Paz (Springer Berlin Heidelberg, Berlin, Heidelberg, 2006) pp. 39–61.

[15] A. Dembo, T. M. Cover, and J. A. Thomas, *Information theoretic inequalities,* IEEE Transactions on Information Theory **37**, 1501 (1991).

[16] M. Pelikan, K. Sastry, and E. Cantú-Paz, eds., *Scalable Optimization via Probabilistic Modeling*, Studies in Computational Intelligence, Vol. 33 (Springer, 2006).

[17] P. A. N. Bosman and D. Thierens, *More concise and robust linkage learning by filtering and combining linkage hierarchies,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2013* (ACM Press, New York, New York, USA, 2013) pp. 359–366.

[18] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)* (Wiley-Interscience, 2006).

[19] I. Gronau and S. Moran, *Optimal implementations of UPGMA and other common clustering algorithms,* Information Processing Letters **104**, 205 (2007).

[20] H. Mühlenbein, *The equation for response to selection and its use for prediction,* Evolutionary Computation **5**, 303 (1997).

[21] M. Pelikan, A. Hartmann, and T.-K. Lin, *Parameter-less hierarchical Bayesian optimization algorithm,* in *Parameter Setting in Evolutionary Algorithms. Studies in Computational Intelligence.*, Vol. 54 (Springer Berlin Heidelberg, 2007) pp. 225–239.

[22] J. C. Pereira and F. G. Lobo, *A Java implementation of parameter-less evolutionary algorithms.* CoRR **abs/1506.0** (2015).

[23] M. Pelikan and F. G. Lobo, *Parameter-less genetic algorithm: A worst-case time and space complexity analysis. IlliGAL Report No. 99014*, Tech. Rep. (University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL., 1999).

3

Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithms

Wie op twee hazen tegelijk jaagt, vangt geen van beide.
Who goes after two hares at the same time will catch neither of them.

Dutch proverb

In this chapter, we bring the strengths of the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) to the multi-objective (MO) optimization realm. We modify the linkage learning procedure and the variation operator in GOMEA to better suit the need of approximating the Pareto-optimal front rather than finding a single best solution. Based on state-of-the-art studies on MOEAs, we further pinpoint and incorporate two other components that are essential for a scalable MO optimizer. First, the use of an elitist archive of non-dominated solutions is beneficial for keeping track of the non-dominated front when the main population size is limited. Second, clustering techniques can be crucial if different parts of the Pareto-optimal front need to be handled differently. By combining these elements, we create a multi-objective GOMEA (MO-GOMEA). Experimental results on various MO optimization problems confirm the capability and scalability of our MO-GOMEA that compare favorably with those of the well-known multi-objective Genetic Algorithm NSGA-II and the more recently introduced multi-objective Estimation-of-Distribution Algorithm mohBOA. We then modify MO-GOMEA to remove the need to specify important parameters a priori, namely the population size and the number of clusters, without severely compromising the performance of MO-GOMEA. This makes MO-GOMEA convenient to use for practitioners, which is important for real-world applications.

Parts of this chapter have been presented at *GECCO '14* [1] and published in [2].

3.1. Introduction & background

3.1.1. Multi-objective optimization & evolutionary algorithms

Many real-world optimization problems involve two or more conflicting objectives (i.e., the optimization function is then a vector function). A typical example is when we need to draw up a manufacturing plan that maximizes the quality of a product and, at the same time, minimizes its production cost. For such a multi-objective problem, a single *utopian* solution that simultaneously optimizes all objectives does not exist. Instead, the optimum is a set of equally favorable trade-off alternatives that are all optimal in the sense that an improvement in any objective degrades other objectives. This chapter is about the design of an efficient and practical evolutionary algorithm, especially for solving the class of multi-objective optimization problems that exhibit certain efficiently exploitable structures. Considering the combinatorial nature of the Distribution Network Expansion Planning (DNEP) model in this thesis, we focus on the class of discrete optimization problems.

A multi-objective optimization problem consists of m objective functions $f_i(\boldsymbol{x}), i \in \{1, 2, \ldots, m\}$ that, without loss of generality, all need to be maximized with $\boldsymbol{x}$ as decision vector. We assume that solutions $\boldsymbol{x}$'s for the combinatorial optimization problem at hand involve l decision variables that comprise a discrete search space. In particular, we focus on Cartesian search spaces, meaning that for each variable we have a domain (e.g., a set of integers) and the space of entire solutions is the Cartesian product of these individual domains. We here restrict each variable domain to the binary domain $\mathbb{B} = \{0, 1\}$, but all methodologies presented in this chapter can be easily extended to higher cardinality (see Chapter 6). A solution $\boldsymbol{x}$ can then be represented as a binary vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_l) \in \boldsymbol{\Omega} = \times_{i=1}^{l} \mathbb{B}$. The vector of objective values of $\boldsymbol{x}$ is $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$.

Optimality in multi-objective optimization is defined by employing Pareto concepts (see, e.g., [3]).

1. **Pareto dominance**. A solution $\boldsymbol{x}$ *Pareto dominates* a solution $\boldsymbol{x}'$ (denoted $\boldsymbol{x} \succ \boldsymbol{x}'$) if and only if $\forall i \in \{1, 2, \ldots, m\} : f_i(\boldsymbol{x}) \geq f_i(\boldsymbol{x}') \wedge \boldsymbol{f}(\boldsymbol{x}) \neq \boldsymbol{f}(\boldsymbol{x}')$.
2. **Pareto optimality**. A solution $\boldsymbol{x}$ is said to be *Pareto optimal* if and only if $\neg\exists \boldsymbol{x}' : \boldsymbol{x}' \succ \boldsymbol{x}$.
3. **Non-dominated set**. A set $\boldsymbol{\mathcal{P}}$ is called a non-dominated set if and only if $\neg\exists \boldsymbol{x}, \boldsymbol{x}' \in \boldsymbol{\mathcal{P}} : \boldsymbol{x} \succ \boldsymbol{x}'$.
4. **Non-dominated front**. The set of the objective value vectors of all solutions in a non-dominated set $\boldsymbol{\mathcal{P}}$ is called the non-dominated front $\boldsymbol{f}(\boldsymbol{\mathcal{P}})$ of $\boldsymbol{\mathcal{P}}$: $\boldsymbol{f}(\boldsymbol{\mathcal{P}}) = \{\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})) \mid \boldsymbol{x} \in \boldsymbol{\mathcal{P}}\}$.
5. **Pareto-optimal set**. The set $\boldsymbol{\mathcal{P}_S}$ of all Pareto-optimal solutions: $\boldsymbol{\mathcal{P}_S} = \{\boldsymbol{x} | \neg\exists \boldsymbol{x}' : \boldsymbol{x}' \succ \boldsymbol{x}\}$.
6. **Pareto-optimal front**. The set $\boldsymbol{\mathcal{P}_F}$ is the set of the objective value vectors of all Pareto-optimal solutions in $\boldsymbol{\mathcal{P}_S}$: $\boldsymbol{\mathcal{P}_F} = \{\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})) \mid \boldsymbol{x} \in \boldsymbol{\mathcal{P}_S}\}$.

Multi-objective evolutionary algorithms (MOEAs) have been widely used for solving multi-objective optimization problems [4, 5]. Because the number of solutions on the Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$ can be numerous (or even infinite in case of, e.g., continuous optimization), most MOEAs focus on finding a non-dominated set that yields a good approximation of the Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$. The quality of approximations is often assessed based on both proximity to the optimal front (i.e., as close as possible) and diversity along the front (i.e., as well-spread as possible) [5, 6]. Commonly studied MOEAs, such as the Nondominated Sorting Genetic Algorithm II (NSGA-II) [7], the improved Strength Pareto Evolutionary Algorithm (SPEA2) [8], the Multiobjective Evolutionary Algorithm based on Decomposition (MOEA/D) [9], and the Nondominated Sorting Genetic Algorithm III (NSGA-III, especially for problems with more than three objectives) [10], have been demonstrated to be effective in achieving this two-fold goal for a wide range of problems. However, the efficiency and the usability of MOEAs remain two challenging research topics.

3.1.2. Efficiency of MOEAs

Scalability, linkage learning, and gene-pool optimal mixing

The important issue of scalability is often overlooked in multi-objective optimization research [11]. Corresponding to the general algorithmic notion of computational complexity, scalability requires that optimization algorithms maintain their effectiveness and efficiency when the problem size (e.g., the number of decision variables) increases, leading to a use of resources (i.e., time, memory) that scales only as a (low-order) polynomial function. For combinatorial EAs, their scalability is typically highly dependent on their capability for mixing and preserving *building blocks* (i.e., good partial solutions) in the population to create new solutions [12]. Simple variation operators (e.g., uniform/1-/2-point crossover and mutation) of classic MOEAs normally need to be customized with problem-specific expert knowledge so that they respect problem structures and do not disrupt building blocks too often during the optimization process (e.g., see [13]). However, domain knowledge is not always straightforward to be exploited or might even be unavailable as in the case of black-box optimization. Without respecting dependency structures between problem variables (e.g., by using classic crossover and mutation operators), MOEAs cannot solve some decomposable problems efficiently [11, 14]. Addressing this scalability issue of general-purpose MOEAs, there exist multi-objective Estimation-of-Distribution Algorithms (MOEDAs) that replace classic variation operators with model-based variation operators [11, 14, 15]. (MO)EDAs use probabilistic models to estimate the distribution of promising solutions and then sample the learned models, which contain information about variable dependencies, to generate new candidate solutions [16]. Exemplary MOEDAs are the Multi-objective Adapted Maximum-Likelihood Model (MAMaLGaM) [17] for continuous variables and the Multi-objective Hierarchical Bayesian Optimization Algorithm (mohBOA) [11] for discrete variables.

Although MOEDAs are robust optimizers, similar to EDAs in single-objective optimization, the probabilistic model building procedures typically require larger

population sizes and more CPU time as a result of large computational complexity for probabilistic model building [15]. Furthermore, the estimation of complete probability distributions may be unnecessary if *linkage information* (i.e., knowledge about which problem variables should be jointly copied during recombination) by itself suffices to perform variation effectively. Gene-pool Optimal Mixing Evolutionary Algorithms (GOMEAs) [12] are a class of state-of-the-art single-objective EAs that focus on learning and exploiting linkage information. GOMEAs have been found to efficiently and reliably solve a variety of well-known benchmark problems with far smaller population size requirements and much better scalability compared to classic GAs and EDAs [12]. Chapters 4 and 5 of this thesis also show that GOMEAs outperform GAs and EDAs when solving the single-objective DNEP, a complicated real-world optimization problem. Moreover, certain classes of linkage models in GOMEAs can be learned in $\mathcal{O}(Nl^2)$ time [18] whereas learning comparable higher-order probabilistic models in EDAs typically requires $\mathcal{O}(Nl^3)$ time (where l is the number of decision variables and N is the population size). It is these strengths of GOMEAs that we attempt to transfer to multi-objective optimization, ultimately aiming to tackle the multi-objective version of the DNEP problem.

Elitist archiving
State-of-the-art MOEAs are characterized by implementations of the elitism concept [19] in the context of multi-objective optimization (i.e., when there exist multiple equally good trade-off solutions). A common realization of elitism is obtained by the use of a secondary population, termed the elitist archive, for retaining non-dominated solutions found so far during the search. An elitist archive can be beneficial because the sizes for the main population may be smaller than the number of solutions on the Pareto-optimal front. Especially in such cases, some non-dominated solutions can be lost due to selection [20].

Objective-space clustering
The goal of multi-objective optimization is two-fold: finding an approximation set of non-dominated solutions that is both close to the Pareto-optimal front (i.e., proximity) and as diverse as possible (i.e., diversity, especially in the objective space) [6, 21]. Standard MOEAs steer the population toward the optimal front while trying to preserve diversity by different mechanisms, such as selection based on crowding distance in NSGA-II [7] or the environmental selection in SPEA2 [8]. However, it has been shown that these mechanisms are insufficient for achieving good scalability and that different parts of the optimal front should be processed separately [11]. State-of-the-art MOEDAs therefore often implement mixture probability distributions by clustering the selected solutions in the objective space and building a model for each cluster separately (e.g., in mohBOA [11] and in MAMaLGaM [17]). Studies [11, 22] have noted the difficulty for finding the entire optimal front of some decomposable problems. This difficulty holds especially for the *extreme* regions of the optimal front, as for the studied problems the number of solutions that map to the vicinity of the extremes becomes exponentially smaller compared to the *middle* part of the Pareto-optimal front. Furthermore, for multi-objective optimization in general, selection tries to exploit all objectives simultaneously, thus reducing the

pressure towards approaching the optimal front [17]. This problem was solved in MAMaLGaM-X$^+$ [17] by adding a separate single-objective optimizer for each objective and injecting the best solutions found by these single-objective optimizers into the elitist archive. We adopt the idea of clustering from MAMaLGaM-X$^+$ but adapt the process to fit well with GOMEA.

3.1.3. Usability of MOEAs

State-of-the-art MOEAs requires users to set two important parameters that influence its performance: its population size and the number of clusters into which the population will be partitioned. While population size setting is crucial for any population-based EA, it is difficult to determine the optimal population size beforehand in practice. Normally, EAs cannot solve problems well when using populations of inadequate sizes. If the population size is too big, EAs may overly diversify their search efforts and the allowed budget of fitness evaluations or running time is used up before good solutions are obtained. Practitioners often need to run EAs many times with different parameter settings in a trial-and-error manner. Several mechanisms have been proposed to eliminate the population size setting for single-objective EAs with promising results, such as [23, 24]. Similar works, however, are not established for MOEAs. For clustering-based MOEAs, the number-of-clusters parameter ever adds another layer of complexity in terms of usability, since this parameter influences the cluster size, which needs to be sufficient so that the probabilistic/linkage model of each cluster can be properly learned. We argue that complicated parameter settings undermine the usability of MOEAs and therefore should be eliminated.

3.1.4. Contributions

Based on the aforementioned established research on the components that are crucial for the scalability of MOEAs, namely elitist archiving, population clustering, and linkage learning, we construct the multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm (MO-GOMEA). We then make MO-GOMEA an easy-to-use MOEA by removing the requirement to set key parameters: the population size and the number of clusters. The remainder of this chapter is organized as follows. Section 3.2 describes the key components of MO-GOMEA. Section 3.3 introduces all benchmark problems that we use to perform experiments. Section 3.4 describes the indicator that is used to assess the performance of MOEAs. Section 3.5 shows the performance of MO-GOMEA under optimal (or favorable) parameter settings. Section 3.6 shows how the requirement for setting the population size parameter and the number-of-clusters parameter of MO-GOMEA can be eliminated. Section 3.7 demonstrates the performance of MO-GOMEA and the influence of mutation operators. Section 3.8 compares the performance of MO-GOMEA with a version of NSGA-II in which the population size parameter is eliminated similarly and further discusses the parameter-less scheme for EAs in the multi-objective optimization context. Section 3.9 discusses what makes MO-GOMEA a scalable and practical MOEA and concludes the chapter.

3.2. Multi-objective GOMEA

MO-GOMEA is started by randomly initializing a population $\mathcal{P}$ of N candidate solutions. All N solutions are evaluated to obtain their objective values. The population $\mathcal{P}$ is then clustered (in the objective space) into k clusters $\mathcal{C}_j$'s ($j \in \{1, 2, \ldots, k\}$) of equal sizes (see Section 3.2.2). For each cluster $\mathcal{C}_j$, selection is performed separately to obtain the corresponding selection set $\mathcal{S}_j$. From each $\mathcal{S}_j$, a separate linkage model $\mathcal{F}_j$ is learned (see Section 3.2.3). Finally, for each solution $\mathcal{P}_i$ in the population $\mathcal{P}$, the cluster $\mathcal{C}_j$ that it belongs to is determined. Variation is then performed on the solution $\mathcal{P}_i$ by recombining it with other solutions in the same cluster $\mathcal{C}_j$, using the linkage relations captured by the corresponding linkage model $\mathcal{F}_j$ (see Section 3.2.4). This transforms each solution in the population into an offspring solution. These offspring completely replace the population. The pseudo-code for the outline of MO-GOMEA is given in Figure 3.1.

```
MO-GOMEA //population size N, k clusters
 1  t ← 0; t^NIS ← 0
 2  for i ∈ {1, 2, ..., N} do
 3     P_i ← CreateRandomSolution()
 4     EvaluateFitness(P_i)
 5     A^t ← UpdateElitistArchive(P_i)
 6  while ¬TerminationCriteriaSatisfied do
 7     t ← t + 1
 8     {C_1, C_2, ..., C_k} ← ClusterPopulation(P)
 9     for j ∈ {1, 2, ..., k} do
10        S_j ← TournamentSelection(C_j)
11        F_j ← LearnLinkageModel(S_j)
12     for i ∈ {1, 2, ..., N} do
13        C_j ← DetermineCluster(P_i, {C_1, C_2, ..., C_k})
14        if ¬IsExtremeCluster(C_j) then
15           O_i ← MO-GenepoolOptimalMixing(P_i, C_j, F_j, A^t)
16        else
17           O_i ← SO-GenepoolOptimalMixing(P_i, C_j, F_j, A^t)
18     P ← O = {O_1, O_2, ..., O_N}
19     if f(A^t) ≠ f(A^{t−1}) then
20        t^NIS ← 0
21     else
22        t^NIS ← t^NIS + 1
```

Figure 3.1: Pseudo code for MO-GOMEA.

3.2.1. Elitist archive

We use a basic, but effective, implementation of the elitist archive. We denote the elitist archive in generation t by $\mathcal{A}^t$ in the decision-variable space and by $\boldsymbol{f}(\mathcal{A}^t)$ in the objective space. Every newly generated solution is checked to see if it can be added into the archive. If the new solution is dominated by any archive member, it will be discarded. If it is a new non-dominated solution, it will be added into the archive, and archive members that are dominated by this new solution will be

removed. In the case that there exists an archive member with the same objective values, the existing solution will be replaced by the new one if such replacement results in a diversity improvement for the archive in decision-variable space. To this end, we use a simple diversity metric for a solution: the Hamming distance to the nearest neighbor in the archive. Between the existing archive member and the new solution, the one having greater value for this metric will be chosen.

The elitist archive size is normally bounded by physical capacities and/or preferences of practitioners. Because the number of non-dominated solutions are numerous (or even infinite in, e.g., continuous domains), it can happen that the archive size is exceeded and some non-dominated solutions need to be removed to maintain the archive within its allowed capacity. The Adaptive Grid Discretization (AGD) [25] mechanism is implemented into MO-GOMEA to select which solutions should be kept during archive truncations. The AGD discretizes the objective space into equal hypercubes, and each hypercube is allowed to contain at most one solution at a time. Each time the AGD is triggered, the discretization level is computed based on the ranges of the non-dominated front formed by the current elitist archive such that the elitist archive is maintained around the size desired by users. However, because of the nature of our benchmarks and experiment settings (see Section 3.3), we set the elitist archive size to be as large as the Pareto-optimal front, i.e., the size of our elitist archive is unbounded in this thesis.

3.2.2. Clustering

We employ the balanced k-leader-means clustering as in MAMaLGaM [17] to cluster the population into k clusters of equal sizes where clusters may overlap. Note that clustering is performed in the objective space. First, a heuristic is used to select k leader solutions that are as well spread as possible. To do so, the first leader is the solution with maximum value in an arbitrary objective. The nearest-leader distances of all remaining solutions are computed as the distances to this first leader. The solution with the largest distance is chosen as the next leader. For every remaining solution, its nearest-leader distance is then updated if the distance to this new leader is smaller than the previous value. This process is repeated until k leaders are obtained. Second, k-means clustering is performed with the k leaders as the initial cluster means. Third, the distance from every solution to every final cluster mean is computed. Each cluster is then expanded to contain exactly c closest solutions, starting from each cluster mean. It was previously suggested to use $c = \frac{2}{k}|\boldsymbol{\mathcal{P}}|$, where $\boldsymbol{\mathcal{P}}$ are the solutions being clustered, resulting in overlap between neighboring clusters [17]. This reduces the probability that some solutions are not covered by any clusters. Moreover, it is beneficial to have equal-sized clusters so that a comparable amount of resources is used to handle each part of the Pareto-optimal front, thereby supporting that the whole Pareto-optimal front can be evenly approached.

Instead of clustering the selection set as in mohBOA [11] and MAMaLGaM [17], in MO-GOMEA the whole population is clustered. This is done because MO-GOMEA does not generate offspring solutions by sampling new solutions from the learned models as is the case in MOEDAs. Instead, the variation operator in MO-GOMEA is used to improve each solution in the population in a more local-search

like fashion (see Section 3.2.4). To do so, each solution in the population needs to be associated with a cluster.

Clustering helps to handle different parts of the Pareto-optimal front separately. This can be of major importance, especially for the extreme regions of the front where solutions maximize a single objective. This is because solutions from different extreme regions typically differ a lot, and a single search direction (as in the case that no clustering is employed) would not be sufficient nor efficient to approach all regions simultaneously. Moreover, in some problems the number of available solutions in the extreme regions can be much smaller than in the middle regions of the optimal front [22]. The additional use of separate single-objective optimizers to specifically obtain solutions in extreme regions of the front as used in MAMaLGaM-X$^+$ has been shown to be highly beneficial [17]. Although these separate optimizers can be tied in with the MOEA by running them generationally-synchronously parallel, putting the best solutions found by the separate optimizers into the elitist archive, and having the elitist archive participate in variation, such interaction between the populations of the single-objective and multi-objective optimizers is still very limited. Therefore, instead of using external single-objective optimizers, in MO-GOMEA we use only a single population. However, in every generation, for each objective, in MO-GOMEA, the cluster having the largest mean value in that objective is designated to perform only single-objective optimization in that objective during variation. If a single cluster happens to have the largest mean in more than one objective, we choose an objective arbitrarily. Moreover, while the selection procedure for learning linkage models in middle clusters is based on the Pareto-domination concept, the selection for each extreme cluster is based on the corresponding objective only. Figure 3.2 illustrates the clustering concept and the cluster-based operation of MO-GOMEA.

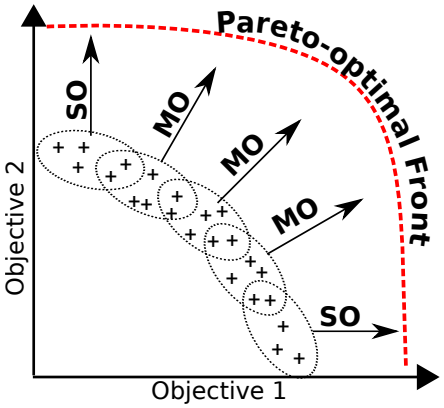

Figure 3.2: The cluster-based operation of MO-GOMEA. Different clusters approach different parts of the Pareto-optimal front. **MO**: Multi-objective optimization on the basis of Pareto dominance. **SO**: Single-objective optimization with respect to the corresponding objective.

3.2.3. Linkage learning

Linkage learning is performed for each cluster separately. Similar to the single-objective GOMEA [12], linkage learning is performed on a selection set that is obtained using tournament selection with tournament size 2 to have a beneficial bias in the model toward solutions that have better fitness values.

Let $L = \{1, 2, \ldots, l\}$ be the set of indices of all l decision variables. To capture the interactions between these decision variables, GOMEAs use a general linkage model termed the Family Of Subset (FOS). A FOS $\mathcal{F}$ consists of subsets of set L, i.e., $\mathcal{F} = \{\boldsymbol{F}^1, \boldsymbol{F}^2, \ldots, \boldsymbol{F}^{|\mathcal{F}|}\}$ where $\boldsymbol{F}^i \subseteq \{1, 2, \ldots, l\}$, $i \in \{1, 2, \ldots, |\mathcal{F}|\}$. A FOS $\mathcal{F}$ is thus a subset of the powerset of L, i.e., $\mathcal{F} \subseteq \mathcal{P}(L)$. Every subset $\boldsymbol{F}^i$ can be seen as a linkage set of problem variables that exhibit some degree of joint dependency and that should thus be copied jointly when performing variation.

Various GOMEA instances exist with different FOS structures [12, 26]. Here, MO-GOMEA employs the Linkage Tree (LT) structure, which is the most commonly used in literature. The LT contains all singleton subsets as its leaves, i.e., $\boldsymbol{F}^i = \{i\}, i \in \{1, 2, \ldots, l\}$, capturing decision variables as being fully independent. The LT also contains combinations of variables, organized in a tree-like fashion. A branch node of the LT is a bivariate or multivariate subset $\boldsymbol{F}^i$, which is created by combining two subsets $\boldsymbol{F}^j$ and $\boldsymbol{F}^k$ such that $\boldsymbol{F}^j \cap \boldsymbol{F}^k = \emptyset, |\boldsymbol{F}^j| < |\boldsymbol{F}^i|, |\boldsymbol{F}^k| < |\boldsymbol{F}^i|$ and $\boldsymbol{F}^j \cup \boldsymbol{F}^k = \boldsymbol{F}^i$. The root node, which contains all the decision variable indices, is the set L itself and is disregarded as it does not generate any new offspring solution when doing recombination. The LT without the root node has $2l - 2$ linkage sets. Details about how a LT model is learned are presented in Chapter 2.

3.2.4. Gene-pool optimal mixing

Classical EAs use blind crossover and mutation to create offspring solutions. EDAs sample the learned probability distribution to generate new solutions. GOMEAs perform an intensive mixing procedure aimed at efficiently exploiting the FOS linkage model to improve all population members, one by one; the resulting solutions are offspring solutions.

Aimed at covering the whole Pareto-optimal front by discerning and exploiting structure in different regions, the population is clustered in MO-GOMEA and for each cluster $\mathcal{C}_j$, a dedicated linkage model $\mathcal{F}_j$ is learned. Therefore, before improving a population member, we need to determine which cluster that solution belongs to. Although the clustering algorithm used here constructs equally-sized clusters, some solutions in $\mathcal{P}$ might actually not be covered by any cluster and some solutions may be covered by more than one cluster. Uncovered solutions are assigned to the cluster with the nearest mean. In case of multiple cluster assignments, ties are broken randomly. Every cluster thereby can be used to improve a more or less equal number of solutions.

Given cluster C_j that an existing (parent) solution $\boldsymbol{p}$ belongs to, and the corresponding FOS $\mathcal{F}_j$ of that cluster, MO-GOMEA changes $\boldsymbol{p}$ incrementally by the Gene-pool Optimal Mixing (GOM) procedure into an offspring solution as follows. First, the offspring solution $\boldsymbol{o}$ is created by cloning $\boldsymbol{p}$ and a backup $\boldsymbol{b}$ of $\boldsymbol{o}$ is created. We then traverse the linkage sets in FOS $\mathcal{F}_j$ in a random order. For every $\boldsymbol{F}^i \in \mathcal{F}_j$,

a donor solution $\boldsymbol{d}$ is randomly chosen from the same cluster C_j. The values of the problem variables whose indices are indicated by the linkage set $\boldsymbol{F}^i$ are copied from the donor $\boldsymbol{d}$ to the current solution $\boldsymbol{o}$. The objective values of this partially-altered solution $\boldsymbol{o}$ are evaluated and are compared with the backed-up state $\boldsymbol{b}$. If such mixing results in an improvement (i.e., the altered solution dominates the backed-up state $\boldsymbol{o} \succ \boldsymbol{b}$) or an equally good solution (i.e., the altered solution has the same objective values as the backed-up state $\boldsymbol{f}(\boldsymbol{o}) = \boldsymbol{f}(\boldsymbol{b})$) or a side step (i.e., the altered solution may not dominate the backed-up state but it is not dominated by any solutions in the elitist archive $\mathcal{A} \not\succ \boldsymbol{o}$), the changes are accepted and recorded as the new backup. Otherwise, the current solution is reverted to its backed-up state. When all linkage sets in the linkage tree are traversed, an offspring is then fully constructed.

It can happen that all the mixing steps of Gene-pool Optimal Mixing do not improve nor at least change the parent solution or that there exist significant plateaus in the problem causing solutions to be changed back and forth. In GOMEA this problem was tackled by using a procedure termed Forced Improvement [26]. Forced Improvement is essentially a second round of Optimal Mixing but the donor solution then is always the currently best found solution. In [26], Forced Improvement is triggered when a parent solution is not changed after going through Optimal Mixing or when the no-improvement stretch (NIS), i.e., the number of subsequent generations that the best solution did not change, exceeds the threshold $1 + \lfloor \log_{10}(N) \rfloor$. Here, we implement a multi-objective version of Forced Improvement through the following modifications. First, NIS is redefined as the number of consecutive generations that the non-dominated front (in the objective space) formed by non-dominated solutions in the elitist archive stayed the same. Second, a new donor solution is now selected randomly from the elitist archive for each linkage set. Third, because Forced Improvement aims to strictly improve the parent solution, a mixing step in the Forced Improvement phase is only accepted if it results in a direct domination (i.e., the altered solution dominates the backed-up state $\boldsymbol{o} \succ \boldsymbol{b}$) or a non-dominated-front improvement (i.e., a truly new non-dominated solution is found: $\mathcal{A}^t \not\succ \boldsymbol{o} \wedge \boldsymbol{f}(\boldsymbol{o}) \notin \boldsymbol{f}(\mathcal{A}^t)$). Similar to the single-objective version, rather than traversing the whole linkage tree, the Forced Improvement procedure is stopped as soon as the first change is accepted. If the parent solution is still unchanged after Forced Improvement, we simply replace it by a solution randomly chosen from the elitist archive. Pseudo-code is given in Figure 3.3.

Gene-pool Optimal Mixing (GOM) in MO-GOMEA can be seen as a direct extension of Gene-pool Optimal Mixing in GOMEA by replacing fitness improvement checking with Pareto-domination checking in every mixing step, and using multiple solutions from the elitist archive instead of a single best solution for the Forced Improvement phase. Multi-objective Gene-pool Optimal Mixing is employed for improving solutions that are determined as belonging to the middle-region clusters. However, as discussed earlier in Section 3.2.2, having a mechanism that puts extra pressure on the individual objectives can be highly beneficial because the Pareto-domination improvement may not give enough pressure to find the extreme solutions (the solutions that maximize a single objective). Therefore, to improve solutions that belong to extreme-region clusters, we employ single-objective Gene-

```
MultiObjective-GenepoolOptimalMixing(p, C, F, A^t)
 1 b ← o ← p; f(b) ← f(o) ← f(p); changed ← false
 2 for i ∈ {1, 2, ..., |F|} do
 3   d ← PickRandomDonorFromCluster(C)
 4   o_{F^i} ← d_{F^i}
 5   if o_{F^i} ≠ b_{F^i} then
 6     f(o) ← EvaluateFitness(o); updated ← false; improved ← false
 7     if [A^t ⊁ o and f(o) ∉ f(A^t)] or
 8       [f(o) ∈ f(A^t) and IsDiversityIncreasedByAdding(A^t, o)] then
 9         updated ← true; improved ← true
10     else if o ≻ b or f(o) = f(b) or A^t ⊁ o then
11         improved ← true
12     if updated then
13         A^t ← UpdateElitistArchive(o)
14     if improved then
15         b_{F^i} ← o_{F^i}; f(b) ← f(o); changed ← true
16     else
17         o_{F^i} ← b_{F^i}; f(o) ← f(b)
18 if ¬changed or t^NIS > 1 + ⌊log10(N)⌋ then
19   changed ← false
20   for i ∈ {1, 2, ..., |F|} do
21     d ← PickRandomDonorFromElitistArchive(A^t)
22     o_{F^i} ← d_{F^i}
23     if o_{F^i} ≠ b_{F^i} then
24       f(o) ← EvaluateFitness(o); updated ← false; improved ← false
25       if A^t ⊁ o and f(o) ∉ f(A^t) then
26           updated ← true; improved ← true
27       else if f(o) ∈ f(A^t) and IsDiversityIncreasedByAdding(A^t, o) then
28           updated ← true
29       if o ≻ b then
30           improved ← true
31       if updated then
32           A^t ← UpdateElitistArchive(o)
33       if improved then
34           b_{F^i} ← o_{F^i}; f(b) ← f(o); changed ← true
35       else
36           o_{F^i} ← b_{F^i}; f(o) ← f(b)
37     if changed then breakfor
38 if ¬changed then
39   d ← Random(A^t); o ← d; f(o) ← f(d)
```

Figure 3.3: Pseudo code for multi-objective Gene-pool Optimal Mixing

pool Optimal Mixing and single-objective Forced Improvement of the as reported in [26]. In other words, if a parent solution $\boldsymbol{p}$ belongs to the extreme cluster that is associated with optimizing single objective f_i, $\boldsymbol{p}$ will be improved by Gene-pool Optimal Mixing in the single-objective manner, and the solution $\boldsymbol{x}_{f_i}^{best}$ that stores the current maximum observed value for f_i will be considered as the single donor solution in Forced Improvement. Note that, however, the NIS concept is still defined in

the overall multi-objective optimization context with regard to the non-dominated front formed by the elitist archive members. Figure 3.2 illustrates the cluster-based operation of MO-GOMEA, in which middle-region clusters employ multi-objective GOM with improvement checks on the basis of Pareto dominance while extreme-region clusters employ single-objective GOM with improvement checks on the basis of their corresponding objectives.

3.3. Benchmark problems

3.3.1. Problems

Because we will use benchmark problems throughout subsequent sections to test and decide on various design choices for MO-GOMEA, we present all problems that we consider in this chapter in this section first. We also describe how the Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$ of each problem instance can be obtained, which is used to evaluate the performance of MOEAs.

Scalable benchmark problems

Zeromax-Onemax

Zeromax-Onemax [11] has two objectives that are defined over a binary string $\boldsymbol{x}$ of l bits as follows:

$$f_{\text{Onemax}}(\boldsymbol{x}) = \sum_{i=1}^{l} x_i; \quad f_{\text{Zeromax}}(\boldsymbol{x}) = l - f_{\text{Onemax}}(\boldsymbol{x}) \tag{3.1}$$

Objective f_{Onemax} counts the number of bits set to 1 while f_{Zeromax} counts the number of bits set to 0, making the objectives conflicting. Every candidate solution is a Pareto-optimal solution; any increase in the number of bits set to 1 will be a decrease in the number of bits set to 0, and vice versa. The Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$ of Zeromax-Onemax consists of $l+1$ points $\boldsymbol{\mathcal{P}}_F = \{(i, l-i) \mid i \in \{0, 1, \ldots l\}\}$, on a straight line. A point on $\boldsymbol{\mathcal{P}_F}$ can correspond to many different solutions, with the exception of the two extreme points that correspond to exactly a string of all 1s (maximizing Onemax) and a string of all 0s (maximizing Zeromax). It can be seen that the niches of the solutions having objective values in the middle regions of the front $\boldsymbol{\mathcal{P}_F}$ are exponentially larger than the niches in the extreme regions of $\boldsymbol{\mathcal{P}_F}$.

Trap-Inverse Trap

Trap-Inverse Trap [11] also has two objectives. A Trap-k is the well-known mutually exclusive, additively decomposable composition of the order-k deceptive subfunctions. In this thesis, we consider Trap-5:

$$f_{\text{Trap-5}}(\boldsymbol{x}) = \sum_{i=1}^{l/5} f^{\text{sub}}_{\text{Trap-5}} \left(\sum_{j=5i-4}^{5i} x_j \right) \tag{3.2}$$

where

$$f^{\text{sub}}_{\text{Trap-5}}(u) = \begin{cases} 5 & \text{if } u = 5 \\ 4 - u & \text{if } u < 5 \end{cases} \tag{3.3}$$

Inverse Trap-5 comprises the same partitions as Trap-5 but each partition is evaluated "inversely":

$$f^{\text{sub}}_{\text{Inverse Trap-5}}(u) = \begin{cases} 5 & \text{if } u = 0 \\ u - 1 & \text{if } u > 0 \end{cases} \tag{3.4}$$

A candidate solution for which each partition has one objective subfunction that evaluates to 5 is Pareto-optimal. The Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$ consists of $l/5 + 1$ points $\boldsymbol{\mathcal{P}}_F = \{(5i + 4(l/5 - i), 5(l/5 - i) + 4i) \mid i \in \{0, 1, \ldots l/5\}\}$, on a straight line. It is well-known known that in order to solve additively decomposable trap functions, it is crucial to preserve the linkage relations between problem variables during variation. Therefore, this benchmark is often used to demonstrate the importance of linkage learning.

Leading Ones Trailing Zeros (LOTZ)
The LOTZ problem consists of two objectives: maximizing the number of consecutive bits set to 1 at the beginning and maximizing the number of consecutive bits set to zero at the end.

$$f_{\text{LO}}(\boldsymbol{x}) = \sum_{i=1}^{l} \prod_{j=1}^{i} x_j; \quad f_{\text{TZ}}(\boldsymbol{x}) = \sum_{i=1}^{l} \prod_{j=i}^{l} (1 - x_j) \tag{3.5}$$

A candidate solution that consists of a substring of all 1s followed by a substring of all 0s, i.e., having the form of $11 \ldots 100 \ldots 0$, is Pareto-optimal. The two extreme solutions are all 1s (maximizing Leading Ones) and all 0s (maximizing Trailing Zeros). The Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$ of LOTZ consists of $l + 1$ points $\boldsymbol{\mathcal{P}}^i_F = \{(i, l - i) \mid i \in \{0, 1, \ldots l\}\}$, on a straight line. However, different from the Zeromax-Onemax problem, any point on the Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$ of LOTZ corresponds to exactly one Pareto-optimal solution.

Multi-objective MAXCUT
Definition
Weighted MAXCUT is defined over a weighted undirected graph $G = (V, E)$, where $V = (v_1, v_2, \ldots, v_l)$ is the set of l vertices, and E is the set of edges (v_i, v_j) with corresponding weights w_{ij}'s. A maximum cut is a partition of l vertices into two disjoint subsets A and $B = V \setminus A$ such that the combined weight of all edges (v_i, v_j) having $v_i \in A$ and $v_j \in B$ is maximized. A cut can be encoded as a binary string $\boldsymbol{x}$ of l bits, in which each bit x_i corresponds to a vertex, and all 0-valued bits indicate vertices of set A while all 1-valued bits indicate vertices of set B. The objective function of the weighted MAXCUT problem is defined as follows

$$f_{\text{MAXCUT}}(\boldsymbol{x}) = \sum_{(v_i, v_j) \in E} \begin{cases} w_{ij} & \text{if } x_i \neq x_j \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

We construct a multi-objective version of the weighted MAXCUT problem by optimizing a different MAXCUT instance in each objective. The instances have identical vertices but different edge weights.

Problem Instances

We experiment with four multi-objective MAXCUT problem instances of size $l = 12, 25, 50, 100$. Each MAXCUT instance is a fully connected graph having $\frac{1}{2}l(l-1)$ edges. The edge weights are set by following the approach described in [26]. For the 12-vertex and 25-vertex problem instances, we can easily obtain the true Pareto-optimal fronts $\boldsymbol{\mathcal{P}_F}$ by enumeration. For the 50-vertex and 100-vertex problem instances, because $\boldsymbol{\mathcal{P}_F}$'s are unknown, we can only obtain reference fronts by consulting the results of [26] as follows. We use the maximum population sizes of GOMEA, that reliably solve the single-objective MAXCUT, reported in [26] to set as the cluster size for MO-GOMEA. We then run five different instances of MO-GOMEA with different number of clusters $k \in \{1, 3, 5, 7, 10\}$. Each MO-GOMEA instance is run 100 times, each time with a budget of 20 million function evaluations. We take the non-dominated front of all results in all these runs over all five instances of MO-GOMEA to be $\boldsymbol{\mathcal{P}_F}$ in this case. A degree of reliability can be taken from the fact that the optimal extreme points were always found to be in the final reference fronts $\boldsymbol{\mathcal{P}_F}$ so constructed.

Multi-objective Knapsack

Problem Definition

The multi-objective knapsack problem involves l items and m knapsacks. Each knapsack k has a specific capacity c_k. Each item i has a weight $w_{i,k}$ and a profit $p_{i,k}$ corresponding to each knapsack k. A solution of the knapsack problem can be encoded as a binary string $\boldsymbol{x} = (x_1, x_2, \ldots, x_l) \in \{0,1\}^l$, where each bit x_i corresponds to an item i, and $x_i = 1$ indicates that item i is selected. Selecting an item i in the multi-objective knapsack context means the item i is placed in every knapsack. A feasible solution is a selection of items such that the total weight does not exceed the capacity of any knapsack. The objectives are to maximize the profits of all knapsacks at the same time as follows.

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{maximize}} && (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})) \\ &\text{where} && f_k(\boldsymbol{x}) = \sum_{i=1}^{l} p_{i,k} x_i \quad k = 1, \ldots, m \\ &\text{subject to} && \sum_{i=1}^{l} w_{i,k} x_i \leq c_k \quad k = 1, \ldots, m \end{aligned} \tag{3.7}$$

Constraint Handling

If a solution violates the capacity constraint of any knapsack, we use the repair mechanism proposed in [27] to iteratively remove items until all constraints are satisfied. The item removal order follows the principle that the items with the lowest maximum profit/weight ratio should be discarded first.The maximum profit/weight ratio r_i of an item i as follows (as in [27]).

$$r_i = \max_{k=1}^{m} \left\{ \frac{p_{i,k}}{w_{i,k}} \right\} \tag{3.8}$$

This repair mechanism tries to satisfy the capacity constraints while diminishing the overall profit as little as possible by considering all knapsacks when removing items. This multi-objective repair works well for approaching middle regions of the Pareto-optimal front, where trade-off solutions that balance all objectives are located. It can be intuitively noticed that Pareto-optimal solutions in the extreme regions, which maximize the profit of a specific knapsack, can be approached more efficiently if they are targeted by employing a repair mechanism dedicated to that knapsack. A repair function dedicated to a knapsack k also iteratively removes items until all constraints are satisfied but the item removal order biases toward knapsack k, such that items with lowest profit/weight ratio $r_{i,k} = p_{i,k}/w_{i,k}$ should be discarded first. The question is which repair mechanism should be used for a specific infeasible solution. The answer is straightforward for MO-GOMEA: middle-region clusters should employ the multi-objective repair mechanism while each extreme-region cluster should employ the single-objective repair mechanism that corresponds to its target objective.

Problem Instances
We use the bi-objective knapsack problem instances of 100, 250, 500, 750 items proposed by [27]. The true Pareto-optimal fronts $\boldsymbol{\mathcal{P}_F}$'s of the 100-, 250-, and 500-item problem instances have been reported in [27]. For the 750-item problem instance, $\boldsymbol{\mathcal{P}_F}$ is unknown and we replace it by a reference set, which is created by combining all non-dominated fronts obtained by all optimizers in all runs.

3.4. Performance indicator

To compare the performance of different MO-GOMEA variants, we evaluate the quality of the Pareto-optimal front approximation obtained by each variant. In our case, this approximation equals the elitist archive. For our settings, we employ the inverse generational distance performance (IGD) indicator $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\rightarrow\mathcal{S}}$ [6]:

$$\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\rightarrow\mathcal{S}}(\mathcal{S}) = \frac{1}{|\boldsymbol{\mathcal{P}_F}|} \sum_{\boldsymbol{f}^0 \in \boldsymbol{\mathcal{P}_F}} \min_{\boldsymbol{x}\in\mathcal{S}} \{d(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}^0)\} \tag{3.9}$$

where $\boldsymbol{\mathcal{P}_F}$ is the true Pareto-optimal front (or a reference front), $\mathcal{S}$ is the final approximation set (i.e., the outcome of that optimizer), and $d(\cdot,\cdot)$ computes the Euclidean distance. The $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\rightarrow\mathcal{S}}$ performance indicator is the average distance in the objective space from a point in $\boldsymbol{\mathcal{P}_F}$ to its nearest point in $\mathcal{S}$. A smaller value of $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\rightarrow\mathcal{S}}$ indicates a better performance result, and the value 0 is achieved if and only if the front formed by the approximation set equals the Pareto-optimal front. Note that calculating the average distance in "reversed order" yields a different indicator $\boldsymbol{D}_{\mathcal{S}\rightarrow\boldsymbol{\mathcal{P}_F}}$, also known as the generational distance (GD). However, obtaining a low $\boldsymbol{D}_{\mathcal{S}\rightarrow\boldsymbol{\mathcal{P}_F}}$ value does not mean that a good approximation set has been found because the GD indicator only considers the proximity of a front to the Pareto-optimal front. An approximation set $\mathcal{S}(|\mathcal{S}| = 1)$ that contains only a single Pareto-optimal solution in $\boldsymbol{\mathcal{P}_F}$ can yield the best GD value 0. On the other hand, the IGD $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\rightarrow\mathcal{S}}$ indicator

takes into account both proximity (i.e., how close $\mathcal{S}$ is to the front $\boldsymbol{\mathcal{P}_F}$) and diversity (i.e., how well-spread $\mathcal{S}$ is along the front $\boldsymbol{\mathcal{P}_F}$) of the approximation set $\mathcal{S}$.

3.5. Benchmarking MO-GOMEA

3.5.1. Experiment setup

We compare the performance of MO-GOMEA with a commonly used instance of NSGA-II [7] (2-point crossover with probability 0.9 and bit-flipping mutation with probability $1/l$). We also consider the best results obtained by the EDAs (MO)UMDA and mohBOA as reported in [11]. We want to assess scalability, i.e., how the population size requirements and the required number of evaluations grow as the problem size increases. For the problems Zeromax-Onemax, Trap-Inverse Trap, and LOTZ, the Pareto-optimal fronts are known, so to this end, we perform bisection [28]. Bisection is a binary-search inspired procedure that aims to find the minimally required population size to solve a problem instance reliably. Here, we define reliable as achieving $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\to\mathcal{S}} = 0$ (i.e., the entire Pareto-optimal front is found) in all 100 out of 100 independent runs. We perform 10 independent bisection searches for every problem size $l \in \{25, 50, 100, 200, 400\}$.

For MO-MAXCUT, because we do not know the Pareto-optimal front for certain, it is difficult to perform bisection. Moreover, MO-MAXCUT is a (NP-)hard problem, for which polynomial scalability may not be expected, so we wish to observe how good of an approximation can be achieved. Therefore, we set the population sizes for MO-GOMEA and NSGA-II by consulting the results in [26], which solved SO-MAXCUT reliably. For each problem size, we use the average required population size of LTGA as the cluster size in MO-GOMEA and as the base population size in NSGA-II. We run MO-GOMEA with the number of clusters $k \in \{1, 3, 5, 7, 10\}$. Similarly, we run NSGA-II with the base population size scaled by $1, 2, 4, 8,$ and 16 times. For every MAXCUT problem instance $l \in \{12, 25, 50, 100\}$, we perform 100 independent runs. We determine performance on the basis of the average convergence graph of the $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\to\mathcal{S}}$ indicator. The budgets of function evaluations are also set by consulting [26]. For every problem size, we set the maximum number of evaluations to be 10 times the average number of evaluations that LTGA required to solve SO-MAXCUT (we multiply by 10 because we have an MO-GOMEA instance with at most $k = 10$ clusters). Note that we always consider one function evaluation to include evaluating both objectives.

The knapsack problem is a constrained problem. Different constraint-handling methods, which are problem-specific, can have different effects on the performance of optimization algorithms. Because we would like to focus on benchmarking the scalability of general-purpose MOEAs in this section, we prefer unconstrained problems. Therefore, we do not perform experiments on the knapsack problem here but save it for Sections 3.7 and 3.8.

To support our conclusion from experimental results, we perform the Mann-Whitney-Wilcoxon statistical hypothesis test for equality of medians with $p < 0.05$ to see whether the final result obtained by one optimizer is statistically different from that of another optimizer.

3.5.2. Results & discussions

Zeromax-Onemax

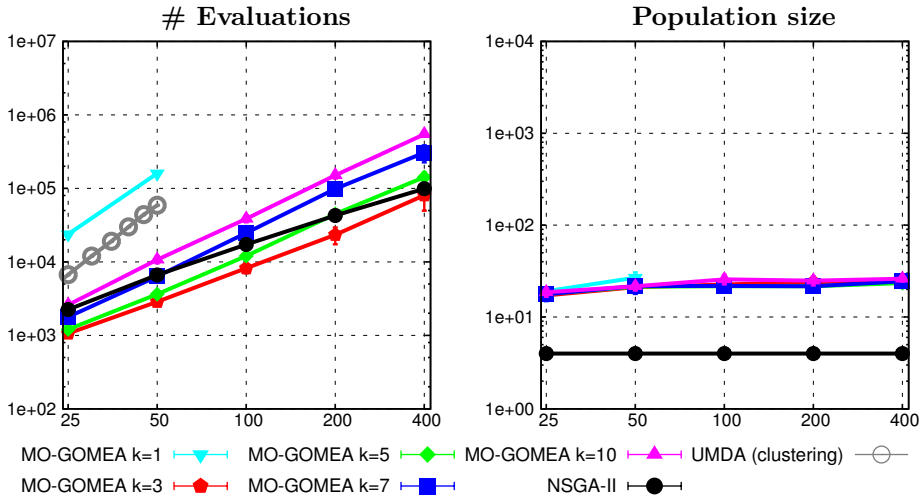

Figure 3.4: UMDA, NSGA-II, and MO-GOMEAs solving Zeromax-Onemax. Horizontal axis: problem size.

Figure 3.4 shows that MO-GOMEA with clustering (i.e., $k > 1$) outperforms UMDA with clustering in terms of number of evaluations as well as scalability. Note that, in [11], UMDA with clustering did not have an elitist archive and the required population size was not reported. It can be inferred however, that in order to cover all points of the Pareto-optimal front, the population size of UMDA without elitist archive must be at least as large as the Pareto-optimal front. Our MO-GOMEA instances require a smaller population size that grows quite slowly as the problem size increases. It can furthermore be observed that for solving this problem we do not really need many clusters in MO-GOMEA because all problem variables are independent from each other in all regions of the search space. That explains why MO-GOMEA with $k = 3$ has the best performance, and as we increase the number of clusters ($k = 5, 7, 10$), the number of evaluations also increases accordingly, but the scalability is relatively the same. However, MO-GOMEA without clustering ($k = 1$), and thus without internal SO optimizers, solves the problem much less efficiently. This is because the two extreme solutions (maximizing Zeromax vs. maximizing Onemax) are in very small niches that contain only one solution. Without clustering to distribute enough resources and without SO optimizers to approach these extreme regions efficiently, it is much harder to cover the entire Pareto-optimal front. The differences in the required numbers of fitness evaluations of all MO-GOMEA variants ($k = 1, 3, 5, 7, 10$) solving each Zeromax-Onemax instance ($l = 25, 50, 100, 200, 400$) are found to be statistically significant. However, there is no significant difference between the minimally-required population sizes of MO-GOMEAs $k = 3, 5, 7$, and

10. In [11], an NSGA-II version without an elitist archive was found to hardly be able to solve the problem up to 50 problem variables. Here, being equipped with an elitist archive, NSGA-II can solve all Zeromax-Onemax instances with the smallest possible population size for NSGA-II of 4 individuals and with a performance much better than UMDA. This confirms the necessity of elitist archiving for scalable MO optimization.

NSGA-II does not have clustering nor SO optimizers. MO-GOMEA with $k = 3$ requires (statistically significantly) fewer fitness evaluations than NSGA-II in solving instances $l = 25, 50, 100$, and 200. However, on the problem instance $l = 400$, there is no statistically significant difference between the performance of MO-GOMEA $k = 3$ and NSGA-II. The fact that NSGA-II can solve all Zeromax-Onemax of various problem sizes with such a small population size (and with better scalability than MO-GOMEA) can be explained by its use of bit-wise mutation, which is not by default employed in MO-GOMEA. To validate this, we equip MO-GOMEA with a simple mutation operator: the mixing step at singleton linkage sets is replaced by randomly assigning a 0 or a 1 value (i.e., not genepool-based). The performance of MO-GOMEAs with this mutation operator is shown in Figure 3.5. The differences in the required numbers of evaluations of all optimizers are found to be statistically significant (except between MO-GOMEA $k = 3$ and $k = 5$ on the instance $l = 100$). With mutation, indeed the MO-GOMEAs now have a scalability relatively similar to NSGA-II. Moreover, the MO-GOMEAs need a fewer number of evaluations. The fact that MO-GOMEA with mutation but without clustering ($k = 1$, and thus without SO optimizers) still has worse performance indicates that clustering and internal SO optimizers are important features of MO-GOMEA. It is however also a result of the fact that mutation here is key to finding the extreme points. This however requires time in terms of generations and all higher-order mixing operations attempted by MO-GOMEA that follow from the use of the full linkage tree in the meantime are not useful.

Trap-Inverse Trap

Figure 3.6 clearly demonstrates that MO-GOMEA outperforms both mohBOA and NSGA-II. Different from the Zeromax-Onemax benchmark of all-independent problem variables, Trap - Inverse Trap requires linkage learning to detect and preserve linkage relations between problem variables during recombination processes. Linkage learning is not employed in NSGA-II and even though it uses a recombination operator (two-point crossover) that is linkage friendly because we have encoded the trap functions tightly, it is still highly inefficient in finding the optimum. The strengths of the linkage model and optimal mixing procedure of GOMEA, which are known to contribute to its superior performance in solving the SO version of Trap-5 [12], is successfully extended to the MO realm. Given a properly learned linkage model, the GOM procedure helps MO-GOMEA achieve successful mixing events which respect the dependencies between problem variables. The intensive local-search-inspired characteristic of the GOM procedure moreover results in MO-GOMEA requiring only a small and slow-growing population size, similarly to the SO case [12]. The differences in the required numbers of evaluations of all MO-GOMEA variants ($k = 1, 3, 5, 7, 10$) and NSGA-II solving each Trap-Inverse Trap

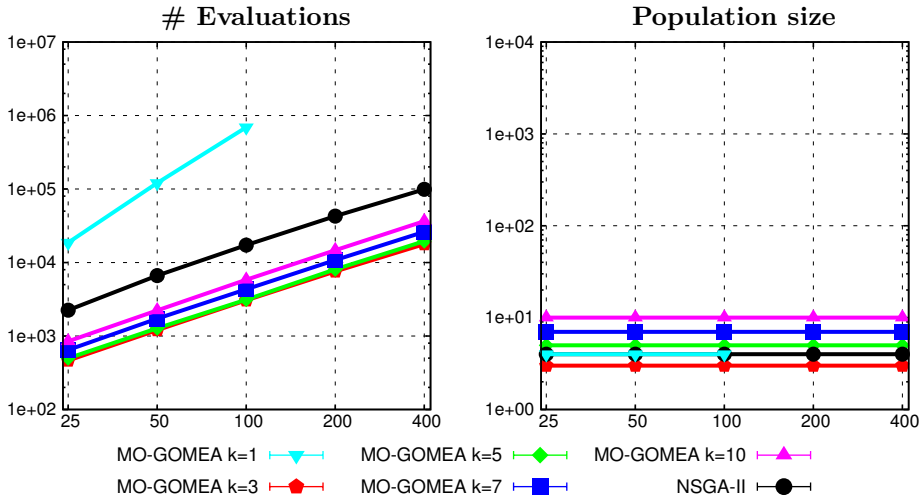

Figure 3.5: MO-GOMEAs with mutation solving Zeromax-Onemax. Horizontal axis: problem size.

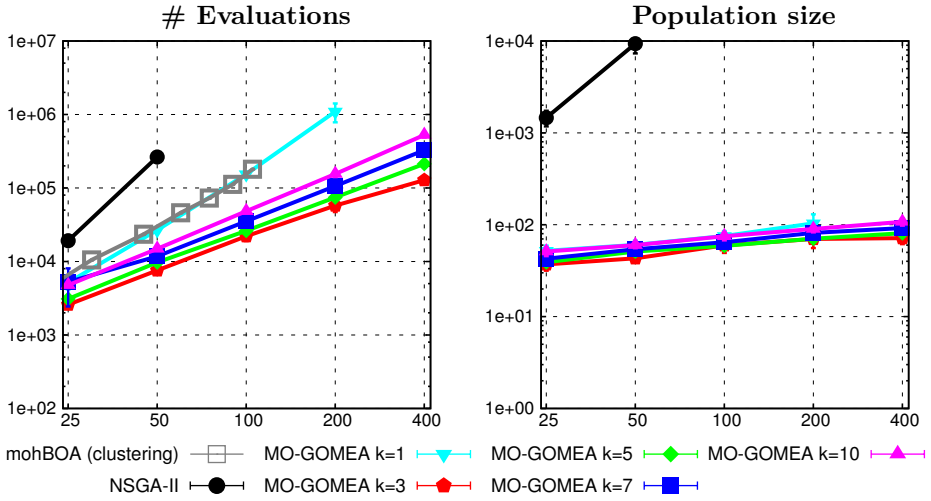

Figure 3.6: mohBOA, NSGA-II, and MO-GOMEAs solving Trap-Inverse Trap. Horizontal axis: problem size.

instance ($l = 25, 50, 100, 200, 400$) are found to be statistically significant (except between MO-GOMEAs $k = 1, 7$, and 10 on the instance $l = 25$). The minimally-required population sizes of all MO-GOMEA variants solving each Trap-Inverse Trap instance, however, are not statistically significantly different from each other.

The best results are again obtained with MO-GOMEA with three clusters. Us-

ing more clusters again leads to similar scalability but requires larger numbers of evaluations. This is again because the Pareto-optimal front of Trap-Inverse Trap has the same linkage structure (i.e., concatenated groups of five inter-dependent problem variables) in both objectives, resulting in redundant behavior when adding more clusters. However, MO-GOMEA with $k = 1$ clearly performs worse (but still better than NSGA-II) because the extreme regions of the front have again very small niches (similar to Zeromax - Onemax). Figure 3.6 also shows the performance of the MOEDA mohBOA with restricted tournament replacement and clustering, which had the best performance reported in [11]. Here, we obtain similar behavior for $k = 1$ but we obtain clearly better results for $k > 1$. Because in [11] mohBOA did not have an elitist archive, it is difficult to directly compare MO-GOMEA and mohBOA, which we did not re-implement. However, it has been reported in [12] that, in SO optimization for Trap-5, GOMEA had better performance than a typical EDA. Even if we assume that the probabilistic model building and sampling in mohBOA are as effective as the LT building and the GOM procedure in MO-GOMEA, the elitist archive and internal SO optimizers cause MO-GOMEA to have better scalability.

Lead Ones Trailing Zeros (LOTZ)

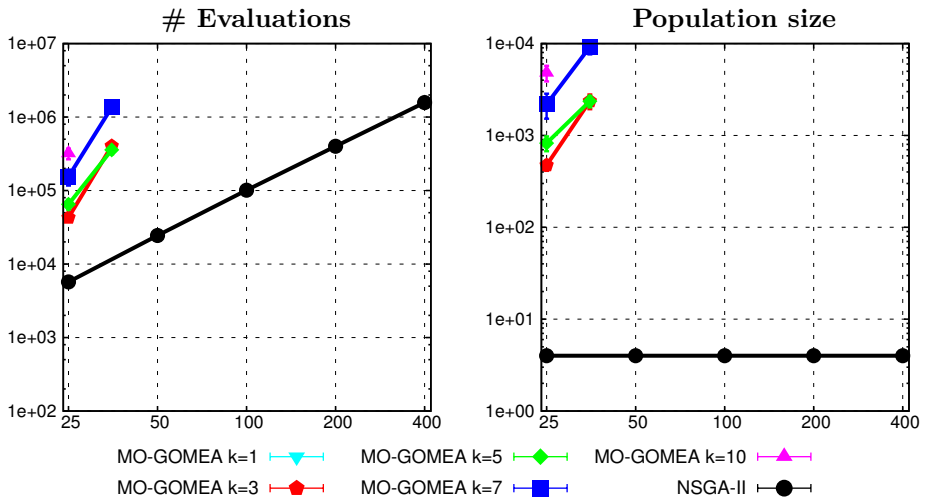

Figure 3.7: NSGA-II, and MO-GOMEAs solving LOTZ. Horizontal axis: problem size.

Figure 3.7 shows that LOTZ is a challenge for MO-GOMEA. The reason is that we aim to cover the entire Pareto-optimal front reliably while LOTZ has a peculiar linkage structure. Only substrings of consecutive 1 bits at the beginning (Leading Ones) and the substring of consecutive 0 bits at the end (Trailing Zeros) contribute to the objective values. This means that all the mixing events of 0s and 1s in the middle are useless and can be considered as *noise*, affecting the effectiveness of linkage learning in MO-GOMEA. Moreover, the LT model fails to capture efficiently

the type of operations needed to solve LOTZ. What further makes LOTZ a complicated problem is that selection pressure based on Pareto dominance makes it very difficult to obtain the extreme solutions of LOTZ (i.e. a string of all 1s and a string of all 0s). For example, a string $\boldsymbol{x}$ beginning with a 0 will be dominated by any strings beginning with a 1 and ending with a 0. This string $\boldsymbol{x}$ will therefore soon be deleted from of the population. The GOM of GOMEA cannot improve $\boldsymbol{x}$ while still keeping the leading 0 bit because GOM will prefer any dominating solutions $\boldsymbol{x}' \succ \boldsymbol{x}$ with a leading 1 bit. Even the SO version of GOM in the extreme clusters cannot efficiently preserve this leading 0 bit because the leading 0 bit is only useful when it is combined with the substring of trailing 0s to create the extreme solution of all 0s. Figure 3.7 shows however that NSGA-II has little difficulty in solving LOTZ, requiring a population size of only 4. This is due to the mutation operator of NSGA-II. It can be inferred from Figure 3.7 that, e.g., in order to solve LOTZ of 400 bits, NSGA-II needs to be run over than 300,000 generations, waiting for mutation to flip the right bit at the right time to obtain a Pareto-optimal solution. It should further be noted that the two-point crossover operator of NSGA-II has a favorable search bias here because it can preserve and recombine very large substrings of all 1s or all 0s at the beginning and end of a solution. Figure 3.8 validates this by showing that NSGA-II with uniform crossover performs (statistically significantly) worse.

LOTZ clearly poses additional challenges, which for MO-GOMEA include finding more appropriate linkage models and filtering useless subsets. However, LOTZ can still be solved fairly efficiently using a simple local search (LS) operator. Following the typical design of genetic local search, we apply LS at the end of every generation on each offspring solution by traversing the variables of a solution in a random order and flipping every variable, followed by a Pareto-improvement check. If a flip does not result in a solution dominated by the previous state nor dominated by the beginning state, it is accepted. Otherwise, it is reverted to the previous state. Figure 3.8 shows the performance of MO-GOMEA and NSGA-II with this LS. LS greatly improves the performance of MO-GOMEA, but not that of NSGA-II in terms of number of evaluations. These results underline just how beneficial it can be to use a proper problem-specific LS operator to bring back diversity that might have gotten lost.

Multi-objective MAXCUT

Although Zeromax-Onemax, Trap-Inverse Trap and LOTZ are interesting benchmark problems with known Pareto-optimal fronts, they are fairly artificial in the sense that the Pareto-optimal fronts always have the shape of a straight line. We therefore also perform experiments on the MO-MAXCUT problems where the Pareto-optimal front can be shaped very differently. The convergence graphs in Figure 3.9 show that for $l = 12$, only NSGA-II 16×, the one with the largest population size, outperforms MO-GOMEA. As the problem size increases however, the picture starts to change. For $l = 25$, only NSGA-II 16× has a convergence result as good as MO-GOMEA with $k = 5, 7$, but it is still (statistically significantly) outperformed by MO-GOMEA with $k = 10$. For larger problem sizes, $l = 50, 100$, MO-GOMEAs with clustering ($k > 1$) distinctly outperform all NSGA-II instances, and the differences are found to be statistically significant. The facts that MO-GOMEA with

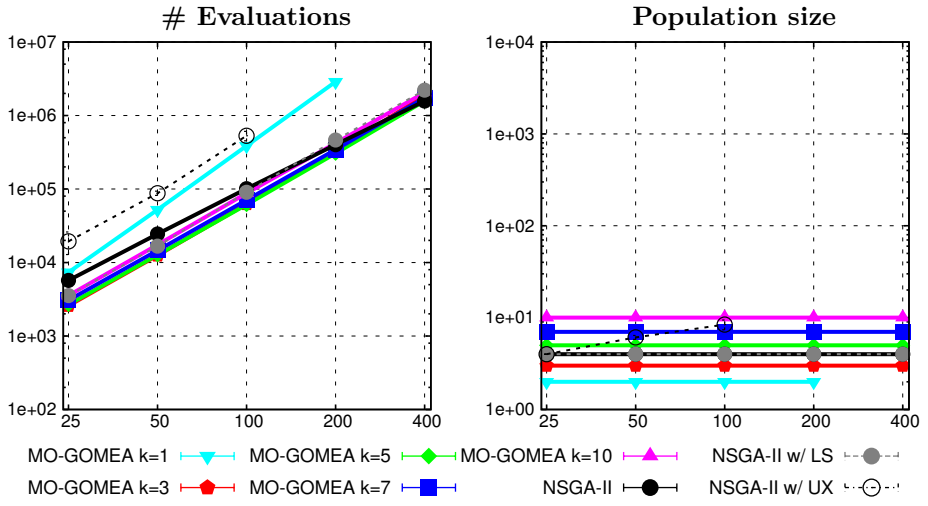

Figure 3.8: MO-GOMEAs with local search and NSGA-IIs solving LOTZ. Horizontal axis: problem size.

clustering exhibits better convergence for problem sizes $l > 25$ and that the bigger the problem, the wider the performance gap between the various MO-GOMEAs $k > 1$ and NSGA-IIs becomes, indicate the intrinsic superior scalability of MO-GOMEA over NSGA-II. MO-GOMEA without clustering ($k = 1$) has performance results relatively the same as those of NSGA-IIs, which do not have clustering either. This emphasizes again the importance of clustering to handle different parts of the Pareto-optimal front separately. Also, although we did not perform bisection to find the minimally required population size for each instance (because the true Pareto-optimal front is unknown), it can be inferred, to a certain degree, from MAXCUT $l = 12, 25, 50$ that the more clusters MO-GOMEA has the better the result is although it may take more time to obtain this result. This is because, different from the other benchmark problems, the structures of the objectives may now differ, causing different structures to be required to be exploited along the front, which is supported by clustering.

Discussions

The experimental results on the scalable benchmark problems and MAXCUT problem presented in this section demonstrate that MO-GOMEA is a robust multi-objective optimization algorithm with superior scalability compared to the state-of-the-art MOEAs NSGA-II. However, similar to other MOEAs, the robustness of MO-GOMEA is maximally attained only if its parameters (i.e., the population size N and the number of clusters k) are optimally set. For scalable benchmark problems, we can perform bisection to find the minimally required population size. For a small MAXCUT problem instance, we have a reliable reference for setting the population size. But for many optimization problems in practice, we do not have a

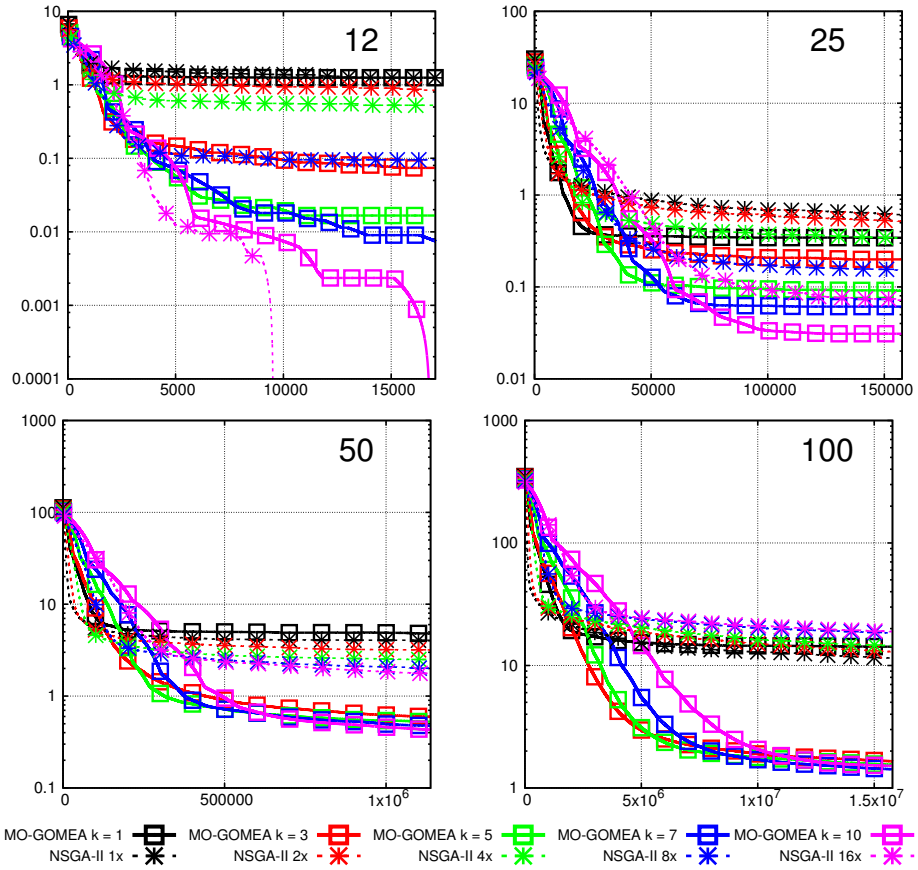

Figure 3.9: Average performance of instances of MO-GOMEA and instances of NSGA-II on multi-objective MAXCUT problems of different problem sizes. Horizontal axis: number of evaluations (both objectives per evaluation). Vertical axis: $\boldsymbol{D}_{\boldsymbol{\mathcal{P}}_{\boldsymbol{F}} \to \mathcal{S}}$.

solid guideline for setting this important parameter because the optimal population size depends on the problem instance at hand, the employed optimization algorithm itself, and the time available. This is particularly true for the DNEP problem in this thesis, for which there exists no result on optimal population sizes of MOEAs solving DNEP. Besides, even though the tournament selection operator for linkage learning with tournament size 2 appears to be a good choice in the experiments here, it can be argued that this is also a parameter that practitioners might need to consider in general. Practitioners often need to experiment with all these different parameter values in a trial-and-error manner, which is far from efficient. Therefore, in the following section, aiming to make MO-GOMEA an easy-to-use optimizer, we propose an approach to remove all parameter setting requirements that might discourage practitioners from employing MO-GOMEA.

3.6. Removing the requirement of parameter settings in MO-GOMEA

In this section, we present how we remove the requirement of parameter settings in MO-GOMEA. First, we perform experiments to show that the selection phase for linkage learning can be omitted. We then propose different potential parameter-less schemes and perform experiments to verify their performance. Based on the experimental results, we pick the scheme that has the most similar performance compared to MO-GOMEA where the population size is set to its ideal value for each problem anew. We employ two scalable benchmark problems, namely Zeromax-Onemax and Trap-Inverse Trap. The true Pareto-optimal fronts of these two problems can be deterministically calculated, which is convenient for comparing performance of different optimizers. We do not employ the LOTZ problem here for benchmarking parameter-less MO-GOMEAs because, as demonstrated in Section 3.5, it is much more efficient to solve LOTZ by mutation or local search, which does not exist in the standard version of MO-GOMEA. We will return to LOTZ in Section 3.7 when we discuss the influence of mutation operators.

3.6.1. Eliminating the selection operator

The original implementation of MO-GOMEA performed linkage learning on selection sets obtained by using tournament selection in each cluster with tournament size 2 (see Section 3.2.3). However, it can be argued that the selection pressure is already effectively enforced by Gene-pool Optimal Mixing because the quality of an offspring solution is always better or at least equal to its parent solution (see Section 3.2.4). Furthermore, linkage learning on the entire cluster is better for diversity preservation, which is important in MOEAs. Therefore, different from other MOEAs where the selection typically introduces a required search bias toward promising solutions, the selection phase for model building might not be necessary for MO-GOMEA. Figure 3.10 supports our claim. For each problem instance $l \in \{25, 50, 100, 200, 400\}$ and each MO-GOMEA variant $k \in \{1, 3, 5, 7, 10\}$ with/without selection, we perform 10 independent bisections to find the minimally-required population size and its corresponding number of fitness evaluations to reliably solve the problem instance. Figure 3.10 shows no deterioration in the scalability of all MO-GOMEA variants. There is no statistically significant difference in the required numbers of evaluations between the MO-GOMEA variants with and without the selection operator when solving Zeromax-Onemax. On Trap-Inverse Trap problem instances, the MO-GOMEA variants without selection perform slightly better than ones with selection. The differences between MO-GOMEAs $k = 5, 7, 10$ with and without selection are found to be statistically significant. Therefore, we deem the additional selection for model building to be unnecessary for MO-GOMEA and decide to remove the selection operator in all following experiments.

3.6.2. Eliminating the population size parameter

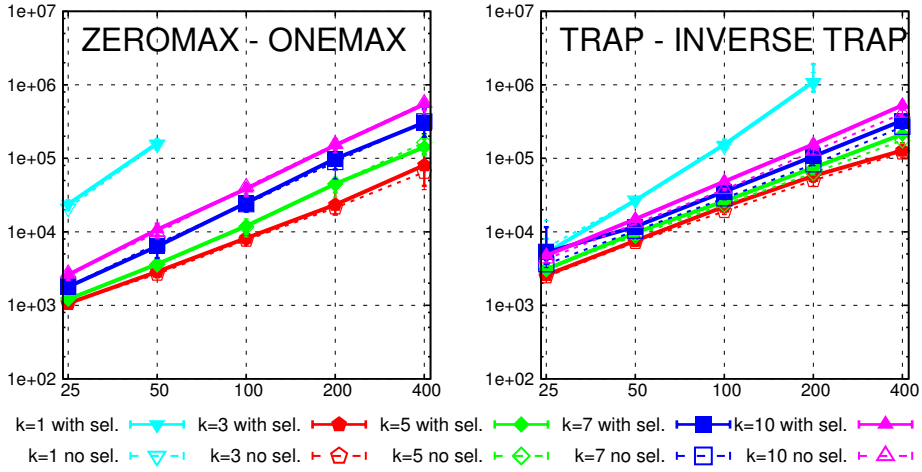

Figure 3.10: Scalability in terms of the required number of fitness evaluations for MO-GOMEAs with and without selection solving Zeromax-Onemax and Trap-Inverse Trap. Horizontal axis: problem size.

Adapting the Harik-Lobo population-sizing-free scheme for MO-GOMEA
Because of the proven effectiveness of the Harik-Lobo scheme in designing population-sizing-free variants of single-objective EAs [23, 29, 30], we here similarly employ the Harik-Lobo scheme to remove the population size parameter for multi-objective EAs, and in particular, for our MO-GOMEA. Note that related approaches such as designing an adaptive parameter control mechanism or in-depth studies for parameter tuning are interesting for future work but not in the scope of this thesis.

In an MOEA with the Harik-Lobo scheme, multiple populations P_i's of different sizes N_i's are maintained and are operated in an interleaved fashion. The first population P_1 has some small size N_1 (e.g., $N_1 = 8$). Next, every population P_{i+1} is created by doubling the size of the previous population P_i, i.e., $N_{i+1} = 2N_i$ for $i \geq 1$. For every b generations of population P_i, one generation of population P_{i+1} is run. In other words, population P_i executes a generational step every b^{i-1}-th generation of population P_1. When a population P_i converges, it will be terminated because, for the sake of simplicity, mutation operators are not employed. Converged populations, therefore, cannot explore the search space any further. The Harik-Lobo scheme has been experimented with on single-objective optimization problems with two commonly-used generation bases $b = 2$ in [24, 29] and $b = 4$ in [23, 30]. We will perform experiments to determine which generation base is more suitable in the multi-objective optimization context.

While the Harik-Lobo scheme works well for single-objective optimization, it requires some adaptations for multi-objective optimization. First, we do not necessarily need to employ the termination criterion of converged populations because MOEA populations normally do not converge to a single solution. In single-objective population-sizing-free EAs, because smaller populations are given a bigger or at least

equal number of fitness evaluations (i.e., generation base $b = 4$ or $b = 2$, respectively), if the average fitness of a population P_i is less than that of a larger population P_j ($j > i$), then that population P_i is regarded as costly and inefficient, and should be terminated. For population-sizing-free MOEAs, we do not employ the termination criterion of small populations based on average fitness values. Calculating the average fitness value for a population is not sensible in the multi-objective context. All populations of parameter-less MOEAs are thus kept running without termination of small populations. However, we will further discuss this issue in Section 3.8. Third, instead of establishing a race among completely independent populations as in single-objective population-sizing-free EAs, we share the elitist archive among all running populations so that the final approximation set is built from all populations of different sizes.

We also experiment with another population-sizing-free scheme with a different mechanism that has the advantage of requiring only a single population, albeit with a dynamically changing size. At the end of every generation i, the next population P_{i+1} is created as follows. If the current elitist archive size is bigger than the size N_i of the current population P_i, N_i well-spread solutions can be chosen from the archive by the leader selection heuristic presented in Section 3.2.2. If the current elitist archive size is smaller than N_i, a combined pool of solutions is formed by combining the population and the archive, and discarding duplicated solutions. If the size of this combined pool is still smaller than N_i (which might happen because of the deletion of duplicated solutions), new solutions are randomly generated to fill the remaining spots. If the size of the combined pool is larger than N_i, N_i solutions are then chosen from the best non-dominated fronts in the pool. This scheme was reported to alleviate a potential problem of MO-GOMEA that the new population might not be well-spread along the obtained-so-far non-dominated front [31]. This is due to the fact that only the last altered version of each parent solution is kept in the population, while all the intermediate solutions that have been accepted into the elitist archive during the Gene-pool Optimal Mixing procedure are not retained in the population [31]. After every generation, the population size is increased by adding new N_1 randomly-generated solutions, i.e., $N_{i+1} = N_i + N_1$ (N_1 is the initial population size, which can be set to some small number, e.g., $N_1 = 8$). In accordance with the fact that the Harik-Lobo scheme increases the population size in an exponential fashion, we name this mechanism the linear scheme. An MO-GOMEA variant with this linear scheme has been employed to solve the real-world multi-objective optimization of wind farm layouts [31].

Results

We put MO-GOMEA into the Harik-Lobo (HL) scheme to create two population-sizing-free MO-GOMEA variants with generation bases $b = 2$ and $b = 4$. We also create another population-sizing-free MO-GOMEA variant with the linear scheme as discussed above. For the sake of simplicity, we fix the number of clusters $k = 5$, which was previously found to give good results. We use these three MO-GOMEA variants to solve Zeromax-Onemax and Trap-Inverse Trap with different sizes $l = 25, 50, 100, 200, 400$. For every problem instance, we run each optimizer 100 times

independently and obtain the number of fitness evaluations until the whole Pareto-optimal front is found in each run. We perform the Mann-Whitney-Wilcoxon statistical hypothesis test for equality of medians with $p < 0.05$ to see whether the number of evaluations required by one optimizer is statistically different from that of another optimizer.

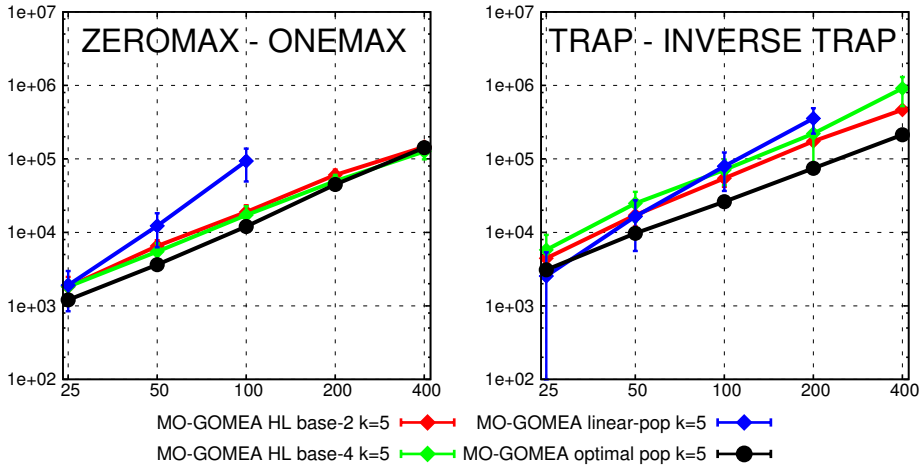

Figure 3.11: Experiments on eliminating the population size parameter for MO-GOMEA. Horizontal axis: Problem size. Vertical axis: The number of evaluations until the whole Pareto-optimal front is obtained.

Figure 3.11 shows the performance of MO-GOMEA implemented with different population-sizing-free schemes compared to the MO-GOMEA using the optimal population size settings found by bisection as aforementioned. The Harik-Lobo scheme with base 2 and 4 has similar performance on Zeromax-Onemax while the scheme with base 2 is the (statistically significantly) better one in solving Trap-Inverse Trap. This is due to the fact that diversity maintenance is of higher importance for multi-objective EAs compared to single-objective EAs. Smaller generation bases are faster in introducing larger populations with more diverse information. Both population-sizing-free MO-GOMEAs with base 2 and base 4 show some overhead compared to the MO-GOMEA using the optimal populations as a few generations are needed before populations with proper sizes are initialized. The linear scheme has the worst scalability in solving Zeromax-Onemax and Trap-Inverse Trap. This is for a large part due to the fact that this scheme constantly focuses on incrementally improving the non-dominated front formed by the elitist archive. To some extent, this can limit the ability of the optimizer to explore other regions of the Pareto-optimal front that might have structures very different to the obtained-so-far non-dominated front.

Based on the results in Figure 3.11, we decide that the Harik-Lobo scheme with generation base $b = 2$ is the most suitable population-sizing-free scheme for MO-GOMEA (among all the schemes tested here).

3.6.3. Eliminating the number of clusters parameter

Design of The Number-of-clusters-free Scheme

In this section, we aim to remove the number of clusters k parameter for MO-GOMEA. Every population P_i is characterized by its population size N_i and the number of clusters k_i. Because of the effectiveness of the Harik-Lobo population-sizing-free scheme, we accordingly propose three similar k-free mechanisms as follows.

The first mechanism starts with the initial population P_1 of some small size N_1 (we use $N_1 = 8$) and the number of clusters $k_1 = 1$ (i.e., no population clustering). For every even-indexed population P_i ($i = 2, 4, 6, \ldots$), the population size N_i doubles the size of the preceding population P_{i-1} while the number of clusters k_i remains the same, i.e., $N_i = 2N_{i-1},\ k_i = k_{i-1}$. For every odd-indexed population P_i ($i = 3, 5, 7, \ldots$), the population size N_i equals the size of the preceding population P_{i-1} while the number of clusters k_i increases by m clusters (m is the number of objectives), i.e., $N_i = N_{i-1},\ k_i = k_{i-1} + m$.

The second mechanism starts with the initial population P_1 of size N_1 and the number of clusters $k_1 = m + 1$ (m is the number of objectives). Section 3.5.2 shows that MO-GOMEA works better when employing population clustering with single-objective optimization for m extreme clusters and multi-objective optimization for middle clusters, which means every population should have at least $m + 1$ clusters. For every even-indexed population P_i ($i = 2, 4, 6, \ldots$), the population size N_i doubles the size of the preceding population P_{i-1} while the number of clusters k_i remains the same, i.e., $N_i = 2N_{i-1},\ k_i = k_{i-1}$. For every odd-indexed population P_i ($i = 3, 5, 7, \ldots$), the population size N_i equals the size of the preceding population P_{i-1} while the number of clusters k_i increases by $(m + 1)$ clusters (m is the number of objectives), i.e., $N_i = N_{i-1},\ k_i = k_{i-1} + (m + 1)$.

The third mechanism starts with the initial population P_1 of size N_1 and the number of clusters $k_1 = m + 1$ (m is the number of objectives). For every succeeding population P_i ($i = 2, 3, 4, \ldots$), the population size N_i doubles the size of the preceding population P_{i-1} while the number of clusters k_i increases by 1 cluster, i.e., $N_i = 2N_{i-1},\ k_i = k_{i-1} + 1$.

Results

Figure 3.12 shows the experimental results on solving Zeromax-Onemax and Trap-Inverse Trap with the base population-sizing-free MO-GOMEA $k = 5$ and its three k-free variants. For every problem instance, we run each optimizer 100 times independently and obtain the number of evaluations until the whole Pareto-optimal front is found in each run. The differences in the numbers of evaluations of all MO-GOMEA variants solving each problem instance are found to be statistically significant (except in the cases of Zeromax-Onemax $l = 25$ and Trap-Inverse Trap $l = 100$). The first variant MO-GOMEA $1 \rightarrow +m$ has a slightly better performance in solving Trap-Inverse Trap but the worst performance in case of Zeromax-Onemax. The second variant MO-GOMEA $m + 1 \rightarrow +(m + 1)$, on the other hand, performs slightly better in solving Zeromax-Onemax but it has the worst scalability in case of Trap-Inverse Trap. The third variant MO-GOMEA $m + 1 \rightarrow +1$ has balanced

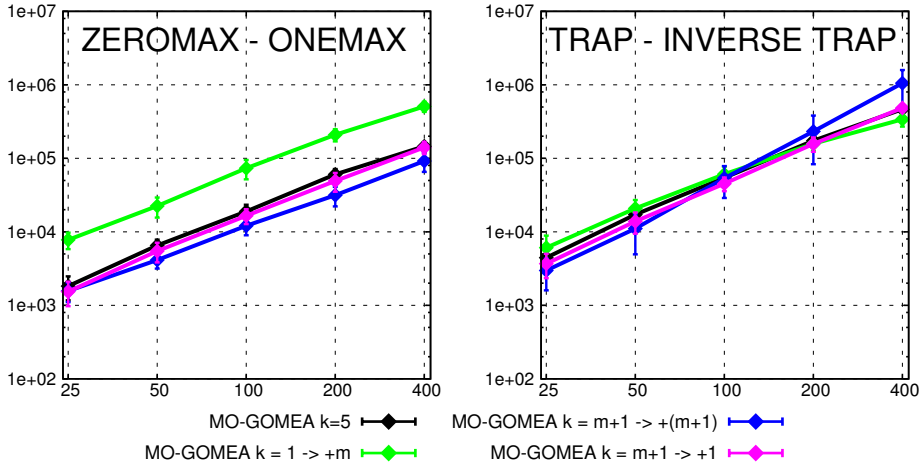

Figure 3.12: Experiments on eliminating the population size parameter for MO-GOMEA. Horizontal axis: Problem size. Vertical axis: The number of evaluations until the whole Pareto-optimal front is obtained.

results on both test problems and also the most similar performance with the base population-sizing-free MO-GOMEA $k = 5$. Therefore, we decide that the third k-free scheme $m + 1 \rightarrow +1$ is the most suitable method (among the methods tested here) to remove the number of clusters parameter.

3.7. Performance of the MO-GOMEA and the influence of mutation operators

In this section, we conduct experiments on all benchmark problems (as presented in Section 3.3) to study the performance of our newly created parameter setting-free MO-GOMEA and the influence of the use of different mutation operators.

3.7.1. Design of mutation operators

The standard version of MO-GOMEA works well without any mutation operators. However, empirical results showed that mutation could be beneficial to the performance of MO-GOMEA on some problems. We here implement mutation as an additional component that the user can easily switch on or off as desired. Mutation is employed with some probability p_m on every problem variable in the linkage set at every mixing step during the Optimal Mixing procedure. Mutation should be performed after copying values from a donor to the current solution and before fitness evaluation of the intermediate solution. We propose two mutation operators: weak mutation and strong mutation. Weak mutation uses a fixed mutation probability $p_m = 1/l$ (l is the number of problem variables). Strong mutation uses an adaptive mutation probability $p_m = 1/l_{\boldsymbol{F}^i}$, where $l_{\boldsymbol{F}^i} = \left|\boldsymbol{F}^i\right|$ is the number of

problem variables in a linkage set $\boldsymbol{F}^i$.

3.7.2. Scalable benchmark problems

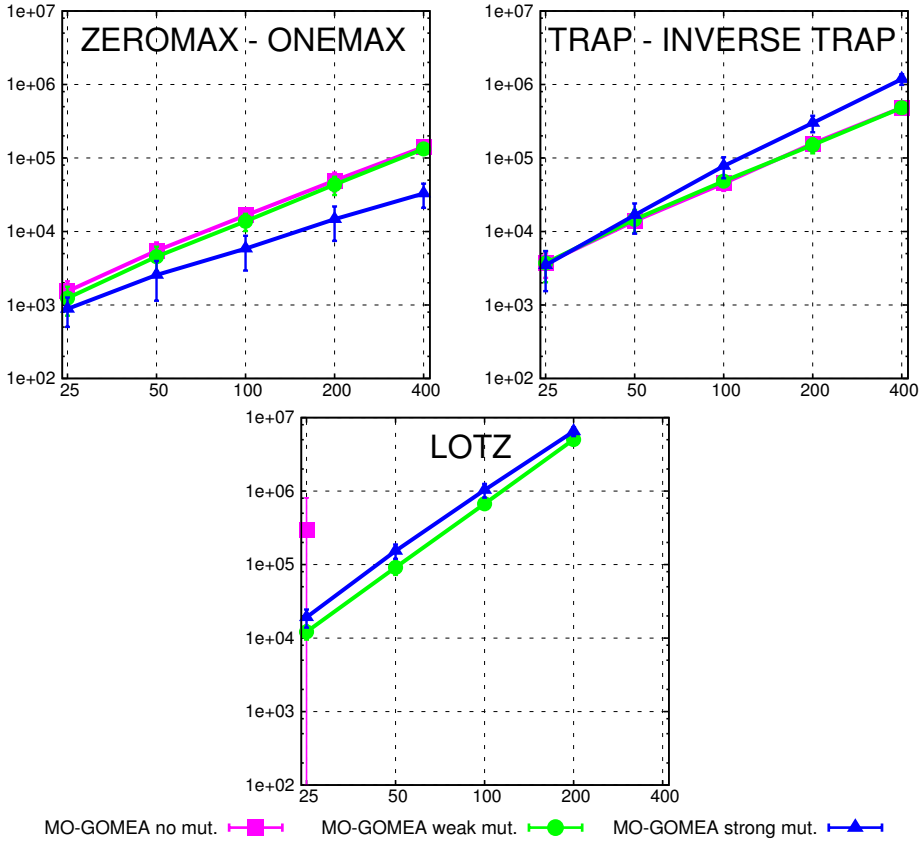

Figure 3.13: Performance of MO-GOMEA without mutation and with weak/strong mutation on scalable benchmark problems. Horizontal axis: Problem size. Vertical axis: The number of evaluations until the whole Pareto-optimal front is obtained.

Figure 3.13 shows that MO-GOMEA with the strong mutation operator is the fastest solver for Zeromax-Onemax. As all problem variables are independent from each other, linkage learning is not required to solve Zeromax-Onemax and a mutation operator with high mutation probability p_m will help obtain the whole Pareto-optimal front quicker by simply performing bit-flips to try out different alternatives without disrupting any building blocks. Such aggressive mutation operators, however, worsen the performance of MO-GOMEA on problems that require linkage learning such as Trap-Inverse Trap. Weak mutation has little influence on the scalability of MO-GOMEA in solving Zeromax-Onemax and Trap-Inverse Trap because its mutation probability becomes increasingly smaller as the problem size increases.

While MO-GOMEA with weak mutation performs slightly better than MO-GOMEA without mutation in solving Zeromax-Onemax, there is no statistically significant difference between the two optimizers in solving Trap-Inverse Trap.

MO-GOMEA without mutation operators has difficulties in solving the LOTZ problem, especially in approaching two extreme solutions (i.e., a string of all 1s and a string of all 0s). As discussed in Section 3.5.2, only leading 1 bits and trailing 0 bits contribute to the objective values during the optimization process. It can be seen that all solutions containing trailing 1 bits and leading 0 bits can easily be dominated and replaced by any solutions ending with a 0 and beginning with a 1. Those trailing 1s and leading 0s, however, are essential to the construction of the extreme solutions at the later stage of the optimization process when they meet the leading 1s and trailing 0s. Gene-pool Optimal Mixing, therefore, cannot find any donor with trailing 1s nor leading 0s remaining in the populations when necessary. This problem, however, can be alleviated by employing mutation operators. Both weak and strong mutation operators significantly improve the performance of MO-GOMEA when solving the LOTZ problem by bringing back the prematurely disappeared bit values.

While these three benchmarks are convenient for scalability benchmarking because all Pareto-optimal solutions can be computed analytically, their Pareto-optimal fronts always have the shape of a straight line and they therefore do not resemble real-world optimization problems. In the following section, we will consider MAXCUT, a problem whose Pareto-optimal fronts can have multiple regions with different shapes, and knapsack, an optimization problem with constraints.

3.7.3. MAXCUT & Knapsack

Figure 3.14 shows the average convergence behavior of MO-GOMEA and the influence of weak and strong mutation in solving four MAXCUT problem instances. The weak mutation operator has little influence on the performance of MO-GOMEA. The strong mutation operator improves the convergence in the 100-vertex instance but worsens the performance in the 50-vertex instance; the differences are found to be statistically significant. Nevertheless, all the performance gaps between MO-GOMEA without mutation and MO-GOMEA with mutation are small. In these MAXCUT problem instances, the linkage information-guided solution recombination of Gene-pool Optimal Mixing is effective enough for MO-GOMEA to explore the search space without requiring any mutation operators.

Figure 3.15 shows the average convergence performance of MO-GOMEA and the influence of mutation operators in solving four multi-objective Knapsack problem instances. MO-GOMEA without mutation has the best convergence performance in all cases. The weak mutation operator exhibits little effects while the strong mutation operator severely degrades the performance of MO-GOMEA. Mutation operators with high mutation rates p_m can disrupt important building blocks acquired by the linkage learning and Gene-pool Optimal Mixing procedures.

While mutation is a crucial variation operator of other MOEAs, it is not a required component for MO-GOMEA. Among all benchmark results, mutation operators only significantly deliver positive improvements when solving the LOTZ

problem. Therefore, we conclude that the type of mutation and the probability of mutation should be, much like local search operators, considered as highly problem-specific and, therefore, should not be seen as a default component of MO-GOMEA.

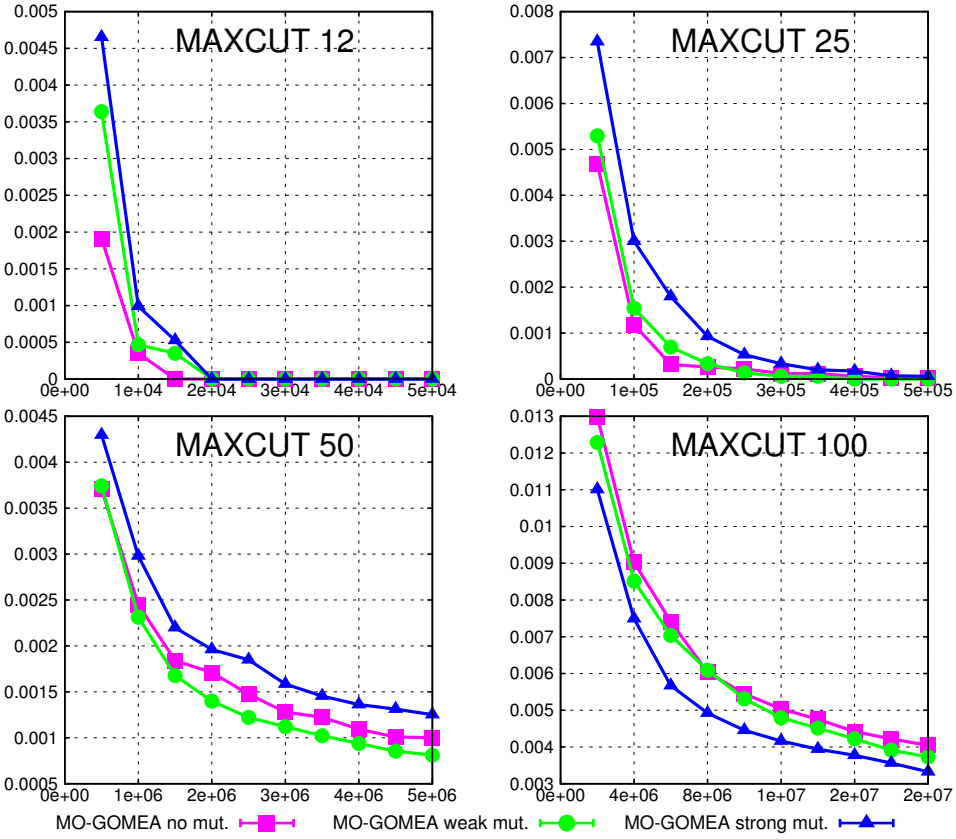

Figure 3.14: Average convergence performance of MO-GOMEA without mutation and with weak/strong mutation on MAXCUT. Horizontal axis: number of evaluations (both objectives per evaluation). Vertical axis: $D_{\mathcal{P}_F \to \mathcal{S}}$.

3.8. Comparison with the NSGA-II and the influence of stopping (inefficient) small populations

3.8.1. A population-sizing-free NSGA-II

For comparison purposes, we place the well-known NSGA-II [7] into the Harik-Lobo scheme that we developed for MOEAs in Section 3.6 to construct a population-sizing-free NSGA-II. We conduct experiments for this NSGA-II version with both generation base 2 and 4 as we do not yet know which values would be more appropriate for the operation of NSGA-II. Note that we only need to remove the population size parameter for NSGA-II because the standard version of NSGA-

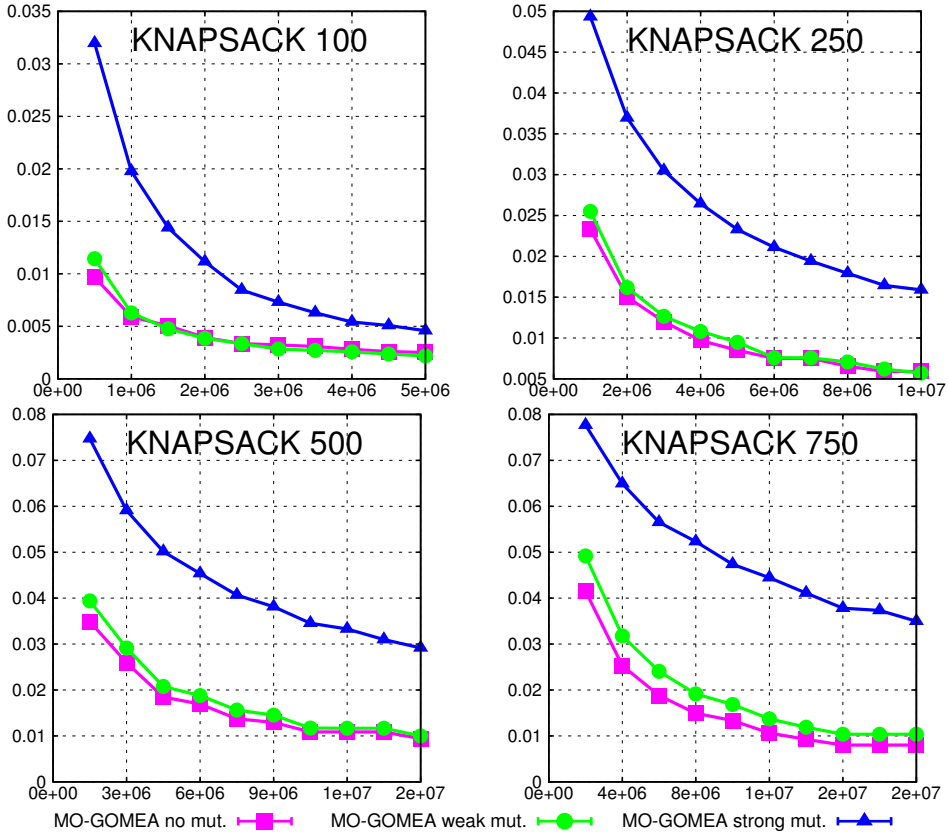

Figure 3.15: Average convergence performance of MO-GOMEA without mutation and with weak/strong mutation on Knapsack. Horizontal axis: number of evaluations (both objectives per evaluation). Vertical axis: $\boldsymbol{D}_{\mathcal{P}_F \to \mathcal{S}}$.

II does not employ objective-space population clustering. While mutation is not a required component in MO-GOMEA, it is crucial for the search mechanism of NSGA-II. Therefore, we keep the mutation operator with probability $p_m = 1/l$ (l is the number of problem variables) as in the original NSGA-II [7].

3.8.2. A termination criterion for (inefficient) small populations

Research on population-sizing-free EAs for single-objective optimization, as in [23, 30], emphasized the importance of terminating smaller populations when they are found to be less efficient compared to a larger population. More specifically, when a population of size N_i reaches an average fitness at least as good as the average fitness of the population of size N_{i-1}, all populations of size N_k with $k < i$ will be terminated [30]. It is not directly clear however that this requirement transfers to the multi-objective case, especially for MO-GOMEA. This is because the elitist

archive is shared among all populations and the Forced Improvement phase in MO-GOMEA causes substantial interaction with the elitist archive, and thus substantial interaction between populations that may increase the efficiency of smaller populations. In this section, we therefore explicitly wish to study the impact of stopping smaller populations in MO-GOMEA. It is more difficult, however, to derive a proper metric to measure and compare the qualities of different populations during a multi-objective optimization process. The average fitness of a population is not a suitable metric in the multi-objective optimization context because the fitness value of each candidate solution involves multiple conflicting objectives. Even the popular hypervolume performance indicator [27] is not an appropriate metric. A population P_{i-1} of size N_{i-1} can have a smaller hypervolume value than the population P_i of size N_i but P_{i-1} is then not necessarily worse than P_i because they could be approaching different regions of the Pareto-optimal front. Furthermore, the choice of a reference point that is needed to calculate hypervolume values is against our purpose of designing a parameter-less MOEA.

We here employ the Pareto domination concept to determine when a small population should be terminated due to inefficiency. A population P_i of size N_i is considered to be inefficient compared to a population P_j of size $N_j > N_i$ if the front formed by P_i is totally Pareto-dominated by the front formed by P_j or if all points on the front formed by P_i also exist on the front formed by P_j. We incorporate this condition into both MO-GOMEA and NSGA-II, and perform the experiments again to observe its influence on the optimizers.

3.8.3. Experimental results

Scalable benchmark problems

Figure 3.16 shows the average number of evaluations (over 100 independent runs) spent by MO-GOMEA and NSGA-II on scalable benchmark problems until the whole Pareto-optimal fronts are obtained. MO-GOMEA variants clearly outperform NSGA-II variants on solving the Trap-Inverse Trap problem. Trap functions can only be efficiently solved by optimizers that have linkage learning abilities, which NSGA-II does not employ. For the Zeromax-Onemax problem, where all variables are independent and linkage learning is not necessary, MO-GOMEA variants still have a better scalability than NSGA-II variants. NSGA-II variants perform better than MO-GOMEA variants, however, when solving the LOTZ problem due to the mutation operator as discussed in Section 3.5.2. Section 5 above shows that if MO-GOMEA is coupled with a mutation operator, it can also solve LOTZ easily and does so more efficiently than NSGA-II. For these three scalable benchmark problems, the termination criterion of small populations shows no influence on MO-GOMEA while it does result in small improvements for NSGA-II (the differences are found to be statistically significant).

Section 3.5.2 shows that NSGA-II with a population of size 4 can solve all Zeromax-Onemax and LOTZ problem instances more efficiently compared to MO-GOMEA. In fact, for these two problems, NSGA-II does not need to employ any solution recombination but only a mutation operator. NSGA-II simply needs to run many generations and wait for the right bits to be flipped at the right time

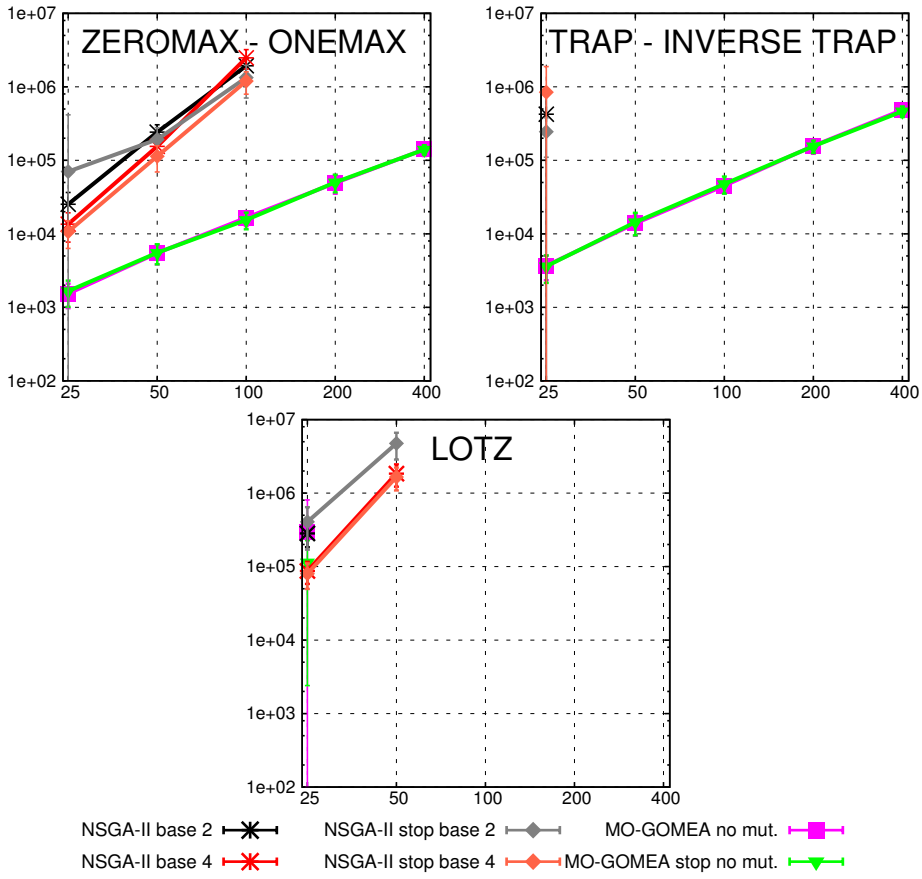

Figure 3.16: Performance of MO-GOMEA and NSGA-II when terminating small populations on scalable benchmark problems. Horizontal axis: Problem size. Vertical axis: The number of evaluations until the whole Pareto-optimal front is obtained.

to obtain a Pareto-optimal solution. The population-sizing-free NSGA-II variant, however, cannot solve Zeromax-Onemax nor LOTZ as efficiently as the original NSGA-II. The population-sizing-free scheme that we employ here introduces too many large populations for NSGA-II too quickly and all fitness evaluations are used up before the meaningful mutation events occur.

MAXCUT & Knapsack

Figures 3.17 and 3.18 show the performance of MO-GOMEA and NSGA-II with and without terminating inefficient populations on MAXCUT and knapsack respectively. In almost all cases, the termination criterion improves (statistically) significantly the performance of NSGA-II. This confirms the fact that when small populations are inefficient for NSGA-II, they should be terminated as soon as possible so that fitness

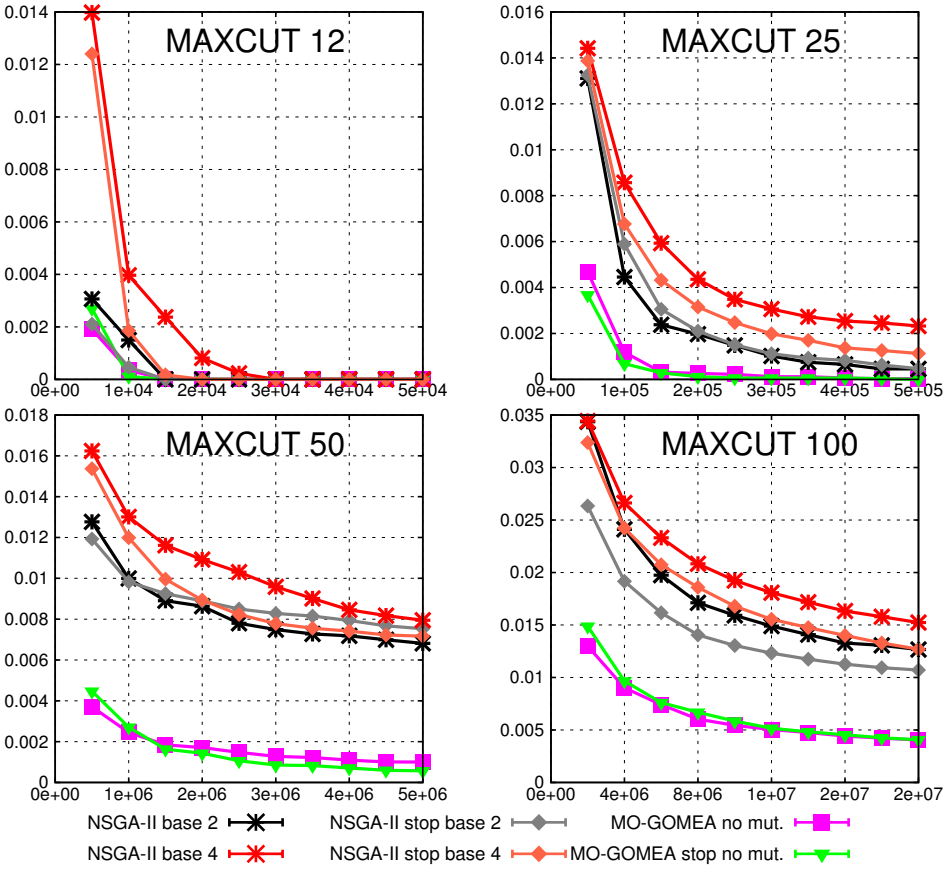

Figure 3.17: Average convergence performance of MO-GOMEA and NSGA-II when terminating small populations on MAXCUT. Horizontal axis: number of evaluations (both objectives per evaluation). Vertical axis: $D_{\mathcal{P}_F \to \mathcal{S}}$.

evaluations would not be wasted on them. Additionally, Figures 3.17 and 3.18 show again that base 2 is a better setting for NSGA-II than base 4. It was suggested that smaller base values are more suitable for the parameter-less GA if it suffers from the effects of genetic drift [29]. Diversity preservation is an important task in multi-objective optimization, and the base-2 parameter-less scheme introduces larger population sizes with more diverse candidate solutions at a faster rate than the base-4 scheme. NSGA-II with base 2 coupled with the termination criterion of small populations can get rid of small and inefficient populations and move to sufficiently larger populations more quickly.

The termination criterion of small populations, however, shows little or insignificant influence on the performance of MO-GOMEA. This suggests that MO-GOMEA can operate effectively with small populations and that indeed the shared use of the elitist archive in all populations make terminating smaller, inefficient populations

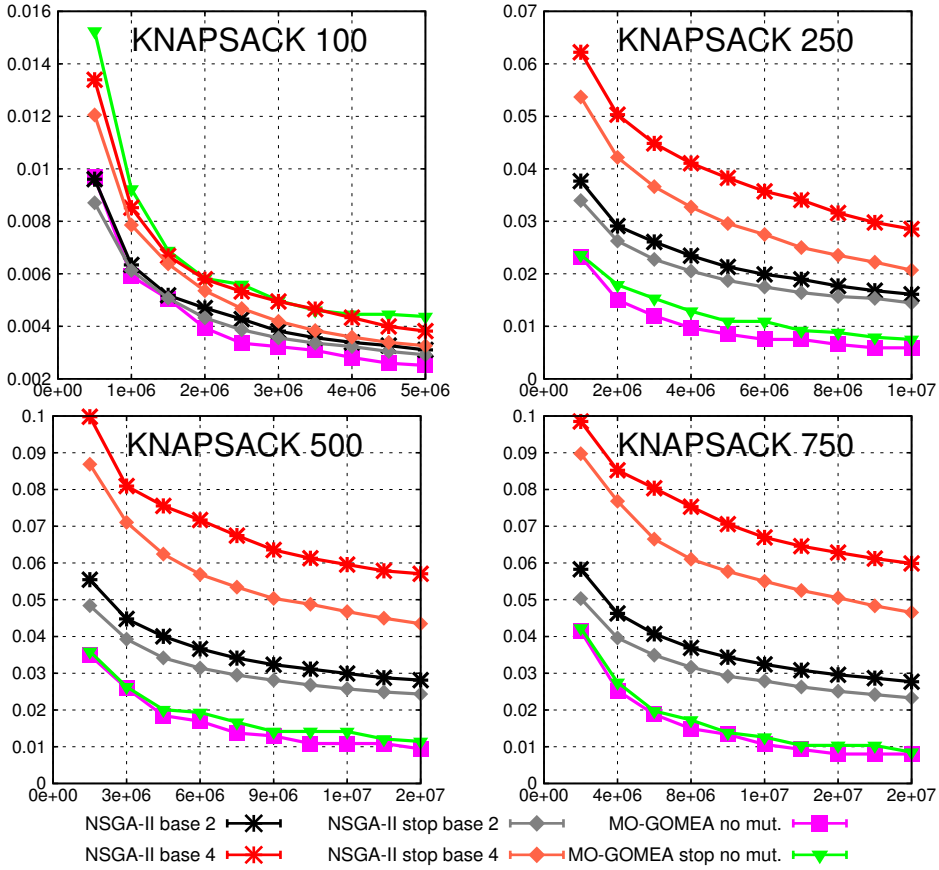

Figure 3.18: Average convergence performance of MO-GOMEA and NSGA-II when terminating small populations on Knapsack. Horizontal axis: number of evaluations (both objectives per evaluation). Vertical axis: $D_{\mathcal{P}_F \to \mathcal{S}}$.

unnecessary. What we observe here conforms with previous research on the scalability of GOMEAs in single-objective optimization, in which GOMEAs generally have minimally required population sizes that are much smaller than other population-based EAs [12, 26].

3.9. Conclusions

We have presented the multi-objective GOMEA (MO-GOMEA). We have shown that for the combination with the linkage tree model, superior scalability for solving different classes of MO optimization problems can be achieved as compared to classic GAs (i.e. NSGA-II) and even state-of-the-art EDAs (i.e. mohBOA). Our experimental results further support that the key features of scalable MO optimizers that we identified and incorporated into MO-GOMEA are indeed responsible for

the observed performance. These features are: an elitist archive to keep track of the non-dominated front, clustering to process different regions of the front differently, linkage learning and an efficient mechanism for exploiting the learned linkage relations to generate offspring solutions.

Population clustering ensures that MO-GOMEA allocates an equal amount of search effort to every region and the whole Pareto-optimal front can thus be evenly approached. Especially the cluster-based operating mechanism of MO-GOMEA is convenient for dedicated adaptations if different regions of the Pareto-optimal front have different characteristics and thus require different strategies to exploit problem structure effectively and efficiently. In the multi-objective knapsack benchmark (see Sections 3.7 and 3.8), by clustering the working population, it is straightforward to assign the multi-objective repair mechanism to the middle-region clusters and the suitable single-objective repair mechanism to the corresponding extreme-region cluster. Population clustering helps MO-GOMEA score on the diversity part of the $\boldsymbol{D}_{\boldsymbol{\mathcal{P}}_{\boldsymbol{F}} \to \mathcal{S}}$ performance indicator.

As each cluster of MO-GOMEA approaches a specific region of the Pareto-optimal front, linkage learning captures problem-variable dependencies that are relevant to that region. Following the structure of the linkage tree dedicatedly learned from a cluster, the Gene-pool Optimal Mixing operator creates new candidate solutions by juxtaposing currently existing building blocks in a way that is specifically suitable to that cluster. The genetic local search nature of Gene-pool Optimal Mixing also ensures that an offspring is better or at least as good as its parent solution. Linkage learning and Gene-pool Optimal Mixing together ensure that the building blocks relevant to each cluster are detected and propagated to ensure effective convergence toward the Pareto-optimal front, helping MO-GOMEA score on the proximity part of the $\boldsymbol{D}_{\boldsymbol{\mathcal{P}}_{\boldsymbol{F}} \to \mathcal{S}}$ performance indicator.

The combined effect of clustering the population and exploiting linkage information results in the better performance for MO-GOMEA over other MOEAs. We then made MO-GOMEA an easy-to-use solver by placing MO-GOMEA in a population-sizing-free framework that eliminates the required setting of the population size parameter, which is notoriously difficult for any population-based EA, and the number-of-clusters parameter. As a consequence, users now only specify how long the algorithm is allowed to run. Alternatively, MO-GOMEA can be used as an anytime algorithm, i.e., the more time it runs, the better solutions would be found, and it can be terminated when a satisfying solution is obtained. The parameter setting-free MO-GOMEA was shown to retain the scalability of the original MO-GOMEA and to have excellent performance on a wide range of benchmark problems. The scalability and usability of MO-GOMEA suggest that MO-GOMEA is a promising solver for tackling complicated (real-world) multi-objective optimization problems. We look at the application of MO-GOMEA to solving the multi-objective DNEP problem later in Chapter 6.

References

[1] N. H. Luong, H. La Poutré, and P. A. N. Bosman, *Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithms,* in *Proceedings of the Genetic*

and Evolutionary Computation Conference - GECCO 2014 (ACM Press, New York, New York, USA, 2014) pp. 357–364.

[2] N. H. Luong, H. La Poutré, and P. A. N. Bosman, *Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm with the Interleaved Multi-start Scheme,* Swarm and Evolutionary Computation **40**, 238 (2018).

[3] D. A. Van Veldhuizen and G. B. Lamont, *Multiobjective evolutionary algorithms: analyzing the state-of-the-art.* Evolutionary Computation **8**, 125 (2000).

[4] C. M. Fonseca and P. J. Fleming, *An overview of evolutionary algorithms in multiobjective optimization,* Evolutionary Computation **3**, 1 (1995).

[5] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms* (John Wiley & Sons, Inc., 2001).

[6] P. A. N. Bosman and D. Thierens, *The balance between proximity and diversity in multiobjective evolutionary algorithms,* IEEE Transactions on Evolutionary Computation **7**, 174 (2003).

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II,* IEEE Transactions on Evolutionary Computation **6**, 182 (2002).

[8] E. Zitzler, M. Laumanns, and L. Thiele, *SPEA2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimization,* in *Evolutionary Methods for Design, Optimisation, and Control* (CIMNE, Barcelona, Spain, 2002) pp. 95–100.

[9] Q. Zhang and H. Li, *MOEA/D: A multiobjective evolutionary algorithm based on decomposition,* IEEE Transactions on Evolutionary Computation **11**, 712 (2007).

[10] K. Deb and H. Jain, *An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints,* IEEE Transactions on Evolutionary Computation **18**, 577 (2014).

[11] M. Pelikan, K. Sastry, and D. E. Goldberg, *Multiobjective hBOA, clustering, and scalability,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2015* (ACM Press, New York, New York, USA, 2005) pp. 663–670.

[12] D. Thierens and P. A. N. Bosman, *Optimal mixing evolutionary algorithms,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2011* (ACM Press, New York, New York, USA, 2011) pp. 617–624.

[13] E. Carrano, L. Soares, R. Takahashi, R. Saldanha, and O. Neto, *Electric distribution network multiobjective design using a problem-specific genetic algorithm,* IEEE Transactions on Power Delivery **21**, 995 (2006).

[14] N. Khan, D. E. Goldberg, and M. Pelikan, *Multi-objective Bayesian optimization algorithm. IlliGAL Report No. 2002009*, Tech. Rep. (University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2002).

[15] M. Laumanns and J. Ocenasek, *Bayesian Optimization Algorithms for multi-objective optimization,* in *Parallel Problem Solving from Nature - PPSN VII*, Lecture Notes in Computer Science, Vol. 2439 (Springer, 2002) pp. 298–307.

[16] M. Pelikan, K. Sastry, and E. Cantú-Paz, eds., *Scalable Optimization via Probabilistic Modeling*, Studies in Computational Intelligence, Vol. 33 (Springer, 2006).

[17] P. A. N. Bosman, *The anticipated mean shift and cluster registration in mixture-based EDAs for multi-objective optimization,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2010* (ACM Press, New York, New York, USA, 2010) pp. 351–358.

[18] I. Gronau and S. Moran, *Optimal implementations of UPGMA and other common clustering algorithms,* Information Processing Letters **104**, 205 (2007).

[19] C. A. Coello Coello, G. Pulido, and E. Montes, *Current and future research trends in evolutionary multiobjective optimization,* in *Information Processing with Evolutionary Algorithms*, Advanced Information and Knowledge Processing (Springer London, 2005) pp. 213–231.

[20] J. Knowles and D. Corne, *Properties of an adaptive archiving algorithm for storing nondominated vectors,* IEEE Transactions on Evolutionary Computation **7**, 100 (2003).

[21] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, *Combining convergence and diversity in evolutionary multiobjective optimization.* Evolutionary Computation **10**, 263 (2002).

[22] K. Sastry, M. Pelikan, and D. E. Goldberg, *Decomposable problems, niching, and scalability of multiobjective estimation of distribution algorithms. IlliGAL Report No. 2005004*, Tech. Rep. (University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2005).

[23] G. R. Harik and F. G. Lobo, *A parameter-less genetic algorithm,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 1999* (Morgan Kaufmann, 1999) pp. 258–265.

[24] M. Pelikan, A. Hartmann, and T.-K. Lin, *Parameter-less hierarchical Bayesian optimization algorithm,* in *Parameter Setting in Evolutionary Algorithms. Studies in Computational Intelligence.*, Vol. 54 (Springer Berlin Heidelberg, 2007) pp. 225–239.

[25] N. H. Luong and P. A. N. Bosman, *Elitist archiving for multi-objective evolutionary algorithms: To adapt or not to adapt,* in *Parallel Problem Solving from Nature - PPSN XII*, Lecture Notes in Computer Science, Vol. 7492, edited by C. A. C. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, and M. Pavone (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012) pp. 72–81.

[26] P. A. N. Bosman and D. Thierens, *More concise and robust linkage learning by filtering and combining linkage hierarchies,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2013* (ACM Press, New York, New York, USA, 2013) pp. 359–366.

[27] E. Zitzler and L. Thiele, *Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach,* IEEE Transactions on Evolutionary Computation **3**, 257 (1999).

[28] M. Pelikan, *Hierarchical Bayesian Optimization Algorithm - Toward a new Generation of Evolutionary Algorithms*, Studies in Fuzziness and Soft Computing, Vol. 170 (Springer, 2005) pp. 1–149.

[29] M. Pelikan and F. G. Lobo, *Parameter-less genetic algorithm: A worst-case time and space complexity analysis. IlliGAL Report No. 99014*, Tech. Rep. (University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL., 1999).

[30] J. C. Pereira and F. G. Lobo, *A Java implementation of parameter-less evolutionary algorithms.* CoRR **abs/1506.0** (2015).

[31] S. Rodrigues, P. Bauer, and P. A. Bosman, *Multi-objective optimization of wind farm layouts: Complexity, constraint handling and scalability,* Renewable and Sustainable Energy Reviews **65**, 587 (2016).

4

Static Distribution Network Expansion Planning

Meet driemaal eer gij eens snijdt.
Measure thrice, cut once.

Dutch proverb

This chapter tackles the Distribution Network Expansion Planning (DNEP) problem that has to be solved by distribution network operators to decide which enhancements to electricity networks should be introduced to satisfy the future power demands. We consider three types of evolutionary algorithms (EAs) for optimizing expansion plans: the classic Genetic Algorithm (GA), the Estimation-of-Distribution Algorithm (EDA), and the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA). Not fully knowing the structure of the problem, we study the effect of linkage learning through the use of three linkage models: univariate, marginal product, and linkage tree. We furthermore experiment with the impact of incorporating different levels of problem-specific knowledge in the variation operators. Based on experimental results, we suggest that when selecting optimization algorithms for real-world applications like DNEP, EAs that have the ability to effectively model and efficiently exploit problem structures, such as GOMEA, should be given priority, especially in the case of black-box or grey-box optimization. The best performance is obtained when both linkage information and problem-specific knowledge can be exploited.

Parts of this chapter have been presented at *EA '13* [1], *PSCC '14* [2], *GECCO '15* [3] and published in [4].

4.1. Introduction

Peak loads on distribution networks normally increase every year due to developments in residential and industrial electricity consumption. Consequently, the magnitude of the power flows that are carried through network components (e.g., cables, transformers) to satisfy customers' power demands will at some point exceed the currently existing network capacity. In order for distribution networks to work properly, distribution network operators (DNOs) have to ensure that the capacities of network assets are sufficient to handle the magnitude of the required power flows. Otherwise, bottlenecks can cause overloads, which heat up the cable wires. This is detrimental to the normal operation and safety of the networks, and may cause blackouts or earlier asset replacements. Therefore, DNOs need to perform distribution network expansion planning (DNEP) to determine *where* on the networks asset reinforcements should be made and *what* types of devices should be installed there. The dynamic DNEP formulation also involves the question *when* those enhancement activities should be started during the planning period while in the static DNEP formulation this time-dependent decision making issue is omitted. The static DNEP problem is the focus of this chapter, and its dynamic version will be tackled in Chapter 5. The goal of DNEP is to find the most economical expansion plan, in terms of investment and/or operation costs, for which the network satisfies the power demand over the planning period.

Evolutionary algorithms (EAs) have been widely applied and achieved practical results in DNEP, see e.g., [5–8]. This is mostly due to the straightforward implementation and broad applicability of EAs. However, most DNEP studies in literature overlook several important issues when employing EAs. First, experiments are usually conducted by using only one, arbitrarily chosen, EA with a customized problem-specific variation operator (VO), omitting both questions why that specific EA should be chosen over other available EAs and what the advantages that VO has compared to other alternatives. Second, the comparison of how effective various constraint-handling mechanisms help the solvers traverse the search space is often disregarded. In this chapter, while aiming to solve a formulation of the DNEP problem that captures many important real-world considerations, we also address these issues. We employ three EA solvers: a classic Genetic Algorithm (GA), a Estimation-of-Distribution Algorithm (EDA), and a Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [9, 10]. The GA is arguably the most popular EA in DNEP literature, but it is rarely used out of the box in practice. Practitioners often customize its VOs (i.e., crossover and mutation) with expert and problem-specific knowledge (PSK) so that important problem structures are respected during variation, e.g., cables in the same feeder group in the network should be treated together when constructing new networks. Taking the perspective of black-box optimization, where such PSK is assumed to be hardly available, linkage learning (LL) can be performed to identify, during optimization, which variables are inter-dependent and should thus be jointly considered when generating new solutions. EDAs, such as BOA [11] or ECGA [12], are well-known examples of EAs that build probabilistic models that exhibit a degree of variable dependency that is aligned with variable linkage to effectively generate high-quality solutions. How-

ever, large population sizes are often required so that probabilistic models can be properly constructed. Building a high-order probabilistic model also introduces significant additional computation time requirements. Being a recently-developed LL EA, GOMEA focuses specifically on linkage, without estimating associated probability distributions, allowing higher-order models to be built much more efficiently. GOMEA also has an efficient variation operator that exploits the learned linkage model to create new solutions that are better or at least equal to existing solutions. GOMEA has been shown to have superior performance and scalability on laboratory benchmarks and recently in power system optimization as well [1, 2]. Linkage learning does not exclude the possibility of combining linkage knowledge with PSK if available. In this chapter, we show how to combine the strength of LL with PSK exploitation.

Population size parameter settings have big impacts on how effective and how efficient problem instances are solved. Population sizes of EAs are often chosen arbitrarily or are customized to the specific problem instance at hand in DNEP literature [5–7]. Practitioners often need to manually try different population sizes to figure out a suitable population size for each problem instance, which is both time-consuming and difficult for comparing the performance of different EAs fairly. This approach is also difficult to generalize to other applications. To get rid of this troublesome parameter, the so-called parameter-less GA with a population-sizing-free scheme was firstly proposed in [13]. A simplified implementation of this scheme was then presented in [14, 15]. The population-sizing-free scheme (hereafter referred to as the Harik-Lobo scheme in reference to the first authors), however, has been mainly applied to unconstrained problems. Here, we adapt the Harik-Lobo scheme in the context of DNEP, a highly constrained optimization problem. We then also employ the adapted Harik-Lobo scheme as a framework for comparing the performance of GA, EDA, and GOMEA.

The remainder of this chapter is organized as follows. Section 4.2 formulates the DNEP problem. Section 4.3 presents the benchmark networks and our experiment setup. Section 4.4 shows the benefit of employing PSK for population initialization on the performance of different solvers. Section 4.5 presents how we adapt the Harik-Lobo population-sizing-free scheme for the DNEP problem. Section 4.6 describes how PSK can be used to design different variation operators and constraint-handling techniques for EAs solving DNEP. The performance of GA, EDA, and GOMEA in combination with different variation operators is also presented in Section 4.6. Finally, Section 4.7 concludes the chapter.

4.2. Optimization problem formulation

In this thesis, distribution network modeling is based on the specifications for medium voltage distribution (MV-D) networks in the Netherlands [2], which are also common for other highly urbanized regions/countries. A typical MV-D network contains lines/cables branching out of MV transmission substations (or HV/MV transformer substations) connecting MV nodes (i.e., MV/LV transformer substations or MV customer substations) into distribution rings or mesh structures. Some specific lines/cables, called normally open points (NOPs), are opened on one side

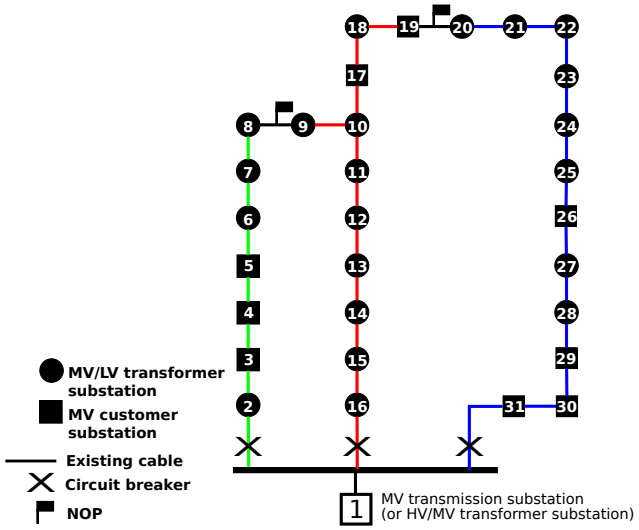

Figure 4.1: An example MV distribution network. The three collections of cables having red, green, and blue colors are three feeders.

and carry no power flow in normal operation. Therefore, the physical topology of an MV-D network is a mesh grid but its operational topology is a radial grid. An MV-D network consists of a number of *feeders*. The commonly used term *feeder* is formally defined as follows. A feeder is a collection of lines/cables that corresponds to a maximal subnetwork that does not contain any NOP or supply substation (i.e., a subtree). Normally, *lines* refer to overhead connections, and *cables* refers to underground connections in the network. Here, without loss of generality, we consider MV-D networks containing only underground cables, which are typically found in urban areas. Figure 4.1 shows an example of an MV-D network.

Cables and MV/LV transformers are two typical types of assets in MV-D networks. They can be modeled by their equivalent impedances so that both can be abstractly seen as branches in a network [16]. In this study, we focus on MV-D cables as our main asset category but the inclusion of transformers can be done without significant changes to the problem formulation. Each cable has a nominal capacity that defines the maximum apparent power that it can carry. Increasingly higher power demands require increasingly greater power flows, and bottlenecks happen when any cable is overloaded. The network should be enhanced beforehand to avoid such bottlenecks. Enhancement options consist of activities to increase the network capacity: replacing legacy cables with higher-capacity cables, installing new cables to connect neighboring distribution networks, or installing a new cable to connect the MV transmission substation (or HV/MV transformer substation) with an MV node [2]. Note that, within an MV distribution network, installing a new cable that connects an MV transmission substation (or HV/MV transformer substation) with an MV node results in a significant increase in network capacity, and is typically preferable to installing a new cable that connects two MV nodes. Installations of

new cables require additional placements of corresponding NOPs to keep the network operating radially [2]. In the remainder of this section we define the variables, objectives, and constraints that are required to define an optimization problem to be solved that corresponds to making the best decision to expand a distribution network.

4.2.1. Decision variables

Let l_e denote the number of currently existing cables in MV-D a network. Let l_p denote the number of potential cable connections (i.e., pairs of nodes that are not directly connected at the moment but are eligible to become directly connected by a cable in the planning horizon). As the possible cable connections are numerous, expert knowledge should be applied here to disregard impractical or undesirable cable connections, such as connections between distant nodes. Let l denote the total number of branches (cable connections) to be considered in the optimization process, i.e., $l = l_e + l_p$. A distribution network can be represented as a vector of l integer elements:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_l), \quad |x_k| \in \Omega(k) \cup \{0\}, \quad k \in \{1, 2, \ldots, l\} \tag{4.1}$$

where $\Omega(k)$ is the set of cable types that can be installed at the k^{th} branch ($\Omega(k) \subseteq \mathbb{N}$). The value of x_k indicates the status and the type of cable installed:

- $x_k = ID > 0$: A cable of type $ID \in \Omega(k)$ is installed.
- $x_k = 0$: No cable is installed at the k^{th} branch.
- $x_k = -ID < 0$: A cable of type $ID \in \Omega(k)$ is installed but is out of normal operation. This is an NOP.

If $\boldsymbol{x}^0 = (x_1^0, x_2^0, \ldots, x_l^0)$ is the configuration of the currently existing network, then each x_k^0 for which k corresponds to a potential branch is assigned the value 0. Regarding the predicted annual peak load growth rate R, we calculate the peak load profile for each year from the beginning year t_0 until the final year of the planning period $t_{horizon}$. We can determine the year t_X when the first bottleneck happens in the current network, which is also the year that the first reinforcement activity is needed. At the end of the planning period $t_{horizon}$, all reinforcement activities in a solution plan will have been carried out, resulting in the final network configuration.

For the static DNEP problem, in which the time of reinforcement is disregarded, we assume that all reinforcement activities in the solution plan are implemented at the same time in year t_X. Any $\boldsymbol{x} \neq \boldsymbol{x}^0$ can be seen as a candidate expansion plan. The element-wise differences between $\boldsymbol{x}^0$ and $\boldsymbol{x}$ indicate which reinforcement activities need to be carried out to transform the current network $\boldsymbol{x}^0$ into the final network $\boldsymbol{x}$ (the new assets typically have equal or greater capacities than the old ones). It is of interest to find the most economical $\boldsymbol{x}$ that satisfies the peak power demand at the final year of the planning period $t_{horizon}$.

4.2.2. Constraints

The following constraints must be satisfied for any candidate network to be considered feasible:

1. **Connectivity**: For each consuming node (i.e., an MV/LV network substation or MV customer substation), there exists a path of concatenating cables connecting that consuming node to an HV/MV supplying substation.
2. **Power flow constraint**: During normal operation, the voltage at each node is within allowable ranges (i.e., $0.9 * V_i^{nom} \leq V_i \leq 1.1 * V_i^{nom},\ i \in \{1, 2, \ldots, n\}$) and the magnitude of the power flow through each cable is within the nominal capacity of the that cable (i.e., $|S_i| \leq S_i^{nom},\ i \in \{1, 2, \ldots, l\}$).
3. **Radiality constraint**: In normal operation, the power demand of each node is supplied by a single feed path.
4. **Reconfigurability constraint**: When an active cable ($x_k > 0$) fails, the part of the feeder group (from an HV/MV substation with circuit breaker to an NOP) containing the failed cable is disconnected from the network. All customers connected to that feeder group are then out of service. The DNO has to bring the network back to operation by closing NOPs to temporarily re-route the power flow through other paths while the failed cable is being repaired. During this emergency situation, the radial operation constraint can be compromised and the network is allowed to endure a mild overload of 130% nominal capacity (as in [2]). Figure 4.2 shows an example of the network reconfigurability.
5. **Substation capacity constraint**: Each HV/MV substation has limited physical space to install new outgoing cables. We assume that at most three new outgoing cables are allowed for each HV/MV substation (as in [2]).

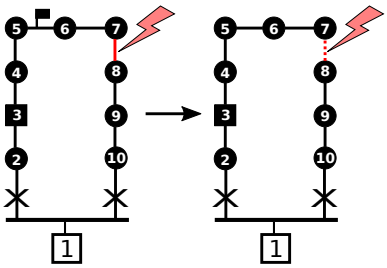

Figure 4.2: An example of network reconfigurability. Cable 7-8 fails and is isolated for repair. The NOP on cable 5-6 is then temporarily closed so that MV nodes 6 and 7 can be re-supplied through a different path.

Constraints 1, 3, and 5 can be verified by checking the topology of the network in each stage. Alternating-current (AC) power flow calculations (PFCs) [17] are required to check constraints 2 and 4 for each solution plan. PFCs, which involve solving AC power flow models, are computationally expensive and dominate the computing time of the optimization process. Note that the simpler DC model is not accurate enough for PFCs in distribution networks since the condition that branch resistance is negligible compared to reactance ([18]) does not hold for distribution networks. However, a complete verification of the reconfigurability constraint (i.e.,

constraint 4), that requires an AC PFC to calculate the power flow for each failure of an active cable in the network, is too cumbersome. Here, we employ the Line Outage Distribution Factor (LODF [19]) method to verify this constraint. LODFs provide rapid assessment of multiple branch outage impacts and require only one (pre-contingency) PFC. LODFs in combination with an AC power flow were shown to require much less computing time while having an acceptable accuracy for the capacity evaluation of MV-D networks [19].

Note that PFCs can only be performed for connected networks. The connectivity constraint is a crucial constraint because constraints 2 and 4 can only be evaluated if the candidate network is connected. Therefore, if the network encoded in a solution plan is unconnected, we do not evaluate other constraints but we quantify its disconnectivity by comparing it with the topology of the existing network $\boldsymbol{x}^0$. Specifically, we loop through all the decision variables and count the number of positions where the existing network has a positive value (i.e., an active branch) but the solution has a negative value (i.e., an NOP), or where the existing network has a non-positive value (i.e., no cable connection or an NOP) and the solution has a positive value. This number is considered to be the disconnectivity value. Note that the case when a position in the existing network has a positive value and that position in the candidate solution has value 0 does not exist because such a case implies that an existing cable connection would be removed, which is generally undesirable according to network operators (see more details in Section 4.4). Connected networks do not need this disconnectivity quantification and are assigned the disconnectivity value 0. Intuitively, the disconnectivity quantification procedure measures, in terms of cable connections, how an unconnected network differs from the currently existing network (which has a connected topology). Disconnectivity values can be employed when two unconnected networks need to be compared during the optimization process because the other constraints cannot be evaluated. Figure 4.3 shows the pseudo-code for the disconnectivity quantification procedure.

```
DisconnectivityQuantification(x)
1 if CheckConnectivity(x) then
2     Return( 0 )
3 count ← 0
4 for k ∈ {1, 2, ..., l} do
5     if [x_k^0 > 0 and x_k < 0] or [x_k^0 ≤ 0 and x_k > 0] then
6         count ← count + 1
7 Return(count)
```

Figure 4.3: Disconnnectivity Quantification

To deal with solutions that violate constraints, we will, as a basis, consider the use of constraint domination [20]. Because we have a cascade of importance in the constraints, we modify constraint domination slightly to work as follows. When comparing two networks, if both networks are unconnected, the one with a smaller disconnectivity value is the better one. If only one network is unconnected, then the connected network is the better solution. If both networks are connected, they

can then be compared by using the other evaluated constraint values and their objective values. The network with smaller total constraint violation (constraints 2,3,4,5) is the better one. If both candidate networks are feasible (i.e., no constraint violation), the one having smaller cost is preferred. Figure 4.4 shows the pseudo-code for our implementation of constraint domination in DNEP. The total constraint violation of a candidate network is taken as the aggregate of the amounts of violation of constraints 2, 3, 4, and 5. The radiality constraint and the reconfigurability constraint are evaluated as Boolean values, i.e., 1-value indicates the constraint is satisfied while 0-value indicates the constraint is violated. More refined methods to quantify DNEP constraint violations are worth further investigation but are outside the scope of this work.

4.2.3. Objective function

We employ the annuity method [21] to calculate the capital expenditure CAPEX for new assets. The investment cost on a new cable is converted into a series of uniform annual payments, called annuities. We assume the length of this series to be equal to the (uniform) economic lifetime of the new asset t_{life}. Since electric cables are considered to be the main asset category in this thesis, we can also assume that the economic lifetime of all new assets is $t_{life} = 30$ years. The annuity of a cable c with the discount rate $i = 4.5\%$ (as in [2]) can be computed as:

$$Annuity_(c) = Price(c) \times \frac{i}{1 - (1+i)^{-t_{life}}} \tag{4.2}$$

where $Price(c)$ is the acquisition cost for the new cable c, which depends on the cable type and the length of that connection. CAPEX for cable c in year t can be calculated as:

$$CAPEX_{cable}(c,t) = \begin{cases} Annuity(c) & \text{if } t_{inst_c} \leq t < t_{inst_c} + t_{life} \\ 0 & \text{else} \end{cases} \tag{4.3}$$

with t_{inst_c} is the time of installing the cable c. Note that for static DNEP, $t_{inst_c} = t_X$, the year when the first bottleneck happens in the current network. Let C denote the set of all new cables c's that will be installed in the planning period $[t_0, t_{horizon}]$. The total CAPEX on the whole network in year t is defined as:

$$CAPEX(t) = \sum_{c \in C} CAPEX_{cable}(c,t) \tag{4.4}$$

We want to minimize the net present value (NPV) (i.e., at time t_0) of the investment cost over the planning period $[t_0, t_{horizon}]$ with a discount rate i.

$$CAPEX_{NPV} = \sum_{t=t_0}^{t_{horizon}} \frac{CAPEX(t)}{(1+i)^{t-t_0}} \tag{4.5}$$

The energy losses can be taken into account in the objective function as the operational expenditure OPEX. The energy loss of the network in year t can be

```
IsBetter(x, x')
 1 dq  ← DisconnectivityQuantification(x)
 2 dq' ← DisconnectivityQuantification(x')
 3 if dq < dq' then
 4     Return (TRUE)
 5 else if (dq > dq' ≥ 0) or (dq = dq' > 0) then
 6     Return (FALSE)
 7 else
 8     con  ← ConstraintViolationCalculation(x)
 9     con' ← ConstraintViolationCalculation(x')
10     if con < con' then
11         Return (TRUE)
12     else if (con > con' ≥ 0) or (con = con' > 0) then
13         Return (FALSE)
14     else
15         cost  ← TotalCostCalculation(x)
16         cost' ← TotalCostCalculation(x')
17         if cost < cost' then
18             Return (TRUE)
19         else
20             Return (FALSE)
```

```
ConstraintViolationCalculation(x)
// n: number of nodes; l: number of branches
// V = (V_1, V_2, ..., V_n): vector of voltage at each node
// S = (S_1, S_2, ..., S_l): vector of power flow through each branch
 1 con ← 0
 2 V, S ← PowerFlowCalculation(x)
 3 for i ∈ {1, 2, ..., n} do
 4     con ← con + max(V_i^min − V_i, V_i − V_i^max, 0)
 5 for i ∈ {1, 2, ..., l} do
 6     con ← con + max(S_i − S_i^nom, 0)
 7 if con > 0 then
 8     con ← con + 1
 9 else
10     con ← BooleanToInt(¬ReconfigurabilityCheck(x))
11 con ← con + BooleanToInt(¬RadialityCheck(x))
12 for i ∈ {1, 2, ..., n} do
13     con ← con + max(NumberOfOutgoingCables(i) − 3, 0)
14 Return(con)
```

Figure 4.4: Constraint Domination for DNEP

estimated as in [2, 21, 22]:

$$E_{loss}(t) = P_{peak\ loss}(t) \times T_{loss}(t) \tag{4.6}$$

where $P_{peak\ loss}(t)$ is the peak loss which can be obtained from the PFC regarding the peak loads in year t. $T_{loss}(t)$ is the service time of peak loss for year t, defined by the area of the yearly loss profile [2, 21]. We here assume that $T_{loss}(t) = 2000$

hours, which is a typically reasonable value for MV distribution networks in the Netherlands [22]. Given the forecast electricity price in year t, we can capitalize the energy loss and regard it as the OPEX in year t.

$$OPEX(t) = E_{loss}(t) * Price_{electricity}(t) \tag{4.7}$$

In this thesis, we take the price of electricity for energy loss capitalization as 0.068 EUR/kWh during the planning period. We want to minimize the net present value (NPV) (i.e., at time t_0) of the total cost of both investment cost CAPEX and operation cost OPEX during $[t_0, t_{horizon}]$ with a discount rate i.

$$COST_{NPV} = \sum_{t=t_0}^{t_{horizon}} \frac{CAPEX(t) + OPEX(t)}{(1+i)^{t-t_0}} \tag{4.8}$$

The goal in static DNEP is to find the most economical solution plan that satisfies the peak load profile at the final year of the planning period. While omitting the installation time, static DNEP can still give DNOs a general picture about what kinds of network reinforcements can be expected. However, the asset installation time is required to calculate the NPV of the investment cost (see Equation 4.3). Also, the operation cost OPEX, which is the capitalized energy loss, depends on the peak load and the specific network configuration in each year (see Equation 4.6). We employ the following method, which has been proposed in [2], to determine the earliest possible installation time. Based on the (predicted) annual peak load growth rate R, we compute the peak load profile for each year from the beginning year t_0 until the final year $t_{horizon}$. We determine the year t_X when the first bottleneck happens in the network. We then assume that all expansion options in the solution plan are installed at the same time in year t_X, i.e., $t_{inst} = t_X$ for all new assets. Therefore, to evaluate the total cost, from t_0 until t_X, we use the current network topology, and from t_X until $t_{horizon}$ we use the new network topology.

Note that, due to the long economic lifetime of electrical assets (i.e., 30 years in this work), annuity payments can exceed the planning horizon $t_{horizon}$. However, because we focus on the planning period, only the costs incurred during $[t_0, t_{horizon}]$ are taken into account (as in Equations 4.5 and 4.8). Figure 4.5 illustrates the concept.

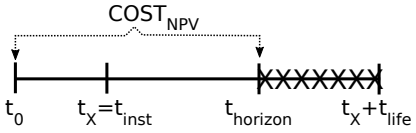

Figure 4.5: Net Present Value (NPV) of the total cost.

4.3. Experiment setup

4.3.1. Benchmark problems

We perform experiments on three benchmark networks of different sizes that represent real distribution networks of DNO Enexis in The Netherlands. Figure 4.6

shows the current topologies of the three networks. More details about current power demands, potential locations for installing new cables, and characteristics of different cables types can be found in the Appendix. The overall sizes of these three benchmarks are summarized in Table 4.1. The planning period consists of 30 years.

ID	# Branches (Variables)	# Nodes	# HV/MV Substations	# Cable types
1	17	10	1	3
2	59	31	1	3
3	190	51	4	12

Table 4.1: Benchmark Network Size

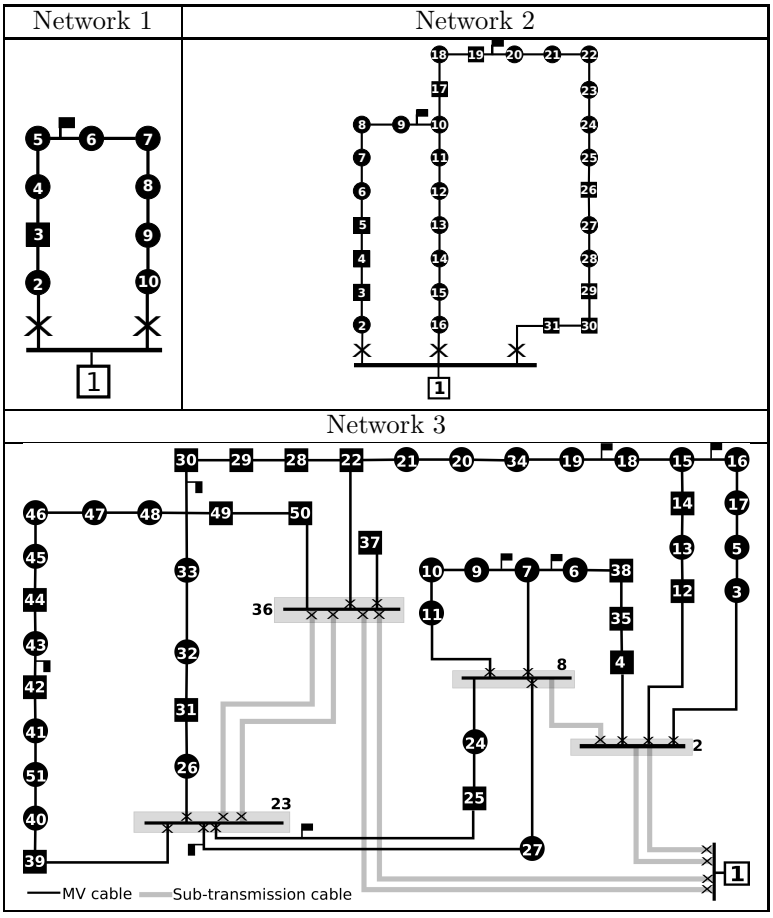

Figure 4.6: Three benchmark MV distribution networks with existing assets. Potential cable connections can be found in the Appendix. Legends are explained in Figure 4.1.

4.3.2. Optimization algorithms

We use evolutionary algorithms (EAs) to search for optimal expansion plans for the three benchmark networks presented above, i.e., by optimizing Equation 4.8 as a function of the variables in Equation 4.1, subject to the constraints in Section 4.2.2. To study the impact of different types of EA and different linkage models, we combine three linkage/probabilistic models (i.e., univariate factorization (UF), marginal product (MP), and linkage tree (LT)) with three EAs (i.e., GA, EDA, and GOMEA) as described in Chapter 2 to create seven EA solvers: GA-UF, GA-MP, EDA-UF, EDA-MP, GOMEA-UF, GOMEA-MP, and GOMEA-LT. Since all these seven EA solvers have been made parameter-less by employing the Harik-Lobo population-sizing-free scheme (see Section 2.4), we don't need to set the population size parameter for any solvers. For every benchmark network, each solver is run 30 times independently. In each run, the computing budget is given as the maximum number of evaluations, which is 50,000 for Network 1, 100,000 for Network 2, and 1,000,000 for Network 3. Better solutions might be found by spending more computing time, but regarding the purpose of demonstration, these computational budgets are reasonably adequate. The average convergence graphs of the elitist solutions obtained during the optimization process from the beginning until termination are used as a basis for assessing the performance of each solver. To support our conclusions from the experimental results, we perform statistical hypothesis testing. In particular, we perform the Mann-Whitney-Wilcoxon statistical hypothesis test for equality of medians with $p < 0.05$ to see whether the final result obtained by one EA is statistically different from that of another EA.

4.4. Problem-specific population initialization

Normally, EAs can start with randomly initialized populations. However, because DNEP is a highly-constrained engineering problem, randomly generated solutions are typically infeasible and violate many constraints. Therefore, we use a repair procedure that partially repairs infeasible solutions by comparing them with the current, i.e., starting, network situation. First, each decision variable x_k (i.e., a network branch) can only receive a random non-negative value (i.e., we do not place any NOPs yet) as long as it does not downgrade the currently existing cable. This also means that currently existing cables can only be left intact or be replaced with cables of higher capacities. Existing connections are rarely removed because cable connection removals decrease network capacity. Solutions that are generated in this way satisfy the connectivity constraint because the current network is connected. Second, we go through HV/MV substations and check the number of cables branching out from each HV/MV substation. If the number of outgoing cables is more than the allowable capacity of the substation (i.e., violating constraint 5), we randomly delete outgoing cables until constraint 5 is satisfied. Third, we go through all variables that have positive values (i.e., active cables) in a random order. For each positive-value decision variable, we try to place an NOP on that cable by negating its value. If the network is still connected, then the NOP can be placed; otherwise, we undo the operation. This procedure returns a network of radial topology with random placements of NOPs (i.e., constraint 3 is satisfied). We do not repair the

power flow and reconfigurability constraints (i.e., constraints 2 and 4) because they involve PFCs, which are computationally expensive. Figure 4.7 shows pseudo-code for randomly generating a network solution, which can be used in the population initialization phase.

```
DNEP::CreateRandomSolution()
// 𝕋 = set of all cable types (including 0).
// x = (x_1, x_2, ..., x_l), |x_k| ∈ Ω(k), k ∈ {1, 2, ..., l}
 1 x^0 ← GetCurrentNetworkConfiguration()
 2 for k ∈ {1, 2, ..., l} do
 3     Ω(k) ← {ID ∈ 𝕋 | capacity[ID] ≥ capacity[|x_k^0|]}
 4     x_k ← Random(Ω(k))
 5 x ← RemoveRedundantConnections(x, x^0)
 6 x ← RadializeNetwork(x)
 7 Return(x)
```

```
RemoveRedundantConnections(x, x^0)
// ℍ = set of all HV/MV substations.
 1 o ← x
 2 for s ∈ ℍ do
 3     C_s = {}
 4 for k ∈ {1, 2, ..., l} do
 5     if IsOutgoingCable(k) and x_k ≠ 0 and x_k^0 = 0 then
 6         s ← GetSubstationIndexOfCable(k)
 7         C_s ← C_s ∪ {k}
 8 for s ∈ ℍ do
 9     if |C_s| > max_s then
10         π ← RandomPermutation({1, 2, ..., |C_s|})
11         for i ∈ {1, 2, ..., |C_s| − max_s} do
12             o_{π_i} ← 0
13 Return(o)
```

```
RadializeNetwork(x)
 1 o ← x
 2 π ← RandomPermutation({1, 2, ..., l})
 3 for i ∈ {1, 2, ..., l} do
 4     if o_{π_i} > 0 then
 5         o_{π_i} ← −o_{π_i}
 6         if ¬CheckConnectivity(o) then
 7             o_{π_i} ← x_{π_i}
 8 Return(o)
```

Figure 4.7: Generating a distribution network configuration.

Figure 4.8 shows the influence of the above-mentioned problem-specific initialization on the performance of seven different EA variants when solving DNEP for Network 1. Figure 4.8 shows that, regardless of the optimization algorithms or the employed linkage (or probabilistic) models, the problem-specific initialization always brings about better convergence performance compared to random initial-

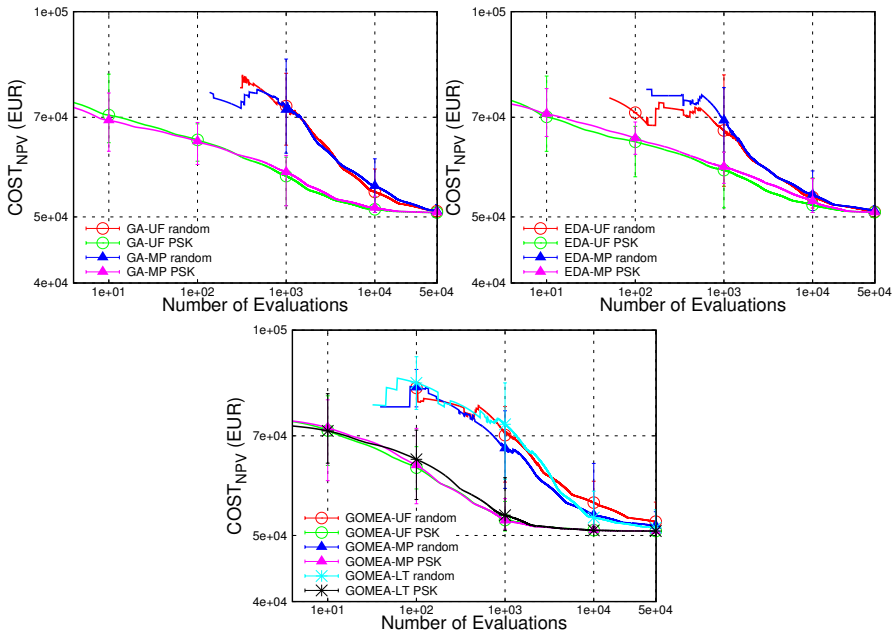

Figure 4.8: Benchmarking the performance of GA (top left), EDA (top right), and GOMEA (bottom) solving DNEP for Network 1 when employing the random initialization or a problem-specific initialization.

ization. Because it is difficult for a fully random initialization to generate connected (and radial) network topologies, starting from randomly-initialized populations, EAs need more exploration to construct feasible topologies before the total cost objective function can be minimized. This can be seen from Figure 4.8 because the convergence graphs only show objective values for feasible solutions. For randomly-initialized EAs these graphs start at a later time (i.e., in terms of the number of evaluations) compared to EAs with problem-specific initialization. On the other hand, when being given initial populations of good candidate networks, which are connected, radial and within the substation capacity, it is easier for EAs to locate feasible regions in the search space, and then, the optimal solutions can be found by using fewer number of evaluations. Because the DNEP-specific initialization scheme (as presented in Figure 4.7) is beneficial to the performance of all solvers, it is used in all the following DNEP experiments in this thesis.

4.5. Adaptations of the Harik-Lobo scheme and linkage model selection for EAs solving DNEP

4.5.1. Adaptations of the Harik-Lobo scheme

For reasons of efficiency, the Harik-Lobo scheme terminates old populations of smaller sizes when they converge or when they are shown to be inefficient in solving

the problem instance at hand. If mutation is not employed, a converged population, in which all individuals become identical, should be terminated because new offspring cannot be generated any more. A smaller population P_i always spends the same number of fitness evaluations (in case $b = 2$) or more (in case $b > 2$) than a larger population P_{i+1}. Therefore, a smaller population is regarded as inefficient if its average fitness is worse than the average fitness values of any larger population. Such smaller and inefficient populations should be terminated as well. However, the Harik-Lobo scheme has only been tested on unconstrained problems previously. For constrained optimization problems like DNEP, it is difficult to compute and compare average fitness values of different populations, especially when a population has a mixture of both feasible and infeasible solutions. Therefore, we will benchmark two variants of the Harik-Lobo scheme: the original scheme with the termination of converged *or* inefficient (smaller) populations and the adapted scheme with the termination of only converged populations.

We employ Network 3 (see Section 4.3 and the Appendix for more details) as the benchmark network. We combine different linkage/probabilistic models with different EAs to create seven solvers: GA-UF, GA-MP, EDA-UF, EDA-MP, GOMEA-UF, GOMEA-MP, and GOMEA-LT (see Chapter 2). Each EA variant is run 30 times separately with a computing budget of 1,000,000 evaluations in each run. Each EA variant is put into the two variants of the Harik-Lobo scheme (both use the generation base $b = 4$ as in the original implementation of the Harik-Lobo scheme [13]) so that setting the population size parameter is not required. The average results of the elitist solutions over 30 runs are shown in Figure 4.9.

Figure 4.9 shows that, for all GOMEA variants, there is no statistically significant difference in performance of the original Harik-Lobo scheme and the adapted scheme. On the other hand, for GAs and EDAs, regardless of the employed FOS models, the variants with the adapted scheme always obtain better solutions within the same computing budget. These differences are found to be statistically significant. Because DNEP is a constrained optimization problem, comparing average fitness values of populations containing both feasible and infeasible solutions might not be informative to determine whether a population is inefficient in solving the problem instance at hand. Besides, Figure 4.9 also indicates that, compared to GA and EDA, GOMEA is the most consistent algorithm, obtaining the same results, regardless of which population-sizing-free scheme is chosen. The experiment suggests that the adapted Harik-Lobo scheme is the better population-sizing-free framework for EAs solving our DNEP formulation. We will use the adapted Harik-Lobo scheme in all following experiments.

The generation bases $b = 2$ and $b = 4$ are the two most widely-used bases in the literature of the Harik-Lobo population-sizing-free scheme [13–15]. Here, we benchmark the adapted Harik-Lobo scheme with both generation bases 2 and 4 to understand which is the better setting for DNEP. We also conduct experiments with seven EA variants on Network 3 as outlined above. Figure 4.10 shows the results. The difference between $b = 2$ and $b = 4$ is negligible when GOMEA is the employed optimizer together with the UF model or the MP model. For GOMEA-LT, the result obtained when $b = 4$ is a little better than the result obtained when $b = 2$, and this

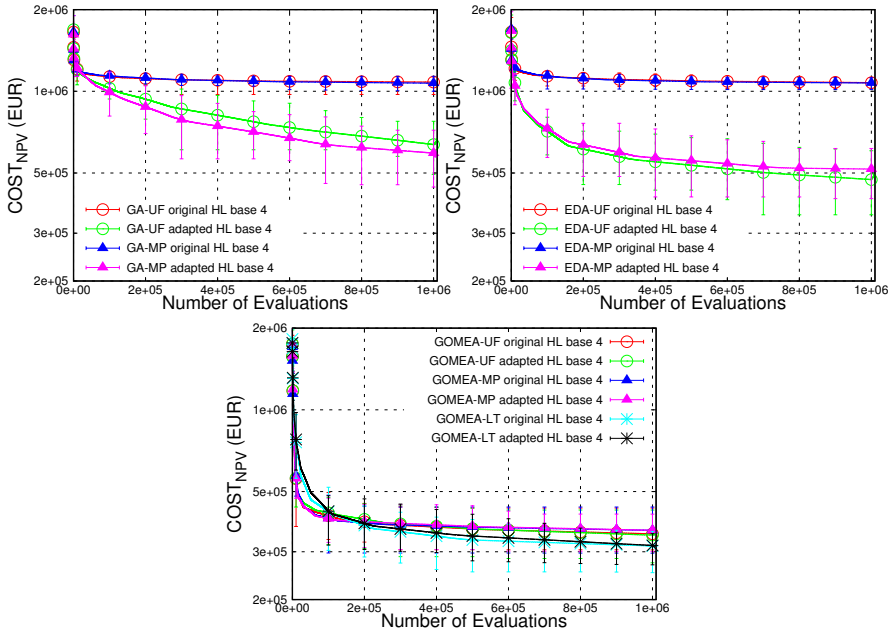

Figure 4.9: Benchmarking the original Harik-Lobo population-sizing-free scheme and the adapted scheme on solving DNEP for Network 3 by GA (top left), EDA (top right), and GOMEA (bottom).

difference is found to be statistically significant. For GAs and EDAs, the Harik-Lobo scheme always has better performance when $b = 4$ than when $b = 2$ regardless of the employed FOS model, and the differences are statistically significant. Therefore, for all following experiments, we employ the adapted Harik-Lobo scheme with the generation base $b = 4$ as the population-sizing-free framework for EAs solving our DNEP formulation.

4.5.2. Linkage model selection

Figure 4.11 shows that all GOMEA variants, regardless of the chosen linkage models, always obtain solutions of (statistically significantly) better quality than those found by all GA and EDA variants within the allowed computing budget. This confirms that the laboratory-benchmarked superior performance of the Gene-pool Optimal Mixing operator is retained when solving the real-world optimization DNEP. GOMEA-UF and GOMEA-MP show no difference in their convergence behavior. The big cardinality of each variable (i.e., much more than 2) causes the complexity term in Equation 10 to be large and thus prohibits the merging of small linkage sets, especially when population sizes are small. The learned MP FOS therefore contains mostly univariate linkage sets, which are similar to the UF model. Figure 4.11 shows that GA-MP performs slightly better than GA-UF while EDA-UF performs slightly better than EDA-MP, and these differences are found to be statistically significant. The fact that EDA-UF outperforms EDA-MP, GA-UF, and GA-MP suggests that

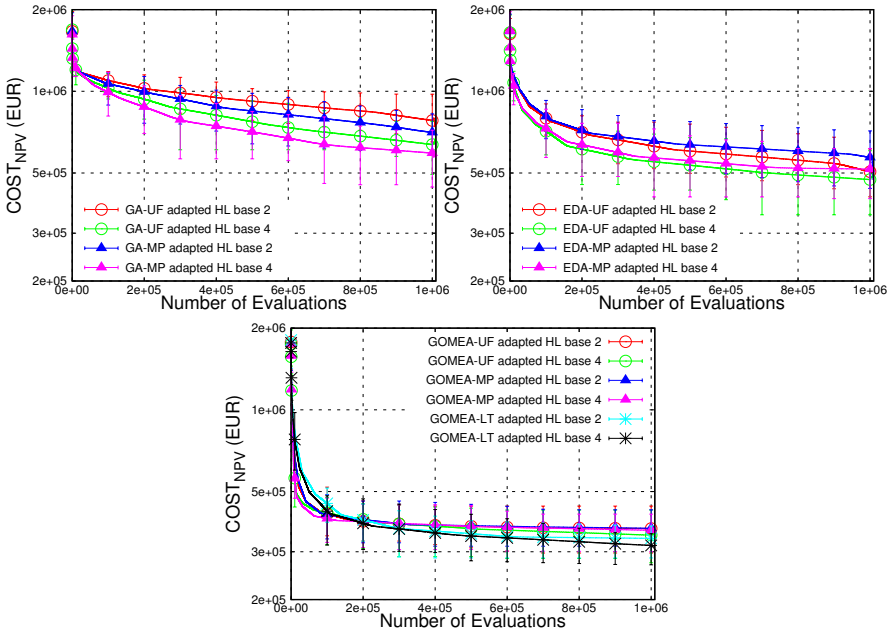

Figure 4.10: Benchmarking the adapted population-sizing-free scheme with two generation bases $b = 2$ and 4 on solving DNEP for Network 3 by GA (top left), EDA (top right), and GOMEA (bottom).

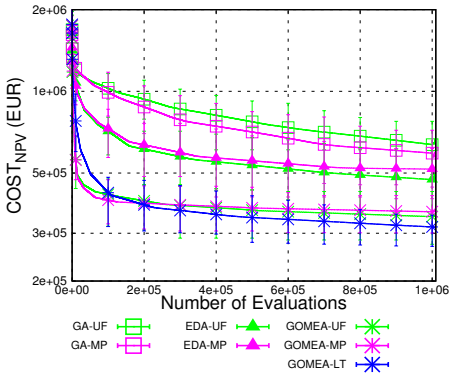

Figure 4.11: Performance of GA, EDA, and GOMEA when combined with the UF, MP, and LT models on solving DNEP for Network 3. Error bars show the maximum and minimum values of the Net Present Value of total cost. The adapted Harik-Lobo population-sizing-free scheme is employed with the generation base $b = 4$ for all EA variants.

the objective function of our DNEP formulation resembles the function OneMax to some degree, i.e., CAPEX is calculated as the sum of the investment cost of each new asset and OPEX is computed as the sum of the capitalized energy loss of every

asset. However, the DNEP constraints certainly cause dependencies to exist among problem variables, e.g., the network cables must form a connected graph such that all demand nodes can be served (the connectivity constraint), or there must exist enough redundancies in the network so that it can be reconfigured when some failure occurs (the reconfigurability constraint). These constraints are difficult and numerous. Thus, it stands to reason that their imposition on the dependency structures of the feasible space is substantial. However, as said, the marginal product model can only capture dependencies to a very limited degree, and therefore its added value, if any, compared to the univariate model, is also limited. The interactions between problem constraints, objective function values, and the model building procedure are worth further research but are outside the scope of this study.

Figure 4.11 shows that GOMEA-LT outperforms all other EA variants, and the results are found to be statistically significant. We argue that the capability of the LT to model different dependency levels at once, makes the LT much better suited for the DNEP structure. For example, upgrading a cable in one region does not affect the power flows through other cables in a different section of the distribution network (i.e., independence in terms of power flows in normal situation), but those separate cables can be reconfigured to be connected when a network failure happens (i.e., inter-dependence in terms of reconfigurability).

Selecting the best FOS structure for each EA, in all following experiments, we employ the MP model for GA, the UF model for EDA, and the LT model for GOMEA.

4.6. Adaptations of variation operators

Being popularly applied in black-box optimization, EAs require little problem-specific knowledge (PSK) and their variation operators (VOs) (i.e., procedures to generate new offspring solutions) can operate on a wide range of problems. However, in real-world applications like DNEP, the problems are often highly constrained such that it is difficult for general-purpose VOs to traverse the search space of feasible solutions efficiently. Efficiently traversing the space of feasible solutions is of high importance because the evaluations of candidate solutions are typically computationally expensive. Full constraint evaluations of solution plans for DNEP involve solving PFCs [17], which dominates the computing time of the optimization process. Thus, it is beneficial to incorporate PSK into VOs of EAs as efficiency enhancement methods. Local search heuristics can normally be used for efficiency enhancement, but a typical operator like hillclimbing with some small neighborhood (e.g., 1/2/3-opt local search) was not found to be helpful in improving the efficiency of EAs solving our DNEP model because such local search often fails to reach expansion options that include inter-dependent activities (e.g., adding new cables and NOPs and relocating existing NOPs).

Among the constraints of DNEP, the connectivity constraint needs to be handled separately because PFCs cannot be performed on unconnected networks and the power flow constraint and reconfigurability constraint cannot be checked without PFCs. Thus, in order to compare solution plans (e.g., when performing tournament selection), we need to quantify their disconnectivity and use that as the comparison

criterion or we have to repair the connectivity so that other constraints can be evaluated. Here, we introduce different VOs for GA, EDA, and GOMEA along with their corresponding connectivity constraint-handling techniques.

4.6.1. Disconnectivity quantification

A simple VO for EAs when solving our DNEP problem is to directly employ the disconnectivity quantification procedure (see Section 4.2.2 and pseudo-code in Figure 4.3). At every recombination, model sampling, or mixing event, a new offspring solution (or partially-new solution in case of GOMEA) is created and is evaluated for its fitness value. The connectivity constraint is checked first, and only a connected network solution is then evaluated for other constraints and objective value. Candidate solutions can be compared by the connectivity-constraint domination mechanism as presented in Section 4.2.2. We name this VO DQ1 as the solution variation is performed only one time (compared with the VO DQ100 in the following paragraph). DQ1 can be considered as an out-of-the-box VO of EAs because it simply employs the disconnectivity qualification procedure, which is part of the constraint evaluation itself. Apart from that, no other connectivity knowledge is assumed when generating offspring solutions.

However, even if we recombine two connected parent networks, it is still difficult for the crossover operator of GA or the model sampling of EDA to generate connected offspring networks, especially for big networks with many cable connections. Thus, we also propose a different VO, in which we allow each recombination of two parent networks to retry crossover maximum 100 times to generate connected offspring networks. Similarly, for EDA, we allow maximum 100 times of model sampling to generate a connected networks before performing disconnectivity quantification. For GOMEA, during the process of constructing an offspring, for each mixing event, if the partially-altered solution is an unconnected network, we allow it to randomly select a different donor for a maximum of 100 times. After 100 times of solution variation (i.e., crossover in GA, model sampling in EDA, and mixing in GOMEA), if the offspring networks are still unconnected, they will be evaluated for the disconnectivity value with the disconnectivity quantification procedure as in DQ1. We call this VO DQ100 as the solution variation is allowed 100 trials each time an offspring solution is created (or a partially-altered solution in the case of GOMEA). Figure 4.12 shows the pseudo-code for DQ100 with its corresponding realization in GA, EDA and GOMEA.

4.6.2. Connectivity repair

Inspired by the Forced Improvement (FI) operator of GOMEA, in which solutions that cannot be altered by GOM will be mixed with the elitist solution $\boldsymbol{x}^{elitist}$, we propose a repair procedure to fix any unconnected network $\boldsymbol{x}$ by matching it with the (overall) best-found-so-far solution $\boldsymbol{x}^{best} = \boldsymbol{x}^{elitist}$. For each decision variable k, if $x_k^{best} > 0$ and $x_k < 0$, then $x_k \leftarrow -x_k$; if $x_k^{best} > 0$ and $x_k = 0$, then $x_k \leftarrow x_k^{best}$. On the other hand, if $x_k^{best} < 0$ and $x_k > 0$, then $x_k \leftarrow -x_k$; if $x_k^{best} = 0$ and $x_k > 0$, then $x_k \leftarrow 0$. In other words, we try to transform the topology of the unconnected network to the best solution's topology. We call the

```
GA::Recombine::DQ100(p^0, p^1)
1 o ← Recombine(p^0, p^1); count ← 1
2 while ¬CheckConnectivity(o) and count < 100 do
3     o ← Recombine(p^0, p^1)
4     count ← count + 1
5 Return(o)
```

```
EDA::SampleDistribution::DQ100()
1 o ← SampleDistribution(); count ← 1
2 while ¬CheckConnectivity(o) and count < 100 do
3     o ← SampleDistribution()
4     count ← count + 1
5 Return(o)
```

```
GOMEA::CopyValues::DQ100(x, d, F^i)
1 o ← CopyValues(x, d, F^i); count ← 1
2 while ¬CheckConnectivity(o) and count < 100 do
3     d' ← Random({P_1, P_2, ..., P_n})
4     o ← CopyValues(x, d', F^i)
5     count ← count + 1
6 Return(o)
```

Figure 4.12: Disconnnectivity Quantification DQ100

VO that uses this connectivity repair procedure CRB (i.e., Connectivty Repair by the Best solution). Figure 4.13 shows the pseudo-code for the CRB scheme. CRB can be straightforwardly employed by any EA.

```
ConnectivityRepairByBestSolution(x)
 1 x^best ← GetCurrentBestNetworkConfiguration()
 2 o ← x
 3 for k ∈ {1, 2, ..., l} do
 4     if x_k^best > 0 and x_k < 0 then
 5         o_k ← −x_k
 6     else if x_k^best > 0 and x_k = 0 then
 7         o_k ← x_k^best
 8     else if x_k^best < 0 and x_k > 0 then
 9         o_k ← −x_k
10     else if x_k^best = 0 and x_k > 0 then
11         o_k ← 0
12 Return(o)
```

Figure 4.13: Connectivity Repair by the Best Solution.

We propose a second connectivity repair mechanism for unconnected offspring networks by using the parent solution network. For GA, this VO is much like DQ100, but after 100 trials, if the networks are still unconnected they will be reverted back to the parent solutions. Because all the candidate solutions in the initial population are

connected networks (due to our solution network generator as presented in Section 4.4), the use of this VO implies that only connected offspring solutions are allowed to be evaluated and enter tournament selection. For GOMEA, in each mixing event, if the partially-altered solution becomes unconnected, this is because there exist some variables whose positive values are replaced by some non-positive values from the donor. We can simply revert these decision variables to their backup values. We call the VO that uses this connectivity repair scheme CRP (i.e., Connectivity Repair by Parent). Note that we do not combine EDA with CRP because offspring solutions in EDA do not have direct parent solutions like GA or GOMEA, but are generated by sampling the learned probability distribution. Figure 4.14 shows the pseudo-code for the realization of the CRP scheme in GA and GOMEA.

```
GA::ConnectivityRepairByParent(p^0, p^1)
1 o ← GA::Recombine:DQ100(p^0, p^1)
2 if ¬CheckConnectivity(o) then
3     o ← p^0
4 Return(o)
```

```
GOMEA::CopyValues::ConnectivityRepairByParent(x, d, F^i)
1 o ← GOMEA::CopyValues(x, d, F^i)
2 if ¬CheckConnectivity(o) then
3     for k ∈ F^i do
4         if x_k > 0 and o_k ≤ 0 then
5             o_k ← x_k
6 Return(o)
```

Figure 4.14: Connectivity Repair by the Best Solution.

4.6.3. Branch exchanging

This VO aims to directly generate connected offspring networks by following the principle that during the recombination of two connected networks $\boldsymbol{p}^0$ and $\boldsymbol{p}^1$, if we bring a cable connection from $\boldsymbol{p}^1$ to $\boldsymbol{p}^0$ (i.e., $p^0_k \leftarrow p^1_k, p^0_k \leq 0, p^1_k > 0$), we need to bring the corresponding NOP (or a no-connection branch) from $\boldsymbol{p}^1$ to $\boldsymbol{p}^0$ as well (i.e., $p^0_j \leftarrow p^1_j, p^0_j > 0, p^1_j \leq 0$) and vice versa.

For GA, during the recombination of two parents $\boldsymbol{p}^0$ and $\boldsymbol{p}^1$, when we copy values from $\boldsymbol{p}^1$ according to a linkage set $\boldsymbol{F}^i$ in the FOS $\mathcal{F}$ (see Figure 2.2), for variables whose values are both positive (i.e., both are active cables) or both non-positive (i.e., no-connection branches or NOPs), we can copy as normal since these positions have the same structures in both networks. For each variable index k where $p^0_k \leq 0$ and $p^1_k > 0$ (or $p^0_k > 0$ and $p^1_k \leq 0$), we need to find a different variable j where $p^0_j > 0$ and $p^1_j \leq 0$ (or $p^0_j \leq 0$ and $p^1_j > 0$) such that copying values at variables k and j from the network $\boldsymbol{p}^1$ still maintains the connectivity of the network $\boldsymbol{p}^0$. If we cannot find such a variable j then we do not perform crossover at variable index k. Note that while variable k belongs to the current linkage set $\boldsymbol{F}^i$, variable j is searched for over the whole solution. Because the complicated connectivity

```
CopyValues::BranchExchange(x, d, F^i)
 1 𝕏 ← {k ∈ {1,2,...,l} | (x_k > 0 and d_k > 0) or (x_k ≤ 0 and d_k ≤ 0)}
 2 𝕏' ← {1,2,...,l} \ 𝕏
 3 𝕐 ← {k ∈ F^i | (x_k > 0 and d_k > 0) or (x_k ≤ 0 and d_k ≤ 0)}
 4 𝕐' ← F^i \ 𝕐
 5 o ← x
 6 for k ∈ 𝕐 do
 7     o_k ← d_k
 8 for k ∈ 𝕐' do
 9     for j ∈ 𝕏' \ {k} in a random order do
10         if (o_k ≤ 0 and d_j ≤ 0) or (o_k > 0 and d_j > 0) then
11             o_k ← d_k; o_j ← d_j
12             if CheckConnectivity(o) then
13                 𝕏' ← 𝕏' \ {k,j}; 𝕐' ← 𝕐' \ {k,j}
14                 break for
15             else
16                 o_k ← x_k; o_j ← x_j
17 Return(o)
```

Figure 4.15: Branch Exchange.

structure might not be entirely captured by linkage learning, an active cable and its corresponding NOP might not always reside in the same linkage set $\boldsymbol{F}^i$.

For GOMEA, in each mixing event, this procedure works similarly for the current solution $\boldsymbol{o}$ and a donor $\boldsymbol{d}$ and we need to search for the variable j when $o_k \leq 0$ and $d_k > 0$ (or $o_k > 0$ and $d_k \leq 0$). Also, we only need to maintain the connectivity of the current solution. Similar to the CRP procedure, we do not employ branch exchange for EDA because EDA does not create offspring solutions by directly exchanging values between parent solutions but by sampling the probability distribution instead. This VO is problem-specific because it employs connectivity knowledge of DNEP when generating new offspring. We call this VO BX (i.e., branch exchanging). Figure 4.15 shows the pseudo-code for the realization of BX in GA and GOMEA.

Originally, GOMEAs do not have mutation operators by default. We here experiment with a DNEP-specific mutation. After every mixing event with a linkage set, but before the evaluation of the partially-altered solution, we go through every variable in the linkage set and perform a mutation with probability $1/l$ (l is the length of the solution). The mutated values must have the same sign as the original values so that the connectivity and radiality of the network are still maintained. We call the branch exchanging VO with this mutation operator BX-M. We will experiment with the combination of GA and BX-M as well.

We note that in literature the *branch exchange algorithm* exists [23]. This algorithm is similar to our BX operator in the idea of constructing connected and radial networks but is different in the purpose of its usage. In [23], the branch exchange algorithm is an optimization algorithm for DNEP to construct the expansion plan that optimizes the concerned objective. In this thesis, BX is an adaptation for the variation operators of GOMEA and GA to generate new connected and radial networks, while optimization itself is handled at a higher level, i.e., by GOMEA or GA.

4.6.4. Experimental results

Figure 4.16 shows the performance of GA, EDA and GOMEA integrated with different VOs solving the static DNEP. For Network 1 (see Figure 4.16a), almost all EA variants (except EDA-CRB) exhibit the same effectiveness in obtaining (near-) optimal solutions. After spending a certain number of evaluations, most solvers have similar convergence until termination. For Network 2 (see Figure 4.16b), GOMEA-BX(-M) performs slightly better than other EA variants, and the difference is found to be statistically significant. EDA-CRB is the worst optimizer when solving Networks 1 and 2. However, on small networks, although these differences between EA variants and between different VOs are statistically significant, they are practically negligible (e.g., differences of about 1,000-3,000 EUR for a planning period of 30 years).

In contrast, Figure 4.16c shows that, when solving DNEP for a larger network, the performance gaps between different EA variants are considerable. For Network 3, GA-DQ1 is outperformed by all other EA variants. Even though our EDA implementation here employs the UF model, EDA-DQ1 has a (statistically significantly) better performance than GA-DQ1, which suggests that the UF model sampling might be more effective than the MP model-based crossover operator (see Section 4.5.2). While also assuming no connectivity knowledge, within the same number of evaluations, GOMEA-DQ1 outperforms both GA-DQ1 and EDA-DQ1, and obtains solution plans of much better quality. On average, the expansion plans obtained by GOMEA-DQ1 costs about 150,000-280,000 EUR less than those obtained by GA-DQ1 and EDA-DQ1. These differences are found to be statistically significant. DQ100 gives GA multiple trials of recombination to generate new connected networks, and that indeed helps GA improve its performance significantly (but still worse than GOMEA-DQ1 in terms of the quality of obtained solutions at termination). DQ100 induces some small improvements for EDA-DQ1, but the differences are not statistically significant.

While the connectivity repair VOs CRP and CRB significantly enhance the effectiveness and efficiency of GA, they do not show any improvement over GOMEA-DQ1. For GOMEA, repairing by parent solutions (CRP) are (statistically significantly) better than repairing by the best solutions (CRB). GOMEA-CRB actually has the worst convergence behavior among all GOMEA variants, and even GOMEA-DQ1 can significantly outperform GOMEA-CRB. This can be because it is difficult for general-purpose VOs to generate connected networks and keeping matching unconnected offspring networks with the slowly-changing best topology reduces the beneficial diversity in the population, making the algorithm prone to premature convergence.

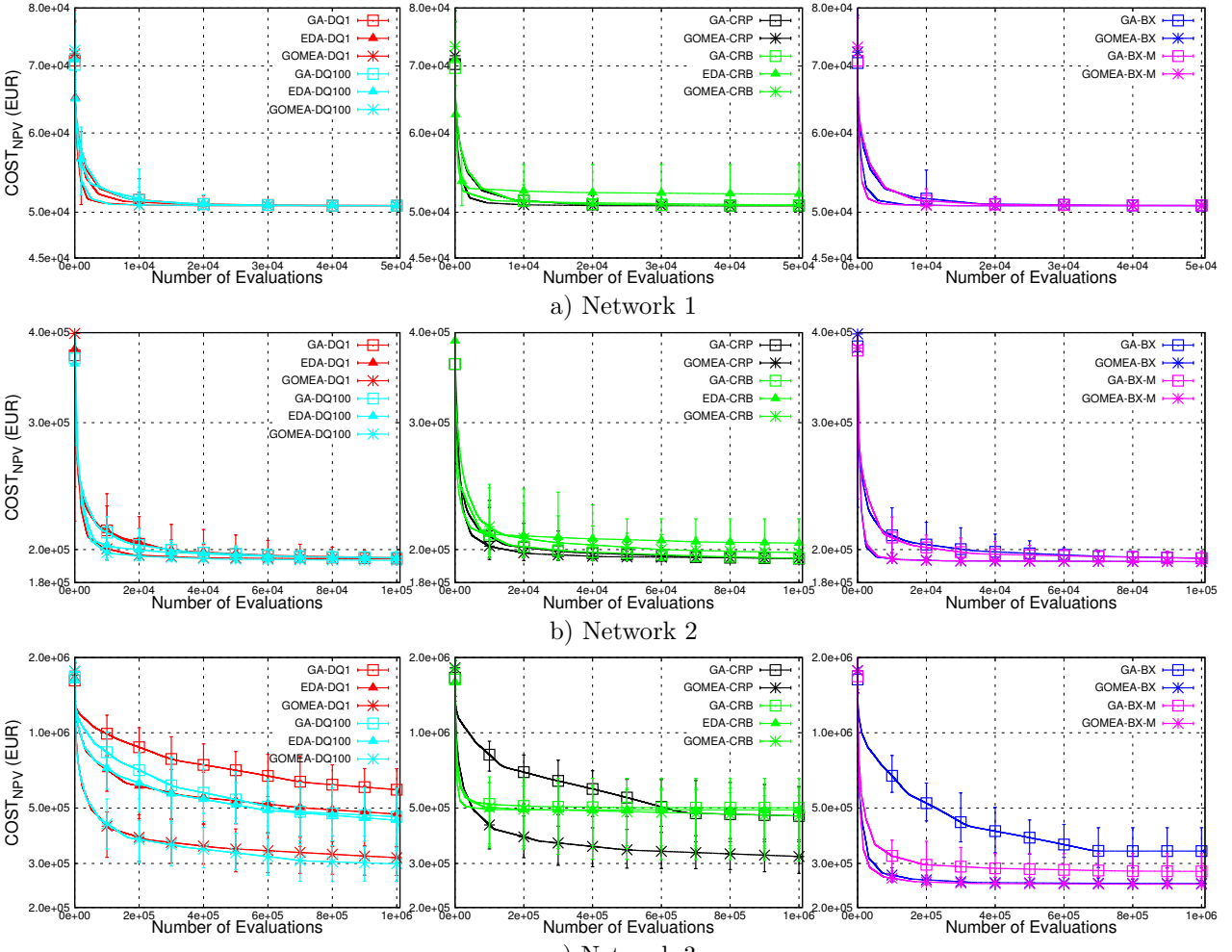

Figure 4.16: Results of computational experiments for static DNEP. Horizontal axis: number of evaluations. Vertical axis: Net Present Value (NPV) of total cost (EUR). Error bars show the maximum and minimum values of the NPV of total cost CAPEX+OPEX.

Solving the issue of maintaining connectivity and radiality, the VO BX brings out significant improvements in efficiency for GA and GOMEA when solving DNEP. GA-BX has excellent performance, obtaining significantly better solutions than GA-DQ1. BX also has positive impacts on GOMEA but the size of the effect is much less substantial than on GA. The results here suggest two conclusions. First, DNEP problem-specific VOs are crucial for GAs, to efficiently solve (larger) real-world networks. Second, GOMEA is the more robust solver, which can be used out-of-the-box and still obtain solutions of good quality close to those found by EAs customized with problem-dedicated VOs (i.e., comparing GOMEA-DQ1 in leftmost graphs with GA-BX and GOMEA-BX in rightmost graphs in Figure 4.16c). GOMEA-DQ1 is even found to obtain results that are statistically significantly better than those found by GA-BX.

The DNEP mutation operator has no influence over GOMEA-BX, but results in a statistically significant improvement for GA-BX. GA-BX-M, however, is still outperformed by GOMEA-BX(-M). On average, the expansion plans obtained by GOMEA-BX(-M) cost about 30,000 EUR less than the plans obtained by GA-BX-M at termination. Statistical tests support that the quality of expansion plans for Network 3 obtained by GOMEA-BX(-M) are significantly better than those obtained by all other EA variants. Thus, GOMEA-BX(-M) is the best solver in this test case (i.e., the fastest solver given the budget of evaluations used in our experiments).

4.7. Conclusions

This chapter contributes guidelines and methodologies for the application of EAs in tackling the real-world optimization DNEP. First, we formulated the DNEP problem, its feasibility constraints, and the objective cost function in a way that the obtained results would be practically relevant while the problem model is computationally feasible. Second, we suggested practitioners to employ population-sizing-free schemes to get rid of the notoriously-difficult-to-set population size parameter when using EAs. Third, we introduced multiple variation operators that can be employed by EAs solving DNEP and showed their impact on the performance of three typical EAs: the classic GA, an EDA, and the GOMEA, which is capable of learning and exploiting hierarchical linkage relations. Fourth, we compared the performance of these EA solvers in tackling the DNEP problem by performing experiments with real distribution network data. GOMEA is shown to be a far more robust solver for solving DNEP on our benchmark networks. Using the same number of solution evaluations as GA and EDA, GOMEA obtains better results, even when assuming a minimal amount of problem-specific knowledge (PSK). Adding PSK to GOMEA improves performance but the improvement gap, for a fixed budget of evaluations, is much less substantial than that for classic GA, which again confirms the usefulness of GOMEA and linkage learning (LL) to detect problem structure. As the problem size increases, LL is of great importance to ascertain efficient scalability of EAs. Lastly, based on our results, we suggest that LL EAs, like GOMEA, should be given priority when selecting EAs for solving DNEP.

References

[1] N. H. Luong, M. O. W. Grond, P. A. N. Bosman, and H. La Poutré, *Medium-voltage distribution network expansion planning with Gene-pool Optimal Mixing Evolutionary Algorithms,* in *11th International Conference, Evolution Artificielle, EA 2013, Bordeaux, France*, Lecture Notes in Computer Science, edited by P. Legrand, M.-M. Corsini, J.-K. Hao, N. Monmarché, E. Lutton, and M. Schoenauer (Springer International Publishing, Cham, 2014) pp. 93–105.

[2] M. O. W. Grond, N. H. Luong, J. Morren, P. A. N. Bosman, J. G. Slootweg, and H. La Poutré, *Practice-oriented optimization of distribution network planning using metaheuristic algorithms,* in *Proceedings of the Power Systems Computation Conference (PSCC 2014)* (IEEE, 2014) pp. 1–8.

[3] N. H. Luong, H. La Poutré, and P. A. N. Bosman, *Exploiting linkage information and problem-specific knowledge in evolutionary distribution network expansion planning,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2015* (ACM Press, New York, New York, USA, 2015) pp. 1231–1238.

[4] N. H. Luong, H. La Poutré, and P. A. N. Bosman, *Exploiting linkage information and problem-specific knowledge in evolutionary distribution network expansion planning,* Evolutionary Computation **26** (2018).

[5] I. J. Ramirez-Rosado and J. L. Bernal-Agustin, *Genetic algorithms applied to the design of large power distribution systems,* Power Systems, IEEE Transactions on **13**, 696 (1998).

[6] E. Diaz-Dorado, J. Cidras, and E. Miguez, *Application of evolutionary algorithms for the planning of urban distribution networks of medium voltage,* Power Systems, IEEE Transactions on **17**, 879 (2002).

[7] C. L. T. Borges and V. F. Martins, *Multistage expansion planning for active distribution networks under demand and distributed generation uncertainties,* International Journal of Electrical Power & Energy Systems **36**, 107 (2012).

[8] A. R. Jordehi, *Optimisation of electric distribution systems: A review,* Renewable and Sustainable Energy Reviews **51**, 1088 (2015).

[9] P. A. N. Bosman and D. Thierens, *More concise and robust linkage learning by filtering and combining linkage hierarchies,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2013* (ACM Press, New York, New York, USA, 2013) pp. 359–366.

[10] D. Thierens and P. A. N. Bosman, *Optimal mixing evolutionary algorithms,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2011* (ACM Press, New York, New York, USA, 2011) pp. 617–624.

[11] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, *Linkage problem, distribution estimation, and Bayesian networks.* Evolutionary computation **8**, 311 (2000).

[12] G. R. Harik, F. G. Lobo, and K. Sastry, *Linkage learning via probabilistic modeling in the Extended Compact Genetic Algorithm (ECGA),* in *Scalable Optimization via Probabilistic Modeling*, Studies in Computational Intelligence, Vol. 33, edited by M. Pelikan, K. Sastry, and E. Cantú-Paz (Springer Berlin Heidelberg, Berlin, Heidelberg, 2006) pp. 39–61.

[13] G. R. Harik and F. G. Lobo, *A parameter-less genetic algorithm,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 1999* (Morgan Kaufmann, 1999) pp. 258–265.

[14] M. Pelikan, A. Hartmann, and T.-K. Lin, *Parameter-less hierarchical Bayesian optimization algorithm,* in *Parameter Setting in Evolutionary Algorithms. Studies in Computational Intelligence.*, Vol. 54 (Springer Berlin Heidelberg, 2007) pp. 225–239.

[15] J. C. Pereira and F. G. Lobo, *A Java implementation of parameter-less evolutionary algorithms.* CoRR **abs/1506.0** (2015).

[16] M. C. da Rocha and J. T. Saraiva, *A discrete evolutionary PSO based approach to the multiyear transmission expansion planning problem considering demand uncertainties,* International Journal of Electrical Power & Energy Systems **45**, 427 (2013).

[17] J. J. Grainger and W. D. Stevenson, *Power system analysis* (McGraw-Hill, 1994).

[18] D. Van Hertem, J. Verboomen, K. Purchala, R. Belmans, and W. L. Kling, *Usefulness of DC power flow for active power flow analysis with flow controlling devices,* in *AC and DC Power Transmission, 2006. ACDC 2006. The 8th IEE International Conference on* (2006) pp. 58–62.

[19] M. O. W. Grond, J. I. Pouw, J. Morren, and J. G. Slootweg, *Applicability of line outage distribution factors to evaluate distribution network expansion options,* in *2014 49th International Universities Power Engineering Conference (UPEC)* (IEEE, 2014) pp. 1–6.

[20] K. Deb, *An efficient constraint handling method for genetic algorithms,* Computer Methods in Applied Mechanics and Engineering **186**, 311 (2000).

[21] R. A. Verzijlbergh, M. O. W. Grond, Z. Lukszo, J. G. Slootweg, and M. D. Ilic, *Network impacts and cost savings of controlled EV charging,* IEEE Transactions on Smart Grid **3**, 1203 (2012).

[22] M. O. W. Grond, *Impact of Future Residential Loads on Medium Voltage Networks*, Master thesis, Delft University of Technology (2011).

[23] K. Nara, T. Satoh, H. Kuwabara, K. Aoki, M. Kitagawa, and T. Ishihara, *Distribution systems expansion planning by multi-stage branch exchange,* IEEE Transactions on Power Systems **7**, 208 (1992).

5

Dynamic Distribution Network Expansion Planning

Twee vliegen in één klap slaan.
Two flies in one hit.

Dutch proverb

This chapter addresses the dynamic distribution network expansion planning (DNEP) problem, which involves the questions which, where, and when enhancements to electricity networks should be introduced to satisfy future power demands. Due to the time-dependent decision making factors, the dynamic DNEP problem cannot readily be tackled with available solvers that have been designed for the static planning case. Here, we propose a decomposition heuristic that can determine asset installation schedules for static expansion plans so that available solvers can be employed without deteriorating their performance. Besides electric cable reinforcements, in this chapter, we also consider the scenario in which battery energy storage systems (BESS) can be used by distribution network operators as an alternative to solve capacity bottlenecks. Experimental results demonstrate that mixtures of both traditional cable reinforcements and BESS can be more cost-effective than installing solely new cable connections.

Parts of this chapter have been presented at *GECCO '15* [1] and published in [2].

5.1. Introduction

Static distribution network expansion planning (DNEP) provides distribution network operators (DNOs) with the general picture of a network configuration that satisfies the peak power demand at the planning horizon. In other words, the static DNEP formulation addresses the questions *where* network enhancements should be made and *what* kinds of components should be installed there so that the network is still feasible in the final year of the planning period. However, to construct more detailed expansion plans with asset installation schedules, the dynamic version of the DNEP problem needs to be solved. The dynamic DNEP formulation involves the question *when* each network reinforcement activity should be started while in the static DNEP formulation this time-dependent decision making issue is omitted. Solving the dynamic DNEP problem is more difficult than the static version because of the presence of the time factor. It is not trivial to design an efficient model that captures both physical assets and the investment time. It is also not trivial to modify available solvers for static planning (e.g., those presented in Chapter 4) to properly and efficiently handle the temporal dimension. A different, and popular, alternative is to model the network configuration in every year during the planning period. In other words, the status in each year of each network location, where network reinforcements can take place, is a decision variable. With this encoding, available solvers in Chapter 4 can be employed without major modifications. However, because the number of decision variables increases considerably (i.e., the number of network components times the number of years in concern), a much larger number of solution evaluations is required for convergence or, at least, before any acceptable expansion plan is obtained. Furthermore, to check the feasibility of a dynamic expansion plan, power flow calculations (PFCs) need to be performed for the network configuration in every year during the planning period, instead of just the final year as in static DNEP. Dynamic DNEP with such a model is, therefore, inefficient and the required computing time might be prohibitively long. In this chapter, we argue that such a model contains a lot of redundancies because the capacity of a distribution network does not necessarily have to change every year. Therefore, it would be beneficial to design a method that allows expansion plans to be evaluated in a dynamic manner but allows the solution encoding to remain similar to the static codification so that the number of decision variables remains the same. We propose a heuristic that can be used during the optimization process to decompose static expansion plans into dynamic ones. Consequently, available solvers for static planning can be employed to solve the dynamic DNEP problem without compromising on their efficiency.

Besides the normal load growth, DNOs need to take into account the ubiquitous emergence of electric vehicles (EVs [3, 4]) and distributed generation (DG [5, 6]) technologies, which have big impacts on the sufficiency of the network capacity. For example, uncontrolled EV charging or unexpected surplus of DG power injection can considerably steepen the peak power consumption/production. The magnitudes of peak loads are very high but their periods are much shorter compared to the lower base loads. Therefore, instead of continuously installing/upgrading network cables to catch up with developments in peak loads, it might be more beneficial if

DNOs can employ smart grid alternatives to flatten the load profile. The option of demand side management [7], which incentivizes consumers to shift their flexible power consumption from peak hours to off-peak hours, is considered in Chapter 6. In this chapter, we investigate the option of battery energy storage systems (BESS) [8], which perform charging during off-peak hours or during times of surplus of DG power injection and discharging during peak hours. Note that we do not optimize the operation of storage systems in this thesis but we focus on their peak-shaving effects as an expansion alternative for DNOs. To solve potential bottlenecks due to peak load developments, DNOs can perform traditional cable enhancements as usual to increase the network capacity. On the other hand, we assume that DNOs can construct storage systems to bring peak loads within the current network capacity and costly new cable installations can, therefore, be postponed. The result of solving the dynamic DNEP problem with BESS is to find the optimal mixture of both cable reinforcements and storage systems during the planning period.

The remainder of this chapter is organized as follows. Section 5.2 presents the decomposition heuristic for the dynamic DNEP and its performance when combined with different evolutionary algorithm variants. Section 5.3 shows how battery storage systems can be modeled into the dynamic DNEP problem as expansion options like traditional electric cables. Experimental results in Section 5.3 show mixtures of cable reinforcements and storage installations that result from running our optimizer on a benchmark network in different scenarios of storage system prices. Section 5.4 gives some further discussion. Finally, section 5.5 concludes the chapter.

5.2. Dynamic DNEP by decomposition heuristic

5.2.1. Network configuration representation

Similar to Chapter 4, let l denote the total number of branches (cable connections) that can be considered in the optimization process (i.e., both existing cables and potential cable connections). While an expansion plan in the static DNEP problem can be represented by using only the network configuration in the horizon year (as in Chapter 4), a dynamic expansion plan requires information about the network configuration in every year during the planning period. Therefore, the network configuration from the beginning year t_0 until (and including) the final year $t_{horizon}$ can be represented as an $n_y \times l$ matrix $\boldsymbol{X}$ where $n_y = t_{horizon} - t_0 + 1$ is the number of years. Let $\Omega(k)$ denote the set of cable types that can be installed at branch k. Each entry x_k^t of $\boldsymbol{X}$, where $t_0 \leq t \leq t_{horizon}$ and $1 \leq k \leq l$, indicates the status of branch k in year t as:

- $x_k^t = ID > 0$: **Active cable**. A cable of type $ID \in \Omega(k)$ is installed at branch k.
- $x_k^t = 0$: **No cable connection**. There is no cable at branch k.
- $x_k^t = -ID < 0$: **A normally-open point (NOP)**. A cable of type $ID \in \Omega(k)$ is installed at branch k but is out of normal operation.

The first row of $\boldsymbol{X}$, $\boldsymbol{x}^0 = (x_1^0, x_2^0, \ldots, x_l^0)$, is the vector representing the currently existing network. The last row of $\boldsymbol{X}$, $\boldsymbol{x}^{horizon} = (x_1^{horizon}, x_2^{horizon}, \ldots, x_{n_l}^{horizon})$, is

the vector indicating the network configuration at the final year $t_{horizon}$. With this matrix encoding or other codifications with similar complexity, which is a common representation for dynamic planning in the literature [9–11], all the year-by-year changes in the network can be represented. However, in real-world DNEP, a network does not change frequently nor arbitrarily, given the current practice of DNOs, which is as follows. First, DNOs may replace a legacy cable with a new cable of higher capacity but DNOs do not completely remove an existing cable connection because such removals will reduce network capacity. Second, network cables (as our main asset category in this study) have very long lifetimes (about 30 years) while planning horizons of a too distant future can be regarded as impractical due to prohibitively high degrees of uncertainties in load growths and the emergence of new technologies. These facts suggest that, during a practical planning period (in this thesis, we assume a planning period $\leq$ 30 years), a network branch requires reinforcement at most once. Expansion activities such as installing a thin cable first and then replacing it with a higher-capacity cable are regarded as impractical because construction costs are typically very high. Because each entry x_k^t of $\boldsymbol{X}$ corresponds with a decision variable in the optimization process, the matrix encoding thus contains a lot of redundancy. Instead, we can compare $\boldsymbol{x}^{horizon}$ with $\boldsymbol{x}^0$ to know what new asset installations are required (i.e., if $x_k^{horizon} \neq x_k^0$). To find the asset installation schedule (i.e., the installation time of each new asset), we use the following decomposition heuristic.

5.2.2. Decomposition heuristic for dynamic DNEP

Given a forecast growth rate R of the peak power consumption, the current network $\boldsymbol{x}^0$, and a feasible candidate static plan $\boldsymbol{x}$ (which is assumed to be a network configuration in the horizon year $\boldsymbol{x} = \boldsymbol{x}^{horizon}$), we derive an installation schedule for new assets in $\boldsymbol{x}$ in two phases.

In the first phase, based on R, we determine the first year t_X when the current network becomes infeasible (i.e., when any operation constraint is violated, see Section 4.2.2). Then, we create the base installation schedule by assuming all new assets are installed at the same time in the year t_X. We evaluate the objective value (e.g., the total cost) of this base schedule (see Section 4.2).

In the second phase, we loop repeatedly through the list of all new assets in a random order. We create a new schedule by delaying the installation of an asset a by one year. We evaluate the feasibility and the objective value of this new schedule. If the new schedule is feasible and its corresponding objective value (e.g., total cost) is better than the previous schedule, we then accept that postponement and a can be considered for another postponement again in a next loop. Otherwise, the installation of that asset a cannot be delayed any further. We continue this checking for postponement until no asset can be postponed any more. Finally, we obtain a detailed year-by-year installation schedule for all new assets in an expansion plan and the concerned objective can be evaluated accordingly. We are argue that, based on the real-world practice of distribution network reinforcement in which a network branch generally requires enhancement at most once during a reasonable planning period (i.e., $\leq$ 30 years), this decomposition mechanism is sufficient for solving

dynamic planning. The codification and the solvers proposed for the static DNEP problem in Chapter 4 can be employed here to model and to solve the dynamic DNEP problem. The only modification is that the candidate solution evaluation in Chapter 4 is replaced by the decomposition heuristic so that an installation schedule of each feasible candidate solution is obtained and its corresponding costs can be calculated accordingly.

We note that the decomposition heuristic that we use in this study is different from the decomposition algorithm that was proposed in [12]. The decomposition algorithm in [12] divides a multi-year DNEP into multiple one-year DNEPs, where each one-year DNEP can be solved independently, and the asset installations of these sub-problems are coordinated through so-called "forward/backward procedures" in a recursive manner [12]. The decomposition algorithm in [12] can be seen as a framework to perform multiple single-year DNEPs and to synthesize the obtained results. Our decomposition heuristic is also different from the common 2-phase approach in dynamic DNEP (e.g., as in [13]). The 2-phase approach divides the optimization process into two separate phases: 1) solving the static DNEP to find the optimal network configuration in the horizon year; 2) determining the installation schedule for the new assets obtained in phase 1. In this thesis, the decomposition heuristic is a procedure that is embedded into the multi-year optimization process. During the optimization process, we evaluate different network configurations $\boldsymbol{x}$'s, and each one is assumed to be a candidate network configuration in the horizon year $\boldsymbol{x} = \boldsymbol{x}^{horizon}$. For each $\boldsymbol{x}$, the decomposition heuristic is used to derive an asset installation schedule of that network configuration so that its investment cost or its total cost (see Section 4.2.3) can be calculated accordingly. In other words, with our approach, optimization considers installation schedules *directly*.

5.2.3. Experimental results

Experiment setup

We evaluate the effectiveness of the proposed decomposition heuristic in transforming solvers for static planning into dynamic DNEP solvers via computational experiments. In Chapter 4, we have proposed three solvers GA, EDA, and GOMEA in combination with six variation operators (VOs) DQ1, DQ100, RB, RP, BX, and BX-M for the static DNEP problem. In this chapter, we employ the same EA variants and only modify their objective evaluation function such that the decomposition heuristic is used to determine the installation schedule for each *feasible* static plan, from which the dynamic total cost can be calculated accordingly. We also employ the same three distribution networks as in Chapter 4 as our benchmark networks here. The planning period for Networks 1 and 2 is still 30 years. Due to long computation time when PFCs for large networks need to be computed, we limit the planning period of Network 3 (i.e., the largest network in this study) to 10 years. For each benchmark network and variation operator, each EA is run 30 times independently. In each run, the maximum number of evaluations is 50,000 for Network 1, 100,000 for Network 2, and 1,000,000 for Network 3. Note that each solution evaluation includes checking the feasibility of the network configurations of that solution in every year during the planning period.

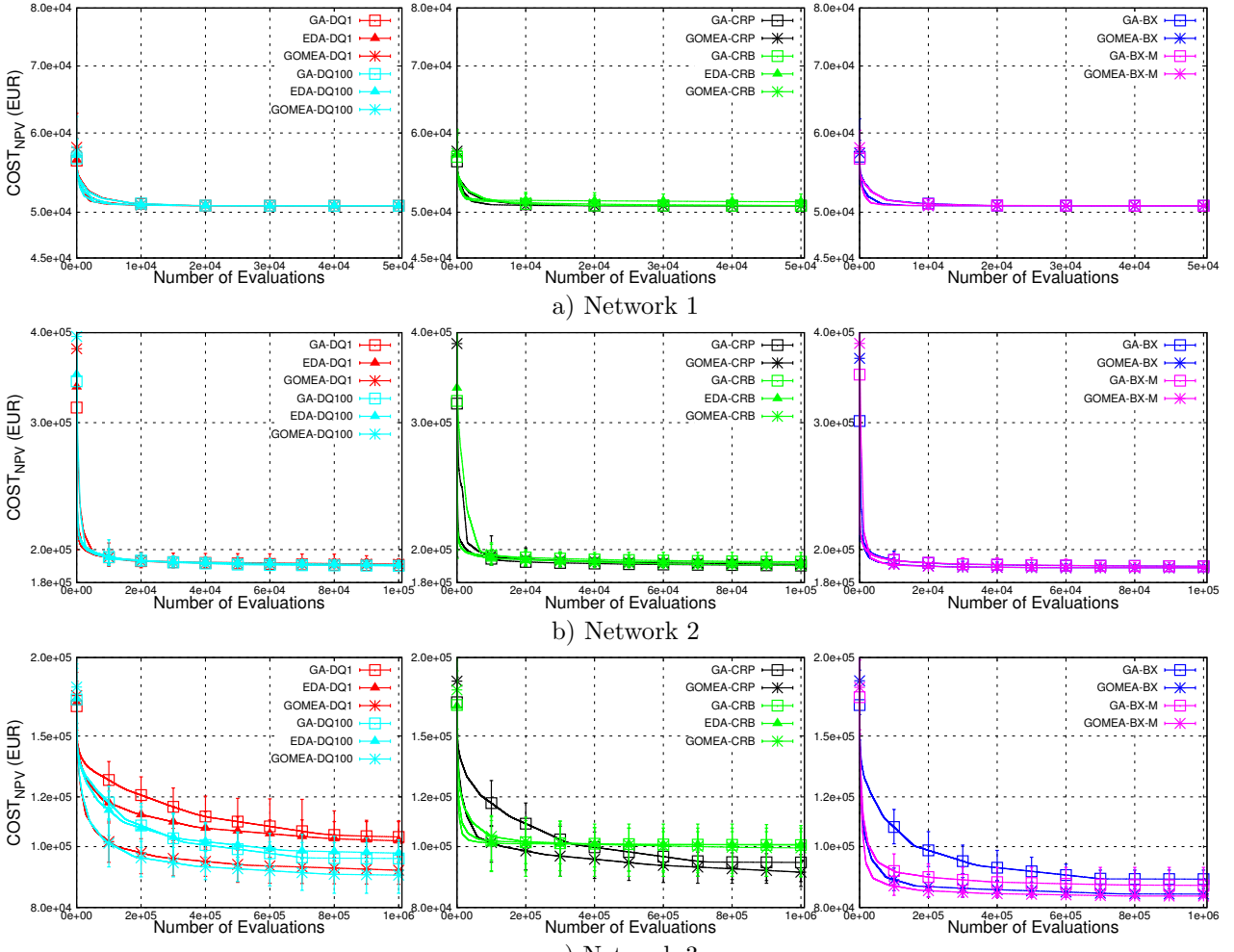

Figure 5.1: Results of computational experiments for dynamic DNEP. Horizontal axis: number of evaluations. Vertical axis: Net Present Value (NPV) of total cost (EUR). Error bars show the maximum and minimum values of the NPV of total cost CAPEX+OPEX.

Results

The average convergence graphs of the obtained elitist solutions during the optimization process (from the beginning until termination) are used as the basis for assessing the performance of the solvers. Figure 5.1 shows the convergence performance of GAs, EDAs and GOMEAs solving dynamic DNEP. We perform the Mann-Whitney-Wilcoxon statistical hypothesis test for equality of medians with $p < 0.05$ to see whether the final result obtained by one EA is statistically different from that of another EA. The general performances of all EA variants are similar to those observed in the case of static DNEP, suggesting that the use of the decomposition heuristic does not introduce additional complexity to the problem. For Networks 1 and 2 (see Figure 5.1a and 5.1b), the differences in the quality of solutions obtained by different EA variants are either statistically insignificant or practically negligible.

For Network 3, Figure 5.1c shows that GOMEA-DQ1 significantly outperforms both EDA-DQ1 and GA-DQ1. Even when only the simplest VO DQ1 is used, GOMEA can still obtain solutions of high quality, which are practically close to the ones found by the DNEP problem-specific EAs, i.e., GA-BX(-M) and GOMEA-BX(-M). The VO BX, which maintains the connectivity and radiality of offspring networks during solution recombination, is again shown to be the best VO, resulting in (statistically significantly) better performance when combined with both GA and GOMEA. In general, the more domain knowledge is well incorporated into variation operators, the better the performance of EAs. However, the improvement gaps that this problem-specific VO brings about for GOMEA-DQ1 (on average about 7,000 EUR) are not as substantial as for GA-DQ1 (on average about 14,000 EUR). For dynamic DNEP, BX-M (branch exchange with mutation) induces some small improvements for both GA-BX and GOMEA-BX, in which the differences are found to be statistically significant. GOMEA-BX-M is the overall best solver in this experiment, which (statistically significantly) outperforms other EA variants. While it might not be necessary for GOMEA solving many laboratory benchmarks, mutation can still be beneficial when tackling real-world problems because the linkage structure is not always the only or the most important structure to be exploited.

5.3. Dynamic DNEP with energy storage systems

In transition toward the future horizon of sustainable energy with smart grid technologies, network capacity expansion should not be limited to traditional physical asset reinforcements (e.g., electric cables, transformers). DNOs can make use of smart grid technologies to (in)directly reduce the magnitude of peak loads on the network so that the currently available network capacity would suffice for the corresponding power flow. Such peak shaving effects can be achieved, for instance, by employing battery energy storage systems (BESS) to charge during off-peak hours and then to discharge during peak hours. The operation of storage systems, however, is not the focus of this thesis and we thus abstract away from this. From the expansion planning perspective, we pursue the questions: what is the optimal mixture of electric cables and storage systems to solve capacity bottleneck problems, and when, and where should each reinforcement activity be carried out.

5.3.1. Storage system representation

In this thesis, we assume that battery storage systems can be installed at suitable substations in a distribution network, usually connected to the LV side of an MV/LV transformer. Compared to the lifetime electric cables, a BESS normally has a shorter lifetime. We also assume that battery storage systems are (re)movable, i.e., a BESS can be relocated among substations or even to a different networks, and its capacity can increase/decrease. Therefore, during the planning period, the status of a BESS can change multiple times rather than at most once, like electric cables. Instead of employing the static representation with the decomposition heuristic, each substation in the network where storage systems can be installed therefore requires a specific decision variable for each year during the planning period. However, similar to electric cables, we only consider storage system installations for the period from the first infeasible year t_X until (and including) $t_{horizon}$, i.e., a total of $n_y = t_{horizon} - t_X + 1$ years. Let S be the set of substations that are suitable for the installation of BESS, and $n_s = |S|$. We can represent the status of storage systems in a distribution network over n_y years as a vector of $n_y \times n_s$ non-negative integer elements.

$$\begin{aligned} &\boldsymbol{y} = (y_{s_1}^1, y_{s_1}^2, \ldots, y_{s_1}^{n_y}, y_{s_2}^1, y_{s_2}^2, \ldots, y_{s_2}^{n_y}, \ldots, y_{s_{|S|}}^1, y_{s_{|S|}}^2, \ldots, y_{s_{|S|}}^{n_y}) \\ &y_s^t \in \mathbb{S} \cup \{0\}, \quad s \in S, \quad t = 1, 2, \ldots, n_y \end{aligned} \tag{5.1}$$

where y_s^t represents the storage type that is installed at the substation $s \in S$ in year t. $\mathbb{S}$ is the set of available storage types, and $y_s^t = 0$ indicates that no storage is installed at the substation s in the year t.

A solution $\boldsymbol{s}$ of the DNEP problem with BESS for a distribution network over a planning period consists of the cable configuration $\boldsymbol{x}^{horizon}$ in the final year and the storage configuration $\boldsymbol{y}$. For the sake of convenience in representation, we shorten $\boldsymbol{x}^{horizon}$ as $\boldsymbol{x}$.

$$\boldsymbol{s} = (\boldsymbol{x}, \boldsymbol{y}) = (x_1, x_2, \ldots, x_l, y_{s_1}^1, \ldots, y_{s_1}^{n_y}, y_{s_2}^1, \ldots, y_{s_2}^{n_y}, \ldots, y_{s_{|S|}}^1, \ldots, y_{s_{|S|}}^{n_y}) \tag{5.2}$$

The total number of decision variables is then $L = l + n_y \times n_s$. An installation schedule of new electric cables in $\boldsymbol{x}$ is determined by the installation decomposition approach. The storage configuration $\boldsymbol{y}$ indicates whether and what type of storage is installed at each suitable substation in each year.

5.3.2. Solution evaluation

Constraint evaluation

For an MV-D network of n nodes (i.e., substations), the vector of peak active power demand at each node in year t is:

$$\boldsymbol{P}^t = (P_1^{t,s_i^t}, P_2^{t,s_i^t}, \ldots, P_n^{t,s_i^t}) \tag{5.3}$$

The accompanying vector of peak reactive power demand is:

$$\boldsymbol{Q}^t = (Q_1^{t,s_i^t}, Q_2^{t,s_i^t}, \ldots, Q_n^{t,s_i^t}) \tag{5.4}$$

Let P_i^{t,s_i^t} and Q_i^{t,s_i^t} be the peak active power demand and the peak reactive power demand at node i in year t. If node i is not suitable for storage installation or there is no storage installed at node i in year t, then $s_i^t = 0$; we have $P_i^{t,s_i^t} = P_i^{t,0}$ and $Q_i^{t,s_i^t} = Q_i^{t,0}$, which are the normal peak power demands when no peak-shaving technology is employed. If a BESS is installed at node i in year t, i.e., $s_i^t > 0$, we have $P_i^{t,s_i^t} < P_i^{t,0}$ and $Q_i^{t,s_i^t} < Q_i^{t,0}$ due to the peak-shaving effect. Because this thesis focuses on optimization methodologies for DNEP, we treat $\boldsymbol{P}^t$ and $\boldsymbol{Q}^t$ as input data. By employing scenario-based load modeling techniques, DNEP practitioners can propose several plausible scenarios and estimate the corresponding load profile. Interested readers can refer to the literature, e.g., network impacts under electric vehicle charging [3, 14], residential demand response [15], or distributed generation [6, 16].

To assess the feasibility of an expansion plan $\boldsymbol{s} = (\boldsymbol{x}, \boldsymbol{y})$ for the dynamic DNEP problem, we need to evaluate the cable configuration $\boldsymbol{x}$ against the peak power demand $\boldsymbol{P}^t, \boldsymbol{Q}^t$ in each year t during the planning period, taking into account the peak-shaving effect of storage system $\boldsymbol{y}$. Note that the cable configuration in each year t is determined by the decomposition heuristic. The solution codification in Equation 5.2 enables the capacity of each storage, in principle, to vary on an annual basis (i.e., from year to year). It is also possible to extend t into a stage of multiple years, where each stage covers a period of at most 5 years and at least 3 years. The decomposition heuristic will then try to delay each cable installation by one stage of multiple years at a time. New cables and storage systems of a stage are assumed to be installed at the first year of that stage. Network feasibility in a stage, however, is evaluated against the peak power demand of the final year in that stage. Because peak loads normally increase monotonically and the network configuration does not change during a stage, if the network capacity is sufficient at the final year of that stage, feasibility should also hold for previous years of the stage. Such a stage-by-stage planning approach is beneficial to the computing budget since the computationally expensive PFCs are only required at the final years of each stage rather than at every year. In this section, we employ the stage-by-stage planning approach.

Objective evaluation

Although we perform stage-by-stage planning, the investment cost CAPEX can still be calculated in a year-by-year formulation by the annuity payment scheme. The annuity for a BESS of type $b \in \mathbb{S}$ with a discount rate $i = 4.5\%$ is:

$$Annuity(b) = Price(b) \times \frac{i}{1 - (1+i)^{-t_{life}^b}} \tag{5.5}$$

where $t_{life}^b = 10$ years is the economic lifetime of all storage systems in this study. $Price(b)$ is the acquisition cost (including construction cost) of a BESS of type b, which can be defined as:

$$Price(b) = Capacity(b) \times C_{kWh} \tag{5.6}$$

Theoretically, a BESS can be constructed up to any desired capacity by assembling multiple batteries. However, for the sake of simplicity, we constrain the capacity of each BESS to three options: type 1 (500 kWh), type 2 (1000 kWh), and type 3 (1500 kWh). These capacities are also reasonable regarding the normal capacities of MV/LV transformers. C_{kWh} is similar to the marginal cost of increasing the capacity of the battery storage by 1 kWh.

Because we assume that the battery storage systems are movable, it is possible that a BESS is installed at a stage in the network, but it can be relocated to a different network in the next stage. The CAPEX of that BESS accounts for only the years in which the storage is present in the network. CAPEX for the BESS at a substation $s \in S$ in a year t can be calculated as:

$$CAPEX_{storage}(s,t) = \begin{cases} Annuity(y_s^t) & \text{if } y_s^t > 0 \\ 0 & \text{if } y_s^t = 0 \end{cases} \tag{5.7}$$

The CAPEX for new cable installations can be calculated as in Equation 4.3 of Chapter 4. The total CAPEX in a year t over the whole network is defined as:

$$CAPEX(t) = \sum_{\substack{new\ cable\ c\ in \\ [t_0,\ t_{horizon}]}} CAPEX_{cable}(c,t) + \sum_{s \in S} CAPEX_{storage}(s,t) \tag{5.8}$$

We minimize the net present value (NPV) of the total CAPEX over the planning period with a discount rate i:

$$CAPEX_{NPV} = \sum_{t=t_X}^{t_{horizon}} \frac{CAPEX(t)}{(1+i)^{t-t_0}} \tag{5.9}$$

Energy losses can be taken into account as operational expenditure OPEX, similar to Equation 4.7 in Chapter 4. Energy losses in year t consist of energy loss on electric cables and energy loss of storage systems.

$$\begin{aligned} E_{loss}(t) &= E_{cable\ loss}(t) + \sum_{s \in S} E_{storage\ loss}(s,t) \\ &= P_{peak\ loss}(t) \times T_{loss}(t) + \sum_{s \in S} E_{storage\ loss}(s,t) \end{aligned} \tag{5.10}$$

where $P_{peak\ loss}(t)$ is the peak loss which can be obtained from the PFC regarding the peak loads in year t. Note that the service time $T_{loss}(t)$ here depends on the shape of the yearly load profile under the peak-shaving effects of the battery storage systems present in the network in year t. In this thesis, we also treat $T_{loss}(t)$ and the annual energy losses of storage systems $E_{storage\ loss}(s,t)$ as input data, which are device-specific and can be modeled by DNEP practitioners. As we conduct the dynamic planning on a stage-by-stage basis, we only need to perform PFCs for the final year of each stage and obtain the accurate value of peak loss on cables $P_{peak\ loss}(t)$ in that year. For the remaining years in the same stage, cable peak

losses can be estimated by taking the assumption that the peak loss has a growth rate related to the load growth rate R as follows [17]:

$$P_{peak\ loss}(t) = P_{peak\ loss}(t-1) \times (1+R)^2 \tag{5.11}$$

We would like to find the expansion plan that minimizes the net present value (NPV) of the total cost CAPEX+OPEX over the planning period with a discount rate i:

$$COST_{NPV} = \sum_{t=t_X}^{t_{horizon}} \frac{CAPEX(t) + OPEX(t)}{(1+i)^{t-t_0}} \tag{5.12}$$

5.3.3. Experiment setup

Benchmark network

We employ the MV-D Network 2 from Chapter 4 as the benchmark network. However, for the purpose of DNEP with battery storage systems, we have some adaptations as follows. All the cables in the current configuration are assumed to be a type-2 cable (i.e., $150mm^2 - 295A$). Since the type-2 cable is relatively new, we also assume that cable replacements are not necessary, and only new cable connection installations are allowed. The set of MV cable types from which new cable installations can be chosen is $\{2, 3\} = \{150mm^2, 240mm^2\}$.

The peak load scenario is estimated for the planning period 2015-2040, taking into account the increasing presence of electric vehicles, like in [3, 14]. The existing peak power demand is assumed to have an annual growth rate of $R = 1\%$. Each year in the planning period is numbered from 0 (i.e., year 2015) to 25 (i.e., year 2040). Without any expansion activities, the current network capacity will become infeasible in the 3rd year, i.e., $t_X = 3$. The remaining period is divided into five stages as: $S_1 = [3-5], S_2 = [6-10], S_3 = [11-15], S_4 = [16-20], S_5 = [21-25]$. The set of BESS types from which new storage installations can be chosen is $\mathbb{S} = \{1, 2, 3\} \equiv \{500kWh, 1000kWh, 1500kWh\}$. Due to uncertainties about the actual price of storage systems in the future, we propose four scenarios of the marginal BESS cost $C_{kWh} = 1$, 5, 10, and 20 EUR/kWh. Note that these (fictive) prices are much lower than the current actual construction costs of BESS (e.g., see [18]) since it is an on-going research technology. If we use the current BESS prices, which are quite expensive, the most economical expansion plan will only consist of traditional electric cable installations. Besides, our considered prices can be justified if subsidies are taken into account. The peak-shaving effect of BESS is modeled by employing a scenario-based peak load modeling technique [14, 16] while considering a BESS pilot project of DNO Enexis [19].

Optimization algorithm

We employ GOMEA-BX-M to solve DNEP with BESS for the benchmark network because GOMEA-BX-M has been shown to be the best solver in previous experiments in this thesis. We consider two cases for optimization: 1) minimizing the investment cost CAPEX (i.e., Equation 5.9), and 2) minimizing the total cost that combines both the investment cost CAPEX and the capitalized energy losses OPEX

(i.e., Equation 5.12). In each case, we set up one scenario where no BESS is used (i.e., the traditional DNEP) and four scenarios with different BESS prices: 1, 5, 10, and 20 EUR/kWh. For every scenario, we run GOMEA-BX-M 10 times independently with the computing budget of 500000 solution evaluations in each run. The adapted Harik-Lobo scheme (see Chapter 4) is employed so that we do not need to set the population size for the solver. The best expansion plans obtained in each scenario are presented in the following sections.

5.3.4. DNEP with only cable expansions

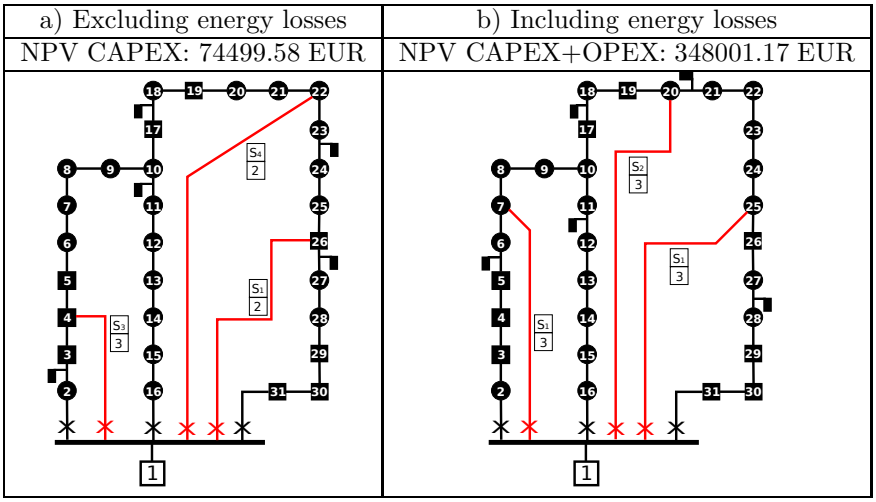

Figure 5.2: Best expansion plans found for the dynamic DNEP with only electric cable installations. a) Minimizing NPV investment cost CAPEX; b) Minimizing NPV total cost CAPEX+OPEX.

Figure 5.2 shows the two best found expansion plans when using GOMEA-BX-M to solve the dynamic DNEP problem for the benchmark network above in two cases: a) minimizing CAPEX, and b) minimizing the total cost of CAPEX and OPEX. In both cases, the effect of the decomposition heuristic can be seen: the required cable installations are scheduled over the entire planning period. When optimizing for only the investment cost CAPEX, the solution in Figure 5.2a indicates that we can firstly install a new cable connection of type 2 in branch 1-26 to solve the first bottleneck. This reinforcement provides the network with enough capacity until the 3rd stage (i.e., years 11-15), when another cable connection of type 3 needs to be newly installed in branch 1-4. The new network capacity is sufficient until the 4th stage (i.e., years 16-20), and we need to enhance the network capacity again with a new cable connection of type 2 in branch 1-22. It can be seen that new cables are installed to solve the capacity problem when bottlenecks happen on the network.

On the other hand, when we capitalize energy losses as the operational cost OPEX and include that into the objective function, the best found expansion plan, as in Figure 5.2b, indicates that two new cable connections in branches 1-7 and 1-25

should be added in the first stage. In the second stage, another cable connection should be installed in the branch 1-20. All new cable installations are carried out with cable type 3, which is the highest-quality cable type in this experiment. Network reinforcements in this case are not only for solving capacity bottlenecks but mainly for reduction of energy losses, since cable type 3 suffers less losses than cable type 2. Comparing the two expansion plans in Figures 5.2 and their corresponding costs shows that, within the planning period, the OPEX dominates the total cost function. Thanks to the annuity scheme, the acquisition cost of each asset does not need to be wholly paid at once but can be divided into equal payments during the economic lifetime of that asset. Electric cables have long economic lifetimes which can exceed most practical planning periods, which are usually not more than 30 years. Some asset payments, therefore, will be beyond the planning horizon while the CAPEX is formulated to be computed up to the final year of the planning period. This explains why the OPEX has a much bigger impact on the total cost within a fixed planning period. Even though cable type 3 is more expensive than cable type 2, the reduction in energy losses can offset the increase in CAPEX.

5.3.5. DNEP with storage systems

As smart grid alternatives, and energy storage systems in particular, are on-going technologies, their implementations and prices can vary considerably, which makes it difficult for a DNEP practitioner to design network enhancement strategies. The methodology and optimization technique presented in this chapter can be used by practitioners to investigate different scenarios. While the obtained expansion plans might not be the concrete implementation, they can be used as a good basis for reference and further developments.

Minimizing CAPEX

Figrue 5.3 shows the best found expansion plans for solving DNEP without energy losses on the benchmark network in four scenarios with different BESS prices: 1, 5, 10, and 20 EUR/kWh. In all four scenarios, the obtained solutions are different combinations of cable and storage system installations, where their associated total investment costs are more economical than using only cable options. The composition of each expansion plan depends on the storage price in its corresponding scenario. In general, when storage prices are favorable, it is more cost-effective to employ BESS to reduce peak loads and the magnitudes of power flows will then be brought back within the network's asset capacities.

For the purpose of demonstration, we analyze the result obtained at the storage price of 1 EUR/kWh. It can be seen in Figure 5.3a, that new cable installations are not necessary at the first infeasible year as in the traditional case (see Figure 5.2a). Instead, storage systems of type 1 (i.e., 500 kWh) are installed at four different substations in the network. The first cable expansion is then deferred until the second stage. Due to such cable expansion, the network capacity increases considerably, and as a result, no storage system is required at this stage. Because we employ BESS as storage options and we assume that the batteries are movable, it can be understood that the storage systems installed in the previous stage can be relocated to be used

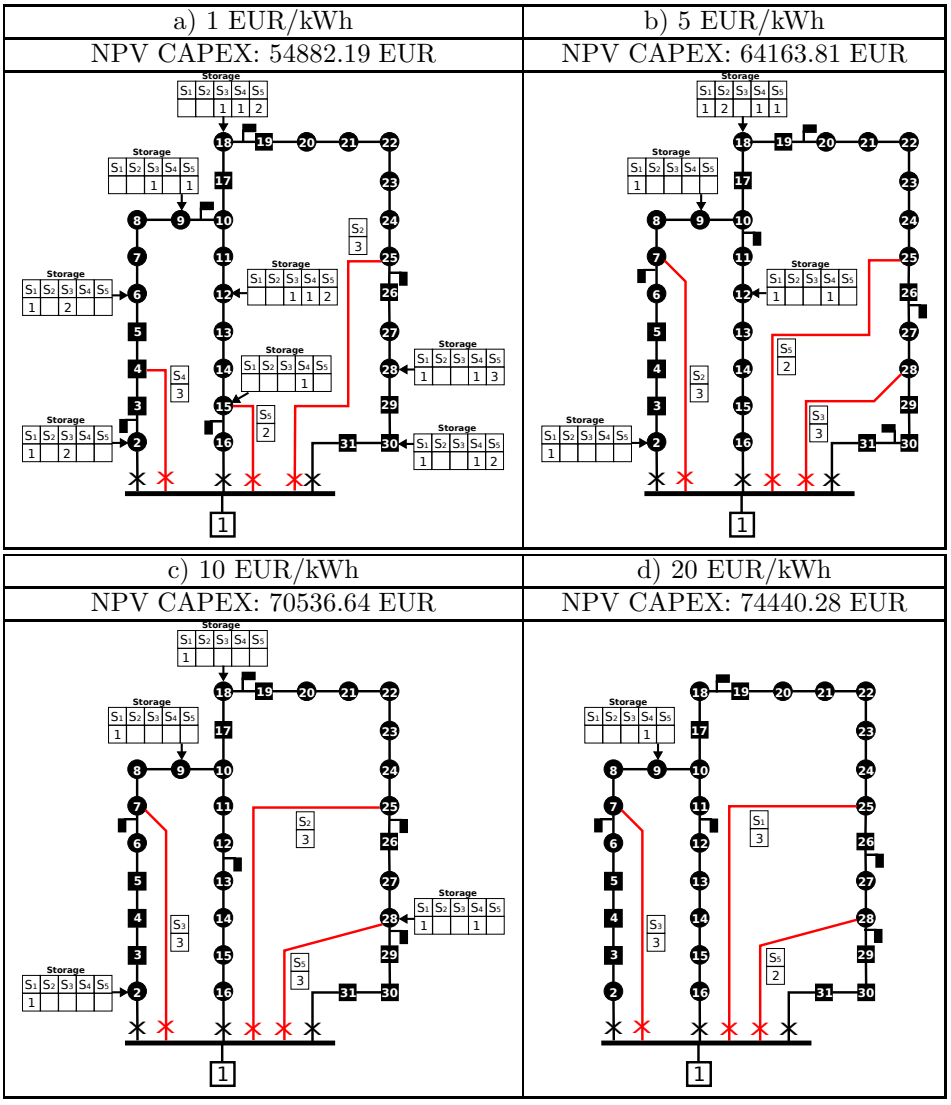

Figure 5.3: Optimal expansion plans of minimized NPV CAPEX for the dynamic DNEP with battery energy storage systems (BESS) in four scenarios of BESS prices: a) 1 EUR/kWh, b) 5 EUR/kWh, c) 10 EUR/kWh, and d) 20 EUR/kWh.

in other different distribution networks, and their associating annuity payments are not included in the CAPEX of this network anymore. In the third stage, instead of performing the second cable reinforcement as in Figure 5.2a, storage systems are again preferable and widely employed while the new cable installation is postponed until the 4th stage. In the 4th and 5th stages, we can observe the presence of both cable expansions and storage systems. This is due to several factors: 1) annual

peak loads of the last years of the planning period are usually higher than previous years; 2) the cost of installing new cables in the branches of short distances (i.e., 1-4 and 1-15) is quite low; 3) the annuity payment scheme excludes a large part of the acquisition costs of the assets installed in the last stage from the CAPEX of our concerned planning period. Figure 5.3 shows the trend that as the price of BESS increases, its usage becomes less economically attractive while traditional cable expansions are then carried out sooner. For example, at the most expensive storage price of 20 EUR/kWh in this study (see Figure 5.3d), the best found expansion plan employs only a small-scaled BESS in the 4th stage, and overall, it does not considerably differ from the scenario of using only cable expansions.

Minimizing CAPEX+OPEX

We perform similar experiments as in the above section again but with a new objective function that consists of capital expenditures (CAPEX) and monetized energy losses (OPEX). Energy losses in this case are calculated from both losses over cables and losses of storage systems. Interestingly, the best found solution in all scenarios of storage prices is the same expansion plan that was obtained when we only considered traditional cable installations, i.e., no storage system is installed. Figure 5.2 and our analysis in Section 5.3.4 show that installing three new cable connections with the higher-quality cable type (i.e., type 3) provides enough capacity during the whole planning period and considerably reduces energy losses. Consequently, it is not necessary to install any storage system since the capacity bottleneck has been removed. Besides, due to the annuity payment scheme, the OPEX dominates the total cost function of the planning period and it is thus cost-effective to invest in cable capacity as early as possible to reduce energy losses. Note that the result here pertains to this specific benchmark network and the formulated BESS types. Performing optimization on different distribution networks with different inputs (e.g., peak-load profile, cable and storage types, asset prices) will yield different expansion plans.

5.4. Further discussions

DNEP practitioners can design different possible optimization scenarios, e.g., which cable types can be used, including/excluding energy losses, with/without smart grid devices, or how much storage systems cost. For each scenario, GOMEA, and specifically GOMEA-BX-M, can be used to find a high-quality solution corresponding to the input parameters. However, our experimental results also indicate a limitation of single-objective optimization when trying to find the optimal solution for two objectives at the same time that are very different in their nature (i.e., investment cost and energy losses). Because single-objective optimization techniques optimize for a single objective, the two objectives must be aggregated into a single cost function. To do so, we need to capitalize the non-financial term energy losses as the operational cost (OPEX) before combining it with the investment cost (CAPEX). This is based on assumptions about energy prices in the future, which might fluctuate during the planning period. In this experiment, when the OPEX dominates the cost function, the solver yields the solution that chooses to heavily invest in electric

cables to minimize energy losses. Energy loss is an important issue that should not be excluded when performing DNEP, but it might not be the deciding factor in selecting a concrete expansion plan to be carried out. Single-objective optimization techniques normally return a single solution with a minimized lump sum cost where the contribution of each constituent is not explicit to DNEP practitioners. Instead, practitioners might want to consider a number of different alternatives that explicitly show the trade-off between CAPEX and OPEX before finalizing any plan. To this end, true multi-objective optimization methods are required that are scalable and also customizable for the DNEP problem. In Chapters 3 and 6 we focus on such methods.

5.5. Conclusions

This chapter contributed guidelines for modeling and tackling the dynamic distribution network expansion planning problem (DNEP). We observed that during practical planning periods (i.e., normally not more than 30 years) each traditional asset, such as long-lifetime electric cables, requires reinforcement/installation at most once. We then proposed a decomposition heuristic that enables available static planning solvers to tackle the dynamic DNEP without considerably increasing problem complexity. Experimental results demonstrated that with the decomposition heuristic, EA variants that we proposed previously (i.e., GA, EDA, and GOMEA with different variation operators as in Chapter 4) can solve the dynamic DNEP problem while still retaining their general performance as in the static planning case. The linkage learning EA GOMEA with the problem-specific branch exchange operator with mutation (i.e., GOMEA-BX-M) was again shown to be the best optimizer.

We also considered battery energy storage systems (BESS) as an expansion option for the dynamic DNEP problem. Different from electric cables, a BESS normally has a shorter lifetime and its configuration can change multiple times during the planning period. Therefore, in DNEP, we used a dynamic formulation to incorporate BESS and the static formulation in combination with the decomposition heuristic for cable installations. We then performed a demonstration of tackling the dynamic DNEP with BESS on a selected benchmark network by employing the solver GOMEA-BX-M (the best found EA solver in previous sections). Due to uncertainty in BESS prices in the future, we proposed different scenarios of BESS costs and performed experiments for each scenario. Results demonstrated that expansion plans with mixtures of both electric cables and BESS are more economical than using exclusively cable options. BESS was shown to be a smart grid alternative that can be used by distribution network operators (DNOs) to defer expensive electric cable installations. The methodology presented in this chapter is a highly customizable expansion planning tool that can be used by DNOs to assist their decision making process. Lastly, this chapter also pointed out intrinsic limitations of single-objective optimization techniques when dealing with multiple conflicting objectives at the same time. We indicated, therefore, the necessity for the design and application of multi-objective optimization algorithms, which is the focus of Chapters 3 and 6, respectively.

References

[1] N. H. Luong, H. La Poutré, and P. A. N. Bosman, *Exploiting linkage information and problem-specific knowledge in evolutionary distribution network expansion planning,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2015* (ACM Press, New York, New York, USA, 2015) pp. 1231–1238.

[2] N. H. Luong, H. La Poutré, and P. A. N. Bosman, *Exploiting linkage information and problem-specific knowledge in evolutionary distribution network expansion planning,* Evolutionary Computation **26** (2018).

[3] E. Veldman and R. A. Verzijlbergh, *Distribution grid impacts of smart electric vehicle charging from different perspectives,* IEEE Transactions on Smart Grid **6**, 333 (2015).

[4] K. M. Tan, V. K. Ramachandaramurthy, and J. Y. Yong, *Integration of electric vehicles in smart grid: A review on vehicle to grid technologies and optimization techniques,* Renewable and Sustainable Energy Reviews **53**, 720 (2016).

[5] C. L. T. Borges and V. F. Martins, *Multistage expansion planning for active distribution networks under demand and distributed generation uncertainties,* International Journal of Electrical Power & Energy Systems **36**, 107 (2012).

[6] B. Asare-Bediako, W. Kling, and P. Ribeiro, *Future residential load profiles: Scenario-based analysis of high penetration of heavy loads and distributed generation,* Energy and Buildings **75**, 228 (2014).

[7] L. Gelazanskas and K. A. Gamage, *Demand side management in smart grid: A review and proposals for future direction,* Sustainable Cities and Society **11**, 22 (2014).

[8] E. Reihani, S. Sepasi, L. R. Roose, and M. Matsuura, *Energy management at the distribution grid using a Battery Energy Storage System (BESS),* International Journal of Electrical Power & Energy Systems **77**, 337 (2016).

[9] E. Carrano, R. Cardoso, R. Takahashi, C. Fonseca, and O. Neto, *Power distribution network expansion scheduling using dynamic programming genetic algorithm,* IET Generation, Transmission & Distribution **2**, 444 (2008).

[10] P. Maghouli, S. H. Hosseini, M. Oloomi Buygi, and M. Shahidehpour, *A scenario-based multi-objective model for multi-stage transmission expansion planning,* IEEE Transactions on Power Systems **26**, 470 (2011).

[11] H. Saboori, R. Hemmati, and V. Abbasi, *Multistage distribution network expansion planning considering the emerging energy storage systems,* Energy Conversion and Management **105**, 938 (2015).

[12] K. Nara, T. Satoh, K. Aoki, and M. Kitagawa, *Multi-year expansion planning for distribution systems,* IEEE Transactions on Power Systems **6**, 952 (1991).

[13] H. Falaghi, C. Singh, M.-R. Haghifam, and M. Ramezani, *DG integrated multi-stage distribution system expansion planning,* International Journal of Electrical Power & Energy Systems **33**, 1489 (2011).

[14] R. A. Verzijlbergh, M. O. W. Grond, Z. Lukszo, J. G. Slootweg, and M. D. Ilic, *Network impacts and cost savings of controlled EV charging,* IEEE Transactions on Smart Grid **3**, 1203 (2012).

[15] S. Gyamfi and S. Krumdieck, *Scenario analysis of residential demand response at network peak periods,* Electric Power Systems Research **93**, 32 (2012).

[16] E. Veldman, M. Gibescu, J. G. Slootweg, and W. L. Kling, *Scenario-based modelling of future residential electricity demands and assessing their impact on distribution grids,* Energy Policy **56**, 233 (2013).

[17] M. O. W. Grond, N. H. Luong, J. Morren, P. A. N. Bosman, J. G. Slootweg, and H. La Poutré, *Practice-oriented optimization of distribution network planning using metaheuristic algorithms,* in *Proceedings of the Power Systems Computation Conference (PSCC 2014)* (IEEE, 2014) pp. 1–8.

[18] H. L. Ferreira, R. Garde, G. Fulli, W. Kling, and J. P. Lopes, *Characterisation of electrical energy storage technologies,* Energy **53**, 288 (2013).

[19] J. G. Slootweg, R. De Groot, S. Schouwenaar, and F. Van Overbeeke, *Smart storage in the Enexis LV distribution grid,* in *22nd International Conference and Exhibition on Electricity Distribution (CIRED 2013)* (Institution of Engineering and Technology, 2013) pp. 0583–0583.

6

Multi-Objective Dynamic Distribution Network Expansion Planning with Demand Side Management

Wat de boer aan het koren verliest, zal hij aan het spek wel terugvinden.
When the corn decreases, the pig increases.

Dutch proverb

In distribution network expansion planning (DNEP) for future smart grids with renewable energy sources, demand side management (DSM) plays a crucial role. In this chapter, we incorporate the possibility for distribution network operators (DNOs) to employ DSM possibilities besides traditional asset investments. We consider a dynamic planning version of the DNEP problem as well as optimizing for multiple conflicting real-world objectives simultaneously. To this end, we use and extend the multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm (MO-GOMEA). The found results provide a wide range of alternative solutions that represent efficient trade-offs between multiple criteria. From these alternatives, DNOs can obtain valuable insights and make well-informed decisions in the design and expansion of their networks. For example, it can be observed how various DSM options can defer costly new asset installations, how much network reliability must be compromised if DNOs pursue economical expansion plans, or the size of required asset investments to improve network energy efficiency.

Parts of this chapter have been presented at *PES-GM '15* [1] and submitted for a book chapter contribution [2].

6.1. Introduction

6.1.1. Asset investment & demand side management

In the transition toward smart grids and renewable energy integration, distribution network operators (DNOs) face many challenges when tackling the distribution network expansion planning (DNEP) problem [3–6]). Traditionally, DNOs need to ensure that their network assets (e.g., cables, transformers) have enough capacity to support the required power flows that satisfy the power demands, especially in the worst-case scenario of peak loads. Situations of insufficient capacity can lead to overloads on network assets, which are harmful to network safety (e.g., cable failures) and the continuity of electricity supply (e.g., blackouts, or load shedding). However, constructions of new facilities become increasingly expensive such that expansion plans like installing new underground cables in highly-urbanized cities are often not favorable. Moreover, peak loads can become high but last for short duration compared to the lower base loads of much longer periods. This is even more the case with the future increase of renewable energy sources. This fact means that if DNOs invest in physical assets just to satisfy (higher) peak loads, their networks will be severely underutilized for most of the time. Such overcapacity is uneconomic and undesirable. In this chapter, we assume that, in future sustainable energy systems, DNOs are allowed to directly be involved in demand side management (DSM [7–10]) activities so that consumers are incentivized to reduce their electricity consumption during peak-demand hours. The overall peak loads on the distribution network can then be brought toward or within the nominal capacities of network assets, aiming at avoiding overloads of network assets. DNOs can thus reduce or postpone costly physical investments in network capacity [9]. Therefore, in this scenario, DNOs have two different options to deal with the increasing load growth: network reinforcements or DSM contribution options. DSM options can be considered as part of operation cost of DNOs as we assume that consumers are financially compensated for changing their behaviors to reduce peak loads.

6.1.2. Single-objective & multi-objective optimization

Beside investment costs, DNOs need to take into account other factors when solving DNEP, e.g., energy losses and electricity supply interruptions. Traditionally, expansion plans of low investment costs are often favored but, in transition toward a sustainable energy future, energy-efficient systems should be considered because of the increasing importance of environmental issues. It is also reported that outages due to faults on medium-voltage distribution networks have the largest contribution to the total System Average Interruption Duration Index (SAIDI) [11]. Such index reflects the reliability of the networks and also customer satisfaction, which are important for DNOs to pay attention to, especially in deregulated energy markets.

It can be seen that in order to draw out an appropriate solution plan for DNEP, DNOs have to consider many criteria, such as asset investment cost, DSM contribution cost, energy loss, and network reliability. One common approach to tackle this multi-criteria problem is to capitalize and aggregate all criteria into a single objective cost function [3]. Available single-objective (SO) optimization algorithms

can then be employed to minimize this integral cost. With this approach, DNOs are however limited to focus only on the financial perspective. Because SO optimizations can only return solutions of minimum cost, they cannot consider other alternatives if they would like to recognize what kinds of compromise on environmental issues and network reliability would need to be made for carrying out the cheapest expansion plan. Furthermore, it is not always possible to monetize non-financial terms in a sensible way. For example, conversion of the energy losses of an expansion plan into money requires assumptions about energy prices in the distant future and, in addition, the importance of the environment is hard to monetize. Similarly, capitalization of network reliability requires assumptions about penalty amounts that DNOs pay to consumers when electricity supply interruptions happen due to network failures. These assumptions are very likely to fluctuate during a long-term planning period and are difficult to be determined at the beginning. Moreover, it can be argued that even if a monetary conversion is possible, the loss of direct insights into the actual non-monetary objective values and the trade-offs is undesirable. Instead of the aggregation approach by capitalization, we therefore argue that it is more beneficial to keep these criteria separately and treat them as different objectives when solving DNEP.

These objectives are often conflicting with each other. For example, focusing on investment cost reductions often results in lossy and less reliable systems [3]. Another example is the choice of asset installations or DSM contributions or some combinations of both of them to deal with peak load growth. Delaying assets installations (i.e., reducing investment costs) requires higher contributions of DNOs to DSM options (i.e., increasing operation costs) to keep peak loads within assets' nominal capacities. In financial terms, both are costs for DNOs, but in essence, they are two conflicting directions: physical asset investments versus intangible investments (i.e., DSM policies). An ideal solution that optimizes all these objectives at the same time does not exist. Instead, there exists a *set* of so-called Pareto-optimal solutions [12] which are optimal in the sense that no solution can be improved on some objective without diminishing other objectives. So, they can be seen as optimal trade-off alternatives. By inspecting and comparing these trade-offs, DNOs can get insights in multiple aspects when making decisions and subsequently choose the Pareto-optimal solution that they prefer. For example, DNOs can immediately see how much energy will be lost during the planning period in case of the lowest-cost solutions and what the trade-offs between losses and cost are. DNOs may also recognize how many (extra) new assets must be installed to (further) improve the network reliability. It is beneficial if DNOs are exposed to these alternatives before choosing a concrete expansion plan.

6.1.3. Multi-objective optimization algorithms

Classical approaches

Earlier approaches to find multiple alternatives in MO optimization share the same working mechanism: 1) using a conversion procedure to change the MO optimization problem (MOOP) instance at hand to some SO optimization problem (SOOP) instance, 2) employing an available SO algorithm to solve that SOOP instance to

obtain one solution, 3) updating the conversion procedure with new conversion parameters, 4) going back to step 1 to begin a new optimization round with a new SOOP instance, or stop if having obtained enough solutions [12]. Two popular conversion procedures are: the weighted sum method and the ϵ-constraint method [12, 13]. The weighted sum method scales each objective with a coefficient and then adds all the scaled objectives into a composite objective function [12]. The conversion parameters of the weighted sum method are the coefficients. The weighted sum method cannot reach solutions on concave parts (if these exist) of the Pareto-optimal front [12, 14]. The ϵ-constraint method keeps one objective as the master objective and converts other objectives into problem constraints by employing an ϵ vector as upper bounds for the other objective functions [12]. The conversion parameters of ϵ-constraint method are the selection of master objective and and the ϵ vector. These classical approaches are time-consuming and inconvenient because multiple optimization processes must be executed [13]. It can be difficult for practitioners to properly choose good conversion parameters in order to obtain a good non-dominated front, which are problem-dependent and are often obtained from in-depth analysis. In modern MO optimizations, it is preferred that algorithms require parameter inputs from users as little as possible and are able to return a set of multiple non-dominated solutions in a single optimization run that well-approximates the Pareto-optimal front [12]. Applications of MO optimization in power and energy literature are currently indeed gradually evolving from classical approaches [15, 16] with single-objective reformulation to *true* multi-objective approaches [13], especially multi-objective evolutionary algorithms.

Multi-objective evolutionary algorithms

Evolutionary algorithms (EAs) have been shown to be a favorable methodology for industrial optimization due to their relative ease of implementation and excellent results for many applications. Moreover, EAs are population-based algorithms, in which a population of different candidate solutions are maintained and evolved. This makes EAs well-suited to meet the requirement of obtaining a diverse set of trade-off solutions in MO optimization [12]. Two exemplary multi-objective evolutionary algorithms (MOEAs), Non-dominated Sorting Genetic Algorithm II (NSGA-II [17]) and Strength Pareto Evolutionary Algorithm 2 (SPEA2 [18]) have found their applications in numerous power system planning and operation problems [3, 19–22]. However, theoretical research in MO optimization [23] pointed out the main drawback of classical MOEAs (e.g., NSGA-II and SPEA2), namely scalability. An MOEA is regarded to be scalable if it can maintain its effectiveness and efficiency when the problem size increases, resulting in acceptable, i.e., low-order polynomial, increasing runtimes. In the context of DNEP, scalability requires that a good set of trade-off expansion plans for large networks can be obtained by MOEAs within a reasonable amount of computing time. MOEA literature [23–26] pinpoints essential components to construct such scalable MOEAs. Following these design guidelines, in Chapter 3, we developed the multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm (MO-GOMEA). MO-GOMEA was found to have superior performance in solving laboratory benchmarks and to have very promising results when solving the real-world static DNEP problem [1] compared to the classical MOEA NSGA-II.

Therefore, we choose the MO-GOMEA to tackle the multi-objective dynamic DNEP problem.

6.1.4. Our contributions

In this chapter, we consider the dynamic DNEP problem with multiple objectives (MO-DNEP). We incorporate the future options for DSM as an additional investment action for DNOs. We present efficient trade-off solutions for this MO-DNEP problem and discuss their implications on DNEP decision making. We will show that MO-GOMEA can be employed to assist DNOs in designing effective expansion plans regarding the upcoming introductions of smart grid technologies (e.g., DSM or storage systems) into the existing traditional electricity networks.

The remainder of this chapter is organized as follows. Section 6.2 proposes how the DSM can be modeled into the DNEP problem. Section 6.3 formulates the MO-DNEP problem and introduces various objectives of interest. Section 6.4 presents the benchmark networks and experiment setup for the solver MO-GOMEA. Section 6.5 shows and discusses the experimental results on the benchmark networks. Finally, Section 6.6 concludes the chapter.

6.2. Expansion options

We assume that two categories of planning options are available to DNOs when solving DNEP: physical asset investment and DSM options. Asset investments (i.e., installing new facilities, upgrading existing equipment, or changing network topology) can increase the physical capacity of the distribution network. On the other hand, DSM policy contributions can be seen as improving the efficiency of network usage. DSM helps decrease the peak loads so that the current network capacity suffices to handle the corresponding power flow. In this section, we describe how these two options can be modeled together into the DNEP problem.

6.2.1. Network facilities (assets)

To represent the network assets of a candidate expansion plan for the multi-objective dynamic DNEP problem, we can employ the same encoding scheme that was proposed in Chapter 5 for the single-objective case. Let l be the total number of branches that can be considered for expansion planning, i.e., both existing cables and potential cable connections. A network configuration in the horizon year $t_{horizon}$ can be presented as a vector $\boldsymbol{x}$ of l integer elements:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_l), \quad |x_k| \in \Omega(k) \cup \{0\}, \quad k \in \{1, 2, \ldots, l\} \tag{6.1}$$

where $\Omega(k)$ is the set of cable types that can be installed at the k-th branch. The value of x_k indicates the status and the type of cable installed:

- $x_k = ID > 0$: A cable of type $ID \in \Omega(k)$ is installed.
- $x_k = 0$: No cable is installed at the k^{th} branch.
- $x_k = -ID < 0$: This is a normally-open point (NOP): a cable of type $ID \in \Omega(k)$ is installed but is out of normal operation. NOPs can be used to reconfigure the network during network failures.

While electric cables have very long lifetimes of about 30 years, planning periods longer than 30 years can be considered as impractical since there are many uncertainties at such distant horizons. Therefore, during a practical planning period, a network branch is normally reinforced at most once. A vector $\boldsymbol{x}$ of the network configuration in the horizon year suffices to represent an expansion plan (of 30 years). If $\boldsymbol{x}^0 = (x_1^0, x_2^0, \ldots, x_l^0)$ encodes the currently existing network, element-wise differences between $\boldsymbol{x}$ and $\boldsymbol{x}^0$ indicate which new assets should be installed to transform $\boldsymbol{x}^0$ to $\boldsymbol{x}$. To determine the installation year of each new asset for a feasible candidate solution, we employ the decomposition heuristic, that was proposed in Chapter 5), which can be summarized as follows. First, based on the forecast annual peak load growth rate, we determine the year t_X when the currently existing network becomes infeasible. Second, we create a base schedule by assuming that all new assets are installed in the same year t_X and evaluate this base schedule. Next, we loop through all these new assets in a random order and try to postpone each installation by one year if such delay yields a better expansion plan. This postponement checking is continued until further other delay is possible. Note that, in Chapter 5 regarding single-objective DNEP problems, an expansion plan can be considered as better than another plan in terms of a single objective (i.e., the investment cost or the total cost). For MO-DNEP problems, where multiple conflicting objectives are involved, the quality of expansion plans must be evaluated in terms of Pareto domination (see Section 6.3).

6.2.2. Demand side management (policies)

Parts of the power consumption from the network are flexible loads, such as the use of dishwashers, washing machines, tumble dryers, or charging electric vehicles. DSM policies can motivate consumers to shift these flexible loads to different times out of the daily peak power consumption hours by giving consumers, for example, financial compensations [27]. We assume that DNOs, as a stakeholder in energy markets, might be allowed (or required) to contribute parts of those compensations.

In this study, we assume that the flexible loads account for 10% of the peak loads at the beginning of a planning period, and linearly grow to 30% of peak loads at the end of the planning period. This assumption is based on the fact that the emergence of smart household appliances gradually enlarges the magnitude of flexible loads. We also assume that DNOs can contribute to DSM policies through a financial means in EUR/year for each peak power consumption reduction of 1 kW on an MV node in the whole year. Note that the numbers that we use here are simplified to set up a demonstration case. Different and more fine-grained scenarios can be created by customizing these assumptions as input data. In this study, we take into account only the peak shaving effects of DSM because this is the most important aspect to DNEP while other related details, such as the actual mechanism of DSM, who administers DSM, and how consumers are incentivized to participate in DSM, are abstracted away.

Demand side management representation

Unlike (long lifetime) physical assets, DSM options can be seen as operational policies which can be changed from year to year during the planning period. In principle, in order to represent a DSM policy for a planning period of multiple years, we need a DSM decision variable for each year. However, in this study, we use a DSM strategy which applies peak-shaving and only when it is necessary to prevent bottlenecks. Therefore, we only need DSM decision variables for the years from the first infeasible year t_X until (and including) $t_{horizon}$. We have $n_d = t_{horizon} - t_X + 1$ is the number of years in a DSM policy. We represent a DSM policy over n_d years as a vector of n_d non-negative integer elements.

$$\boldsymbol{y} = (y^1, y^2, \ldots, y^{n_d}),\ y^t \in \mathbb{D},\ t = 1, 2, \ldots, n_d \tag{6.2}$$

where $\mathbb{D}$ is the set of DSM option levels that are available to implement. The value of y^t indicates the chosen DSM level in year t. For the sake of simplicity in making decisions, DNOs are assumed to decide the amount of DSM contribution (i.e., corresponding with the desired amount of flexible load reduction) in discrete DSM levels, namely 0 (no DSM is needed), 1 (25% flexible load reduction), 2 (50% flexible load reduction), 3 (75% flexible load reduction) and 4 (100% flexible load reduction). We then have $\mathbb{D} = \{0, 1, 2, 3, 4\}$. Here, we take a holistic approach in which a single DSM level y^t is applied to all network nodes (i.e., MV/LV substations and MV customer stations) in each year t of the planning period. Our formulation can be easily extended to support a more fine-grained approach in which each suitable network node i has its own DSM level per year y_i^t as in the case of battery storage systems considered in Chapter 5.

Effects of DSM on peak loads

Let f^t denote the percentage of flexible loads in peak power consumption in a year t. We assume that $f^0 = 10\%$ in the beginning year and $f^{horizon} = 30\%$ in the end year of the planning period. The values of f^t's, $0 < t < t_{horizon}$ can be calculated by linear interpolation.

For an MV-D network of n nodes, assuming that no DSM policy is used, the vector of peak active power demand at each node in year t is:

$$\boldsymbol{P}^{t,0} = (P_1^{t,0}, P_2^{t,0}, \ldots, P_n^{t,0}) \tag{6.3}$$

The accompanying vector of peak reactive power is:

$$\boldsymbol{Q}^{t,0} = (Q_1^{t,0}, Q_2^{t,0}, \ldots, Q_n^{t,0}) \tag{6.4}$$

In that year t, if we use a DSM policy at a level $y^t > 0, y^t = 1, 2, 3, 4$, then the vector of new peak active power demands will be

$$\boldsymbol{P}^{t,y^t} = (P_1^{t,y^t}, P_2^{t,y^t}, \ldots, P_n^{t,y^t}) \tag{6.5}$$

where $P_i^{t,y^t} = P_i^{t,0} * (1 - (y^t/4) * f^t)$, so that $y^t = 1, 2, 3, 4$ corresponds to $25\%, 50\%$, $75\%, 100\%$ flexible load reduction, respectively. And the corresponding vector of peak reactive power will be:

$$\boldsymbol{Q}^{t,y^t} = (Q_1^{t,y^t}, Q_2^{t,y^t}, \ldots, Q_n^{t,y^t}) \tag{6.6}$$

where $Q_i^{t,y^t} = Q_i^{t,0} * (1 - (y^t/4) * f^t)$. Then, $\boldsymbol{P}^{t,y^t}$ and $\boldsymbol{Q}^{t,y^t}$ form the new load due to the peak shaving effect of the DSM level y^t in the year t.

The total reduction in peak active power demands corresponds with employing the DSM level y in year t is:

$$P_{total}^{t,y^t} = \sum_{i=1}^{n} (P_i^{t,0} - P_i^{t,y^t}) \tag{6.7}$$

6.2.3. Solution representation

A solution $\boldsymbol{s}$ of the DNEP problem for a network over a planning period consists of the network configuration $\boldsymbol{x}^{horizon}$ at the end of the planning period and the DSM policy $\boldsymbol{y}$ (i.e., the list of DSM levels that are applied on the network in each year). For the sake of convenience in representation, we shorten $\boldsymbol{x}^{horizon}$ as $\boldsymbol{x}$.

$$\boldsymbol{s} = (\boldsymbol{x}, \boldsymbol{y}) = (x_1, x_2, \ldots, x_{n_l}, y^1, y^2, \ldots, y^{n_d}) \tag{6.8}$$

The total number of decision variables is then $n_l + n_d$. An installation schedule of new physical assets in $\boldsymbol{x}$ is determined by our installation decomposition approach (see Section 6.2.1). The DSM policy $\boldsymbol{y}$ indicates the DSM levels that affect the peak load profile $\boldsymbol{P}^{t,y^t}$ and $\boldsymbol{Q}^{t,y^t}$ in each year t (see section 6.2.2), and will be used to verify constraint violations of the corresponding network configuration in year t.

6.3. Problem formulation

6.3.1. Constraints

For a network $\boldsymbol{s}$ of n nodes and l total cable connections, in each year t during the planning period, given the forecast peak consumption at every node and the peak shaving effect of the chosen DSM policy y^t in that year, we can determine the peak load profile $\boldsymbol{P}^{t,y^t}$ and $\boldsymbol{Q}^{t,y^t}$. The same constraints as in the case of single-objective DNEP problems (see Section 4.2.2) must be satisfied regarding the peak loads in each year $\boldsymbol{P}^{t,y^t}$ and $\boldsymbol{Q}^{t,y^t}$ (which take the DSM policy into account):

1. Connectivity: All nodes are connected to the network.
2. Normal operation constraints: The voltage at each node is within allowable ranges (i.e., $0.9 * V_i^{nom} \leq V_i \leq 1.1 * V_i^{nom},\ i \in \{1, 2, \ldots, n\}$) and the magnitude of the power flow through each cable is within the nominal capacity of the that cable (i.e., $|S_i| \leq S_i^{nom},\ i \in \{1, 2, \ldots, l\}$).
3. Radiality constraint: The network operates radially due to the positions of NOPs.
4. Reconfigurability constraint: When a cable fails, the DNO can isolate the troubled cable and reconfigure the network by closing NOPs to bring back operation. While maintenance activities take place, a mild overload of 130% nominal capacity is allowed and the radiality constraint can be compromised.

5. Substation capacity constraint: We assume at most three new outgoing cables can be installed at each MV transmission substation (or HV/MV transformer substation) due to limited physical space.

Evaluations of constraints 2 and 4 require computationally expensive alternating current power flow calculations (AC-PFCs [28]) for the peak load profile $\boldsymbol{P}^{t,y^t}$ and $\boldsymbol{Q}^{t,y^t}$. As discussed in Section 4.2.2, instead of performing l AC-PFCs for a complete verification of the reconfigurability constraint, for the sake of efficiency, the Line Outage Distribution Factor (LODF [29]) can be employed, in which only one (pre-contingency) AC-PFC is needed. Note that because PFCs can only be performed for connected networks, the connectivity constraint needs to be satisfied so that constraints 2 and 4 can be properly verified. If the network configuration $\boldsymbol{x}$ of a candidate solution $\boldsymbol{s}$ is unconnected, constraints 2-5 will not be evaluated. Instead, the disconnectivity quantification (DQ) procedure is invoked to measure the difference in terms of network topology between $\boldsymbol{x}$ and the currently existing network $\boldsymbol{x}^0$ (see Section 4.2.2 for more details and pseudo-code). The obtained result is then used as the constraint violation value of that candidate solution $\boldsymbol{s}$.

To compare different candidate solutions of a multi-objective DNEP problem, we employ the constraint-domination principle [30] in combination with their disconnectivity values. For example, we compare two candidate solutions $\boldsymbol{s} = (\boldsymbol{x}, \boldsymbol{y})$ and $\boldsymbol{s}' = (\boldsymbol{x}', \boldsymbol{y}')$. If $\boldsymbol{x}$ is connected and $\boldsymbol{x}'$ is unconnected, then $\boldsymbol{s}$ is better than $\boldsymbol{s}'$. If both networks $\boldsymbol{x}$ and $\boldsymbol{x}'$ are unconnected, then the one with a smaller disconnectivity is considered to be the better one. If both networks $\boldsymbol{x}$ and $\boldsymbol{x}'$ are connected, we then compare $\boldsymbol{s}$ and $\boldsymbol{s}'$ by using their total violation values of other constraints (i.e., constraint 2-5). Feasible solutions have a total violation of value 0, and a feasible solution is always better than an infeasible solution. If both solutions $\boldsymbol{s}$ and $\boldsymbol{s}'$ are infeasible, then the one that has greater total constraint violations is the worse one. If both solutions $\boldsymbol{s}$ and $\boldsymbol{s}'$ are feasible, we then evaluate their objective values and use the Pareto domination principle to compare them.

6.3.2. Objectives

Pareto domination

A feasible solution $\boldsymbol{s}$ of a multi-objective optimization problem is said to *Pareto dominate* (i.e., to be multi-objectively better than) another feasible solution $\boldsymbol{s}'$ (denoted $\boldsymbol{s} \succ \boldsymbol{s}'$) if $\boldsymbol{s}$ is strictly better than $\boldsymbol{s}'$ in at least one objective and $\boldsymbol{s}$ is no worse than $\boldsymbol{s}'$ in all the other objectives. A non-dominated set $\boldsymbol{\mathcal{P}}$ is a set of solutions in which no solution dominates any other solutions that also exist in $\boldsymbol{\mathcal{P}}$. A solution $\boldsymbol{s}$ is said to be *Pareto optimal* if and only if there exists no other solution $\boldsymbol{s}'$ that dominates $\boldsymbol{s}$. The Pareto-optimal set $\boldsymbol{\mathcal{P}_S}$ is the set of all possible Pareto-optimal solutions. The Pareto-optimal front $\boldsymbol{\mathcal{P}_F}$ is the image set of $\boldsymbol{\mathcal{P}_S}$ in the objective space. Because the size of $\boldsymbol{\mathcal{P}_S}$ is usually very large or even infinite, it is very hard to find all Pareto-optimal solutions. MO optimization normally aims to obtain a good non-dominated set $\boldsymbol{\mathcal{P}}$ whose front (i.e., the image set in the objective space) approximates $\boldsymbol{\mathcal{P}_F}$. The approximation quality is evaluated in the objective space because we generally want a diverse set of good solutions with respect to all objectives.

In the following section, we introduce the typical objectives that can be used for MO-DNEP problems.

Physical asset installation cost CAPEX
Minimizing the asset investment cost CAPEX has been discussed in Chapters 4 and 5. The net present value (NPV) of the total CAPEX of an expansion plan over a planning period is formulated by Equation 4.5 in Section 4.2.3. The installation schedule of new assets in the expansion plan is determined by the decomposition heuristic presented in Section 5.2.2 and Section 6.3.3.

Demand side management cost
From Equation 6.7, we can obtain P_{total}^{t,y^t}, i.e., the total reduction of peak active power consumption due to the application of DSM level y^t on the network in year t. The cost that the DNO spends on DSM level y in year t is then computed based on this reduction as:

$$DSM(t) = Price_{unit}^{DSM} * P_{total}^{t,y^t} \tag{6.9}$$

where $Price_{unit}^{DSM} = 1 €$ is the unit price for power consumption reduction per kW. For the purpose of demonstration, we here choose only a single DSM price but this price can be customized as input data to create different scenarios of DSM costs.

We minimize the NPV of the total DSM policy cost over the planning period with a discount rate i.

$$DSM_{NPV} = \sum_{t=t_X}^{t_{horizon}} \frac{DSM(t)}{(1+i)^{t-t_0}} \tag{6.10}$$

We here assume that the DNO would start to apply DSM options from the first year t_X that the current network would become infeasible. From t_0 until t_X, the current network can still operate properly, and the DNO has little practical incentive to employ DSM earlier than needed.

Energy loss
Since the network cable is our main asset category in this chapter, the energy loss of the network in year t can be taken as:

$$E_{loss}(t) = P_{peak\ loss}(t) \times T_{loss}(t) \tag{6.11}$$

where $P_{peak\ loss}(t)$ is the total network peak loss which can be obtained by performing PFC regarding peak demands $\boldsymbol{P}^{t,y^t}$ and $\boldsymbol{Q}^{t,y^t}$, i.e., the peak load profile in year t with respect to a chosen DSM level y. $T_{loss}(t)$ is the service time of peak loss for year t, defined as the ratio of the area under the normalized yearly loss profile shape over the normalized peak loss value [31, 32]. The exact values of T_{loss} depend on the nature of each specific network and its power demands. In this chapter, we take $T_{loss}(t) = 2000$ hours for all t, which has been reported to be a realistic value for real-world MV distribution cables [32]. Note that this value was calculated based on

traditional DNEP [32], but for the purpose of demonstration in this study, this value is sufficient. More accurate and dedicate models to calculate T_{loss} in the presence of smart grid technologies, when available, can be employed and be straightforwardly considered as input data.

We minimize the total energy losses on the network during the planning period:

$$E_{loss} = \sum_{t=t_0}^{t_{horizon}} E_{loss}(t) \tag{6.12}$$

Customer minutes lost (CML) per year

We choose to evaluate the total number of customer minutes lost (CML) per year to quantify the reliability of an MV-D network. Alternatively, we can also measure the network reliability by employing the System Average Interruption Duration Index (SAIDI), which can be easily obtained by dividing the total CML per year over the number of customers in the network [11]. When a network cable fails, the feeder containing that cable (i.e., from an MV transmission substation, or HV/MV transformer substation, with circuit breakers to corresponding NOPs) is put out of operation because the corresponding circuit breaker will be triggered. Customers connected to all the nodes along this feeder suffer a temporary power outage. The DNO needs to find out and isolate the failed cable along the feeder. After the failed cable is isolated, the DNO can close the corresponding NOP and the circuit breaker of the feeder so that the electricity supply is resumed. The number of failures NF_k over a cable k per year can be estimated as:

$$NF_k = F_k \cdot L_k \tag{6.13}$$

where F_k is the annual failure rate of cable k per kilometer and L_k is the length of cable k. The restoration time T_{res} can be defined as the duration between failure occurrence and power supply restoration. This duration depends on the number of nodes connected to the feeder. The average restoration time (in minutes) [33] when a cable k fails can be taken as

$$T_{res}(k) = 75 + \frac{NS(Feeder(k))}{2} \cdot 10 \tag{6.14}$$

where $Feeder(k)$ denotes the feeder containing the cable k and $NS(Feeder(k))$ is the number of MV/LV transformer substations and MV customer substations connected to the feeder $Feeder(k)$. The CML for a cable k per year can now be defined as follows:

$$CML_k = NF_k \cdot NC(Feeder(k)) \cdot T_{res}(k) \tag{6.15}$$

where $NC(Feeder(k))$ is the total number of customers connected along the feeder $Feeder(k)$. In this study, when conducting experiments, we assume that the number of customers stays the same during the planning period. More accurate prediction models about the number of customers, if available, can be straightforwardly employed to provide new input data.

By the application of the asset installation decomposition procedure (see Sections 6.2.1 and 5.2) on the final network configuration $\boldsymbol{x}^{horizon}$, we obtain an installation schedule from which we can derive the network configuration $\boldsymbol{x}^t$ in each year t. The CML of a network $\boldsymbol{x}^t$ in a year t can be computed as:

$$CML(\boldsymbol{x}^t) = \sum_{\substack{k=0 \\ x_k^t > 0}}^{l} CML_k \tag{6.16}$$

where l is the total number of cable connections, and we compute CMLs only for active branches ($x_k^t > 0$) as there is no power flow through a cable with an NOP ($x_k^t < 0$). The total CML during the planning period is:

$$CML_{total} = \sum_{t=t_0}^{t_{horizon}} CML(\boldsymbol{x}^t) \tag{6.17}$$

We minimize the averaged CML:

$$CML_{averaged} = \frac{CML_{total}}{n_y} \tag{6.18}$$

where $n_y = t_{horizon} - t_0 + 1$ is the number of years in the planning period.

6.3.3. Solution evaluation

The objectives CAPEX, DSM cost, and averaged CML per year for a feasible solution can be efficiently evaluated by checking the network topology in each year along the planning period. The total energy loss objective, the normal operation constraint and the reconfigurability constraint, however require computationally-intensive PFCs. Evaluating solutions for large networks in every year during a planning period of 30 years could take a prohibitively long computing time. Moreover, in the current practice of real-world DNEP, DNOs hardly ever add a new cable to a network every year. Adding a new cable can increase the network capacity by a great amount and is normally very expensive. Therefore, instead of evaluating solution constraints and performing the decomposition heuristic in a fine-grained year-by-year manner as presented above, we run experiments with a stage-by-stage approach as in Section 5.3.2, where each stage covers a period of at most 5 years and at least 3 years. Thus, the decomposition heuristic will try to delay each asset installation by one stage of multiple years. New assets in a stage are assumed to be installed in the first year of that stage. For the sake of simplicity and computing time reasons, we also assume that DNOs can make decisions about DSM options per stage, instead of per year as in the formulation in Section 6.2.2. Therefore, the number of DSM variables n_d is equal to the number of stages. For every year in a stage, the same DSM level is applied. The peak load of the final year in each stage is considered to be the peak load of the stage (because peak loads normally increase monotonically if network configuration and DSM policy remain the same during the stage), which is used in constraint evaluations of the network configuration in that

stage. Note that the same DSM level can have different costs per year, depending on the amount of peak load reduction in that year. Although we perform stage-by-stage planning, the objective values of CAPEX, DSM cost, and CML can still be evaluated in a year-by-year manner. As we perform PFCs only for the final year of each stage, we can obtain the accurate value of peak loss only in that year. For the remaining years in the stage, peak losses (to be used for computing energy losses as in Equation 6.11) can be estimated by using the assumption that the peak loss also has a growth rate related to the load growth R as follows [6]:

$$P_{peak\ loss}(t) = P_{peak\ loss}(t-1) * (1+R)^2 \tag{6.19}$$

6.4. Experiment setup

6.4.1. Benchmark networks

We employ the same three distribution networks in Chapter 4 as our benchmark networks here, but some network details are modified for the purpose of demonstration. Details about the current peak loads of the three networks are listed in the Appendix. Details about the planning period year and annual peak load growth rate for each network are shown in Table 6.1. Based on the forecast peak load growth rate, we calculate the first year t_X that each network becomes infeasible. The values of t_X are needed for the decomposition heuristic to determine *when* each new asset should be installed along the planning period (i.e., from t_X until $t_{horizon}$). Note that as mentioned in Section 6.3.3, due to computing time reasons, instead of the more fine-grained year-by-year approach, we perform the expansion planning in a stage-by-stage manner, where each stage covers a period of maximum 5 years and minimum 3 years, which are reasonable stage lengths in DNEP practice. Details about the number of stages and the beginning year and end year of each stage are shown in Table 6.1.

Table 6.1: Planning periods of Network 1, 2, and 3. $t_{horizon}$ is the horizon year. R is the peak load growth rate. t_X is the first year in which the network becomes infeasible. Stage S_i is the duration of each planning stage during $[t_X - t_{horizon}]$. n_d is the number of DSM variables.

ID	$t_{horizon}$	R	t_X	Stages $S_i = [t_{begin} - t_{end}]$	n_d
1	30	2%	8	S_1=[8 - 10];S_2=[11 - 15];S_3=[16 - 20];S_4=[21 - 25];S_5=[26 - 30]	5
2	30	2%	10	S_1=[10 - 12];S_2=[13 - 15];S_3=[16 - 20];S_4=[21 - 25];S_5=[26 - 30]	5
3	15	3%	3	S_1=[3 - 5];S_2=[6 - 10];S_3=[11 - 15]	3

We use different combinations of four objectives presented in Section 6.3.2 to create three MO-DNEP problems: CAPEX versus DSM, total cost (CAPEX+DSM) versus total energy losses, and total cost (CAPEX+DSM) versus CML per year. Note that multi-objective optimization with three or four objectives is also possible but would not necessary yield more illustrative results.

6.4.2. Multi-objective GOMEA

We employ the parameter-less MO-GOMEA as the solver for tackling our multi-objective DNEP problems. The details about MO-GOMEA are presented in Chap-

ter 3. Here, we outline the operation of MO-GOMEA when employed to solve MO-DNEP problems.

First, the distribution network generation procedure (see Section 4.4) is used to initialize the populations of MO-GOMEA with candidate solutions having *good* topologies. These initial networks are randomly constructed in such a way that the connectivity, radiality, and substation capacity constraints are satisfied. Section 4.4 showed that initializing EA solvers' populations with good initial solutions can considerably enhance their performance. In every generation, MO-GOMEA partitions its working population into a number of (overlapping) clusters by performing the balanced k-leader-means clustering algorithm in the objective space. Candidate solutions in the same cluster can be considered as expansion plans that are relatively similar to each other in terms of their objective value vectors. For each cluster, a linkage model is learned to capture the specific dependency structures among decision variables of the candidate solutions in that cluster. Next, the Gene-pool Optimal Mixing (GOM) variation operator uses the learned linkage model to transform each existing solution $\boldsymbol{s}$ into a new candidate solution $\boldsymbol{s}'$ in a step-wise manner. In each step, $\boldsymbol{s}$ is mixed with a donor solution $\boldsymbol{d}$ randomly selected from the same cluster that $\boldsymbol{s}$ belongs to. The mixing event commits if the partially-altered intermediate solution improves its previous state; otherwise, the mixing is reversed. Note that the solution improvement is assessed based on the Pareto-domination principle if $\boldsymbol{s}$ belongs to a *middle*-region cluster in which constituent solutions tries to optimize all objectives at the same time. On the other hand, if $\boldsymbol{s}$ belongs to an *extreme*-region cluster whose constituent solutions excel in a specific objective, the solution improvement checking is in the single-objective manner. Figure 6.1 illustrates the operation of MO-GOMEA in solving the MO-DNEP problem that involves two conflicting objectives: minimizing the investment cost and minimizing energy losses. The middle-region clusters contain expansion plans that attempt to exploit both objectives simultaneously. The two extreme-region clusters consist of expansion plans that are skewed toward minimizing either investment cost or energy loss, respectively. Note that, because we employ the parameter-less MO-GOMEA, apart from assigning a computing time budget, we are exempt from all parameter tuning and settings. The result obtained at the end of the optimization process (when the computing budget is used up) is a diverse set of expansion plans that exhibit possible trade-offs between the investment cost and the energy loss.

The variation operator (VO) of MO-GOMEA needs some adaptations to work with DNEP problems. We consider two VOs that have been proposed in Chapter 4 and Chapter 5 for DNEP problems in the single-objective case: Disconnectivity Quantification (DQ) and Branch Exchange with mutation (BX-M). MO-GOMEA with DQ can be considered as an out-of-the-box variant since the disconnectivity quantification is itself included in the constraint evaluation for candidate solutions (see Section 4.2.2). BX-M is a VO that uses the DNEP problem-specific knowledge of distribution networks to ensure that new candidate solutions are created with connected and radial topologies. BX-M has been shown in Chapter 4 and Chapter 5 to be the best VO for single-objective DNEP problems. By experimenting and comparing the problem-customized solver MO-GOMEA BX-M with an out-of-the-

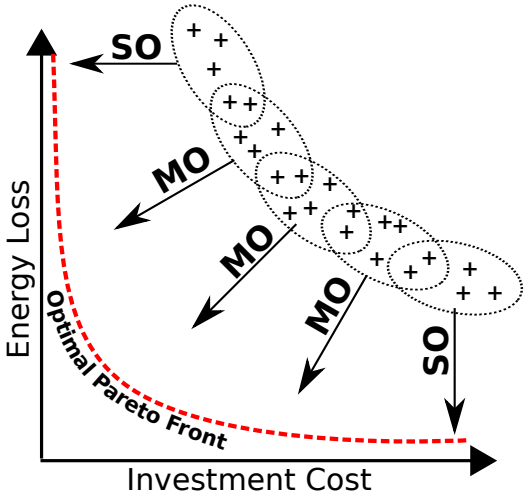

Figure 6.1: Different clusters in MO-GOMEA approach different parts of the Pareto-optimal front. Two *extreme*-region clusters employ single-objective optimization (SO) while *middle*-region clusters employ multi-objective optimization (MO).

box variant (i.e., MO-GOMEA DQ), we can separate the contribution of the domain knowledge from the overall results and then precisely assess the true performance of MO-GOMEA in solving MO-DNEP problems.

Each MO-GOMEA variant is run 10 times independently on each benchmark network proposed in Section 6.4.1. For each problem instance, the final non-dominated front is assembled by combining all results of 10 runs and is considered as the optimization result. Because the parameter-less MO-GOMEA is used, apart from assigning a computing budget, we are exempt from all parameter tuning and settings. The maximum number of candidate expansion plan evaluations as the computing budget in each run is 100,000 evaluations for Network 1, 200,000 for Network 2, and 300,000 for Network 3. Note that each solution evaluation consists of performing PFCs to verify the feasibility of the corresponding network configuration in each stage during the planning period. The experimental results for each problem instance are presented in the following section.

6.5. Experimental Results

6.5.1. CAPEX vs. DSM

To manage the MV-D networks to handle the peak load growth, DNOs have two different strategies. DNOs can choose to deal with the situation by traditional network reinforcement activities: installing new assets into the networks along the planning period. The network capacity is then increased. On the other hand, DNOs can actively promote DSM policies so that more and more flexible loads can be shifted out of peak power consumption hours, and the peak load can then be reduced to be kept within the current network capacity. The efficiency of using the

current network capacity is thereby improved.

Table 6.2: Solutions of DNEP considering CAPEX vs. DSM

Extreme Solution	CAPEX (EUR)€	DSM (EUR)€
NETWORK 1		
Most Asset-pro	40177	0
Most DSM-pro	12437	5697
NETWORK 2		
Most Asset-pro	23285	0
Most DSM-pro	533	19290
NETWORK 3		
Most Asset-pro	35465	0
Most DSM-pro	0	25410

Table 6.2 shows the extreme solutions when solving DNEP to find appropriate solution plans to deal with the forecast load growth with respect to two objectives: minimizing cost of installing new assets and minimizing cost of DSM contributions. It can be seen that for any network, DNOs can simply follow the traditional capacity expansion and not participate in DSM activities at all. A lot of new assets must then be installed to catch up with the surging peak load, especially with the imminent popularity of electric vehicles. DNOs can alternatively choose a more smart-grid-oriented solution: by participating in DSM activities to ensure that peak load would be kept under control. Table 6.2 shows that if DNOs invest in DSM policies, a great deal of new asset installation cost can be saved by being able to delay expensive network reinforcements to many years later. The combined cost of CAPEX+DSM of the most DSM-pro solutions are shown to be significantly less than the CAPEX of the traditional physical asset investment alone. Note that the costs of DSM options here are based on our assumption about the DSM price and different DSM price levels can be used to create multiple scenarios. We here use a single DSM price for the purpose of demonstration only. Interestingly, for Network 3 (i.e., the largest network considered in this study), with a good DSM policy investment, the network does not require any new asset installation during its planning period of 15 years. Only the NOPs need to be relocated to reconfigure the power flow paths and the cost of NOP relocations is generally negligible. Indeed, Figure 6.2 shows three example solution plans for Network 3. First, the DNO can actively stimulate DSM policies to shift all flexible loads and the network can be left intact for 15 years. Second, the DNO can choose a combination of new asset installations and DSM options but the asset installations are delayed until later years in the planning period. In the obtained outcome data, we observed that in the first stage, the network can be reconfigured by changing the locations of NOPs and new assets are only needed to be added in the last stage. Third, the peak load growth is not controlled and the DNO needs to install one new cable connection and replaces three other legacy cables. Note that it is not necessary that a DSM-pro solution is better than an asset-

pro solution. Which solution will be chosen to be carried out depends on specific situations. For Networks 1 and 2, there is no 0-CAPEX solution like the case of the most DSM-pro solution for Network 3. This is because we here perform experiments with a planning period of 30 years for Networks 1 and 2 while the planning period of Network 3 is 15 years. After 30 years, the peak power demands will be much higher such that even if 100% flexible load were to be reduced, the remaining base load would still exceed the currently available network capacity. Therefore, physical asset investments are still required in the most DSM-pro solutions for Networks 1 and 2.

Solving DNEP while considering CAPEX versus DSM results in many interesting alternatives (see Figure 6.3). Although both asset investments and DSM options can be considered as costs for DNOs, they are in essence two different strategies: adding more network capacity versus increasing efficiency in the usage of current network capacity by decreasing peak power consumption. It is important that DNOs are informed about all these alternatives before making a decision, and it is crucial that these alternatives must be (approximately) optimal trade-off solutions so that the best-informed decision can be made. Figure 6.3 shows the non-dominated fronts (i.e., the image set of the best found non-dominated solutions) found by two MO-GOMEA variants DQ and BX-M. While BX-M is a problem-specific variation operator, the non-dominated front found by MO-GOMEA BX-M is only slightly better than the one obtained by MO-GOMEA DQ. Therefore, DNOs can easily use MO-GOMEA out-of-the-box with some minor modifications (i.e., MO-GOMEA DQ) and still obtain reasonably good results. If more problem-specific knowledge is available, a DNEP-dedicate variation operator like BX-M can be constructed and incorporated into MO-GOMEA to find better results within the same computational budget.

6.5.2. CAPEX & DSM vs. Energy loss

We can combine the CAPEX and DSM contributions together to form the total cost of a DNEP solution plan. DNOs often need to know what is the most economical expansion plan, regardless of whether they are asset investments or DSM options. Minimizing the total cost alone (i.e., SO optimization) usually returns a network that uses thin cables of small diameters, which have higher energy losses compared to thicker cables. Energy loss is normally considered as a part of the operation cost OPEX of DNOs, and is usually capitalized to be aggregated with CAPEX so that SO optimization can be employed to find the solution that has the minimum lump sump cost. Being informed about only one solution plan, DNOs will find it difficult to reduce energy losses (in the most efficient way) if they would like to do so. Moreover, in transition toward a greener and environment-friendlier energy consumption future, DNOs will be motivated to consider energy-efficient networks. Here, we retain energy losses out of the total cost and try to optimize it separately by considering it as a separate objective. We have run MO-GOMEA DQ and MO-GOMEA BX-M for the two objectives: total cost and total energy losses. The results are as follows.

Table 6.3 shows the most economical and the most energy-efficient solution plans for DNEP for the three benchmark networks. It can be observed that the most

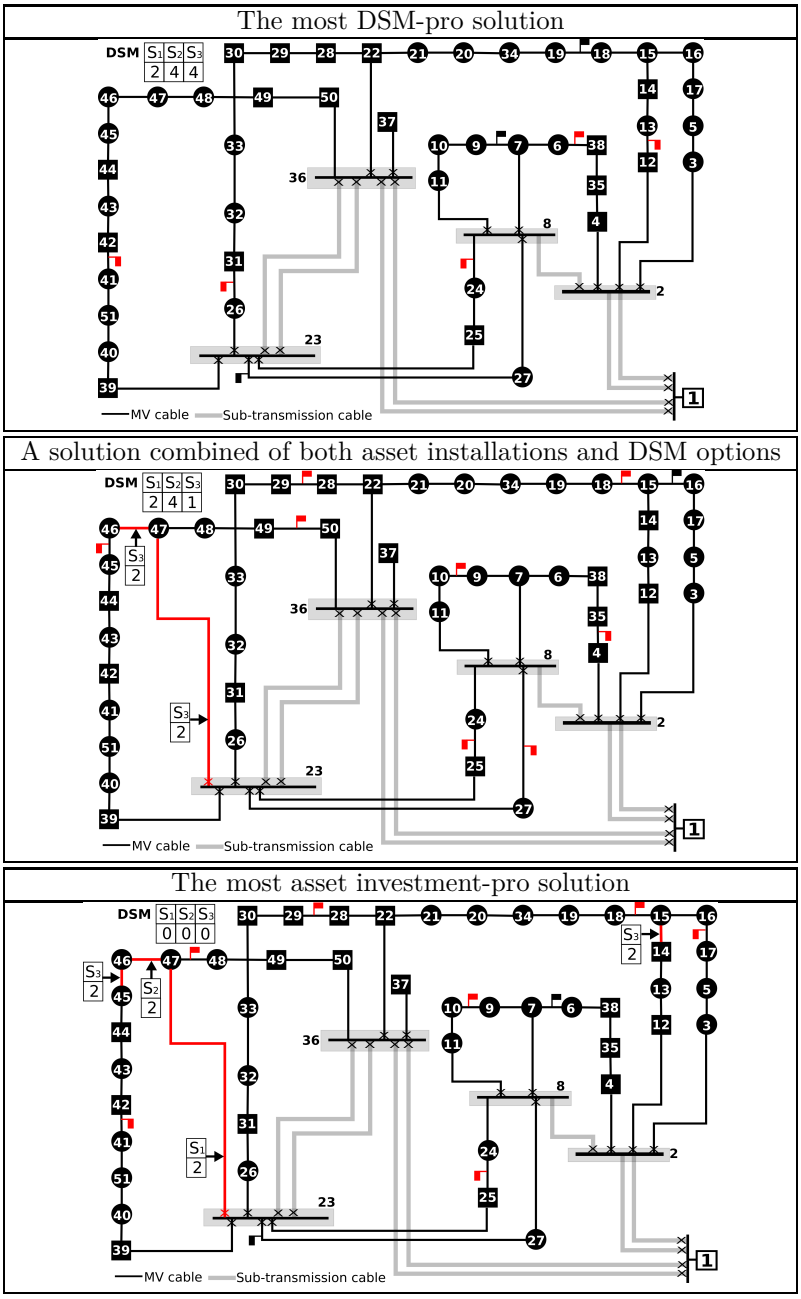

Figure 6.2: Network 3: CAPEX vs. DSM. Red-colored cables indicate new asset installations. Red flags are new positions of NOPs.

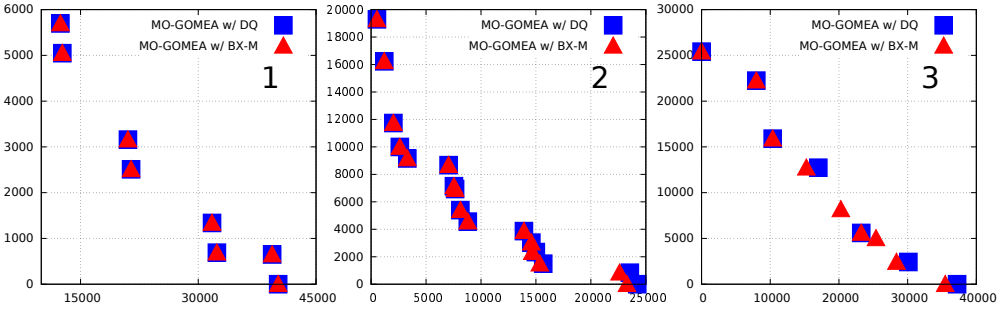

Figure 6.3: Non-dominated fronts of MO optimizations for three networks. **Horizontal axis**: Net Present Value of CAPEX (EUR). **Vertical axis**: Net Present Value of DSM (EUR). MO-GOMEA DQ is shown as blue squares. MO-GOMEA BX-M is shown as red triangles.

Table 6.3: Solutions of DNEP considering Total Cost (CAPEX + DSM) vs. Energy Losses

Extreme Solution	Total Cost (EUR) €	Energy Loss (MWh)
NETWORK 1		
Most economical	18372	2333
Most energy-efficient	327383	804
NETWORK 2		
Most economical	12451	4369
Most energy-efficient	279079	2105
NETWORK 3		
Most economical	25410	2263
Most energy-efficient	1273310	663

economical DNEP solutions are also the least energy-efficient solutions. If we do not take into account the specific locations of NOPs (because NOP relocations do not have a cost in our model), then, for Network 3, the most economical solution is the same as the most DSM-pro solution (compare Table 6.2 and Table 6.3). Because Network 3 has many legacy cables (i.e., cables of very small diameters) and no cable needs to be replaced in the most DSM-pro solution, its total energy loss is therefore the highest. In order to reduce energy losses, many of these legacy cables must be replaced with new cables of bigger sizes, bringing up the total investment cost. Table 6.3 also indicates that the cost of improving energy efficiency is very high. Therefore, instead of choosing the most efficient solution, it might be more reasonable for DNOs to look into solutions that balance cost and efficiency (i.e., the middle regions of the fronts, see Figure 6.5). Figure 6.4 shows three example solutions for Network 2. We can see that the most energy-efficient solution (i.e., the solution with the least energy losses) requires a lot of new cable installations and replacements because new cables of higher capacities generally have less branch resistance. This solution also makes use of the highest level of DSM options during the planning period because

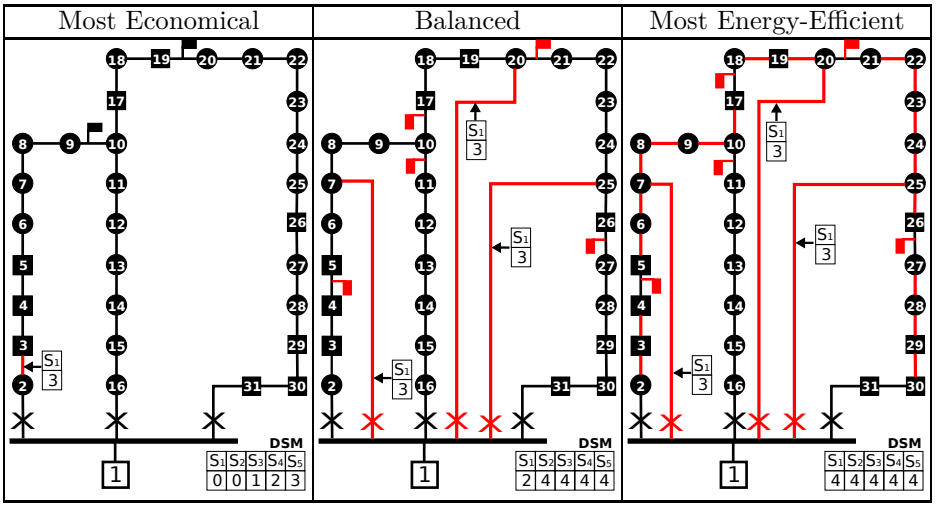

Figure 6.4: Network 2: Total cost (CAPEX+DSM) vs. Energy Loss. Red-colored cables indicate new asset installations. Red flags are new positions of NOPs.

reducing peak load can also reduce energy losses as well. The most economical solution replaces only one cable and has little participation in DSM contributions; this solutions suffers the highest total energy losses. A balanced solution might be a favorable combination of installing new cables and actively contributing to DSM options (with respect to our assumption about DSM price). All three solutions in Figure 6.4 indicate that all new asset installations should be done in the first stage, which is understandable as the sooner we replace legacy cables with new and more efficient cables the less energy will be lost.

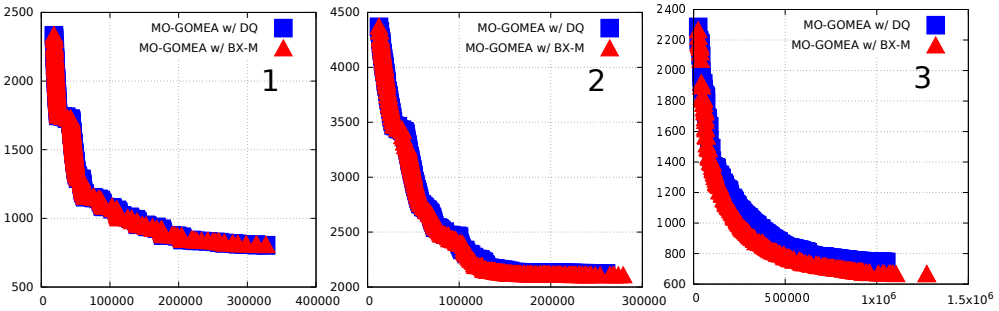

Figure 6.5: Non-dominated fronts of MO optimizations for three networks. **Horizontal axis**: Net Present Value of CAPEX+DSM (EUR). **Vertical axis**: Total Energy Losses (MWh). MO-GOMEA DQ is shown as blue squares. MO-GOMEA BX-M is shown as red triangles.

Figure 6.5 shows the non-dominated fronts of MO-GOMEA DQ and MO-GOMEA BX-M solving DNEP with respect to the total cost (CAPEX+DSM) and the total energy loss. For Network 1, the obtained results of the two variants are quite

similar. For Network 2, MO-GOMEA BX-M found a slightly better front than MO-GOMEA DQ. However, these differences are not critical, especially considering the numerous amount of solutions on the non-dominated fronts (see Figure 6.5), which is different from the much sparser fronts when considering CAPEX vs. DSM (see Figure 6.3). For Network 3, it can be seen more clearly that the solutions found by MO-GOMEA BX-M (Pareto) dominate those found by MO-GOMEA DQ. However, the gap between the DQ's non-dominated front and BX-M's non-dominated front is not large. If we run MO-GOMEA DQ longer, we can obtain results that are similar to MO-GOMEA BX-M's results but within the same amount of computing time, BX-M gives MO-GOMEA some slight advantages over DQ. Compared to the more DNEP problem-dedicated MO-GOMEA BX-M, the performance of the out-of-the-box MO-GOMEA DQ can be considered as quite robust. This suggests again, like in the single-objective case, that the linkage learning capability of MO-GOMEA is powerful enough to help the algorithm reach acceptably good solutions even when problem-specific knowledge (e.g., the way the connectivity constraint should be handled in BX-M) is not available.

6.5.3. CAPEX & DSM vs. CML

The averaged Customer Minutes Lost per year (CML) during the planning period is regarded as the measure for network reliability in this study. CML in our model depends on the distribution of customers along each feeder (i.e., the fewer customers connect to a feeder, the fewer customers would be affected if an outage occurs), on the number of MV nodes on that feeder (i.e., the fewer MV nodes, the less time is required to localize and isolate the failed cable), and also on the total length of the feeder (i.e., the shorter the feeder is, the smaller the failure rate is). Minimizing CML can add more new cable connections to the network (i.e., creating new feeders) so that each feeder connects fewer nodes and fewer customers, which increase the cost of asset installations. On the other hand, to minimize CAPEX, it is more economical to try to relocate existing NOPs first (rather than add new cables immediately) to reconfigure the network so that parts of power flows are re-routed through different paths, avoiding heavily-loaded cables. To this end, NOPs are usually located at locations that make the power flow equally distributed for each feeder. However, these positions might not be optimal locations to minimize CML. Minimizing CML tends to relocate NOPs so that the number of customers are distributed equally per feeder. An MV customer substation node and an MV/LV transformer substation node might have the same power demand but a customer substation is counted as one customer (i.e., one company) while a transformer substation can be counted as many customers (i.e., many households). Therefore, minimizing CAPEX and minimizing CML are two conflicting objectives.

Here, we also combine CAPEX and DSM together and regard that as the total cost of a solution plan. We have run MO-GOMEA DQ and MO-GOMEA BX-M for two objectives: total cost and CML per year. The results are as follows. Table 6.4 shows the extreme solutions for each benchmark network: the most economical solution (i.e., the least total cost) and the most reliable solution (i.e., the least CML). From Table 6.4 and Figure 6.7, it can be seen that the most economical solution is

also the least reliable one and vice versa. For Network 3, we observed that we again obtain the most economical solution that is similar to the most DSM-pro solution as in Figure 6.3 (not taking into account the NOP locations in each case). It can be inferred that due to the peak shaving effects of DSM options and the relocation of NOPs, no new asset installation is required during the planning period. However, those NOP locations are not the favorable positions to reduce CML. Figure 6.6 shows three examples for Network 1. The most economical solution promotes DSM options and replaces three overload cables at much later stages of the planning period. The most reliable solution adds three new cable connections in the first infeasible year t_X to create three new feeders and to reduce the number of customers per feeder. A balanced solution adds only one new cable connections in the year t_X.

Table 6.4: Solutions of DNEP considering Total Cost (CAPEX + DSM) vs. CML per year

Extreme Solution	Total Cost (EUR) €	CML per year (minutes)
NETWORK 1		
Most economical	29670	5634
Most reliable	157581	4005
NETWORK 2		
Most economical	12451	22178
Most reliable	95198	14221
NETWORK 3		
Most economical	25410	22212
Most reliable	411415	12032

Figure 6.7 shows the non-dominated fronts of solving DNEP regarding the total cost (CAPEX+DSM) versus the averaged CML. For Networks 1 and 2, both MO-GOMEA variants DQ and BX-M obtain similar results. For Network 3, MO-GOMEA BX-M obtains a better non-dominated front than MO-GOMEA DQ. However, the performance of the (out-of-the-box) MO-GOMEA DQ is still reasonably good. This confirms the reliability of the basic variant of MO-GOMEA. The effects of DQ and BX-M here conform quite well with the results in Chapters 4 and 5 in the case of solving the single-objective (SO) DNEP with SO GOMEA. Evolutionary optimization algorithms that make use of linkage learning (LLEAs) are usually robust solvers, which can be used out-of-the-box or with minimum modifications to solve complicated problems like MO-DNEP or SO-DNEP. Customizing LLEAs with expert knowledge to make them problem-dedicated solvers can help to further improve the obtained results.

6.6. Conclusions

In this chapter, we considered a multi-objective dynamic DNEP problem formulation that includes both asset investments and DSM policies as expansion options. To handle dynamic planning in an efficient manner, we employed a decomposition heuristic that can convert static (single-stage) network configurations into feasible

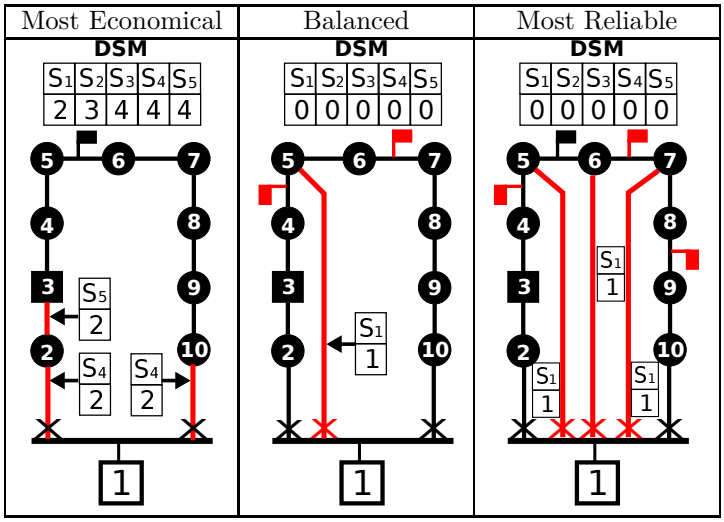

Figure 6.6: Network 1: Total cost (CAPEX+DSM) vs. CML. Red-colored cables indicate new asset installations. Red flags are new positions of NOPs.

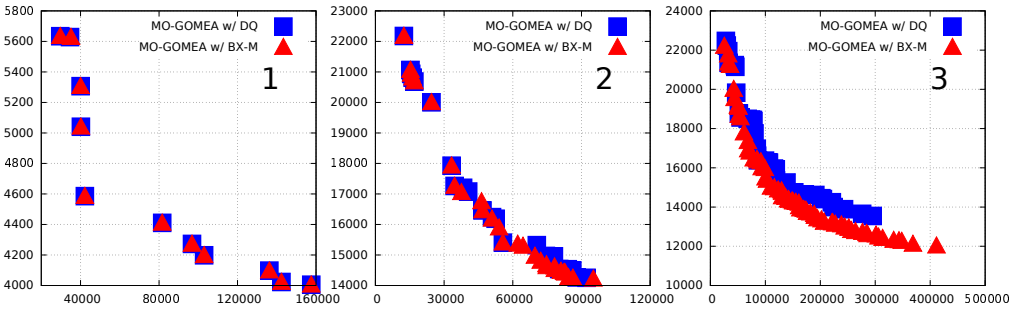

Figure 6.7: Non-dominated fronts of MO optimizations for three networks. **Horizontal axis**: Net Present Value of CAPEX+DSM (EUR). **Vertical axis**: CML per year (minutes). MO-GOMEA DQ is shown as blue squares. MO-GOMEA BX-M is shown as red triangles.

dynamic (stage-by-stage) installation schedules while regarding practical problem constraints. We argued that DNEP is a true MO optimization problem because trade-offs between many conflicting criteria must be taken into account before deciding to carry out a specific expansion plan. We considered the MO-DNEP problem with various combinations of four different objectives: minimizing the physical asset investment cost CAPEX, minimizing the cost of using DSM options, minimizing the total energy loss, and minimizing the customer minutes lost per year. Note that more than two objectives can be used in our framework, but for the purpose of demonstration in this study, considering two objectives at a time is sufficient. We employed the parameter-less MO-GOMEA together with some problem-dedicated adaptations to solve the complicated MO-DNEP for several benchmark networks.

Based on the experimental results, we conclude that solving DNEP in an MO optimization manner returns a far richer set of valuable results and alternatives for DNOs to consider than when solving the SO DNEP. Being exposed to different possible (optimal) trade-offs, DNOs can make well-informed decisions for each specific situation. The non-dominated fronts found by MO optimization can be used as a visualization tool to effectively exhibit how much energy efficiency and network reliability are compromised as DNOs reduce investment costs and also by how much investments must be increased to improve these two objectives. We also showed that by using DSM options to deal with the peak load growth, DNOs can indeed postpone costly asset installations. This can be an incentive for DNOs to actively participate in DSM research, development, and deployment. Finally, MO-GOMEA is shown to be a robust MO solver that can tackle a complicated problem like MO-DNEP well, making it a promising framework to be considered for solving other power and energy optimization problems in the future.

References

[1] N. H. Luong, M. O. W. Grond, H. La Poutré, and P. A. N. Bosman, *Scalable and practical multi-objective distribution network expansion planning,* in *2015 IEEE Power & Energy Society General Meeting* (IEEE, 2015) pp. 1–5.

[2] N. H. Luong, P. A. N. Bosman, M. O. W. Grond, and H. La Poutré, *Evolutionary multi-objective dynamic distribution network expansion planning with demand side management,* in *Application of Modern Heuristic Optimization Methods in Power and Energy Systems*, edited by K. Y. Lee and Z. Vale (IEEE/Wiley (accepted), 2018).

[3] E. Carrano, L. Soares, R. Takahashi, R. Saldanha, and O. Neto, *Electric distribution network multiobjective design using a problem-specific genetic algorithm,* IEEE Transactions on Power Delivery **21**, 995 (2006).

[4] M. Lavorato, M. J. Rider, A. V. Garcia, and R. Romero, *A constructive heuristic algorithm for distribution system planning,* IEEE Transactions on Power Systems **25**, 1734 (2010).

[5] S. Ganguly, N. Sahoo, and D. Das, *Multi-objective planning of electrical distribution systems using dynamic programming,* International Journal of Electrical Power & Energy Systems **46**, 65 (2013).

[6] M. O. W. Grond, N. H. Luong, J. Morren, P. A. N. Bosman, J. G. Slootweg, and H. La Poutré, *Practice-oriented optimization of distribution network planning using metaheuristic algorithms,* in *Proceedings of the Power Systems Computation Conference (PSCC 2014)* (IEEE, 2014) pp. 1–8.

[7] P. Finn, M. O'Connell, and C. Fitzpatrick, *Demand side management of a domestic dishwasher: Wind energy gains, financial savings and peak-time load reduction,* Applied Energy **101**, 678 (2013).

[8] L. Gelazanskas and K. A. Gamage, *Demand side management in smart grid: A review and proposals for future direction,* Sustainable Cities and Society **11**, 22 (2014).

[9] N. Zhang, L. F. Ochoa, and D. S. Kirschen, *Investigating the impact of demand side management on residential customers,* in *2011 2nd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies* (IEEE, 2011) pp. 1–6.

[10] J. Momoh, *Smart Grid: Fundamentals of Design and Analysis* (Wiley-IEEE Press, 2012).

[11] J. G. Slootweg and P. Van Oirsouw, *Incorporating reliability calculations in routine network planning: Theory and practice,* in *18th International Conference and Exhibition on Electricity Distribution, CIRED 2005.* (IET, 2005) pp. 1–5.

[12] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms* (John Wiley & Sons, Inc., 2001).

[13] A. Alarcon-Rodriguez, G. Ault, and S. Galloway, *Multi-objective planning of distributed energy resources: A review of the state-of-the-art,* Renewable and Sustainable Energy Reviews **14**, 1353 (2010).

[14] I. Das and J. E. Dennis, *A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems,* Structural Optimization **14**, 63 (1997).

[15] G. Celli, E. Ghiani, S. Mocci, and F. Pilo, *A multiobjective evolutionary algorithm for the sizing and siting of distributed generation,* IEEE Transactions on Power Systems **20**, 750 (2005).

[16] L. F. Ochoa, A. Padilha-Feltrin, and G. P. Harrison, *Evaluating distributed generation impacts with a multiobjective index,* IEEE Transactions on Power Delivery **21**, 1452 (2006).

[17] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II,* IEEE Transactions on Evolutionary Computation **6**, 182 (2002).

[18] E. Zitzler, M. Laumanns, and L. Thiele, *SPEA2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimization,* in *Evolutionary Methods for Design, Optimisation, and Control* (CIMNE, Barcelona, Spain, 2002) pp. 95–100.

[19] M. H. Ahmadi, H. Hosseinzade, H. Sayyaadi, A. H. Mohammadi, and F. Kimiaghalam, *Application of the multi-objective optimization method for designing a powered Stirling heat engine: Design with maximized power, thermal efficiency and minimized pressure loss,* Renewable Energy **60**, 313 (2013).

[20] A. S. Sousa and E. N. Asada, *Long-term transmission system expansion planning with multi-objective evolutionary algorithm,* Electric Power Systems Research **119**, 149 (2015).

[21] S. Ramesh, S. Kannan, and S. Baskar, *Application of modified NSGA-II algorithm to multi-objective reactive power planning,* Applied Soft Computing **12**, 741 (2012).

[22] P. Murugan, S. Kannan, and S. Baskar, *NSGA-II algorithm for multi-objective generation expansion planning problem,* Electric Power Systems Research **79**, 622 (2009).

[23] M. Pelikan, K. Sastry, and D. E. Goldberg, *Multiobjective hBOA, clustering, and scalability,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2015* (ACM Press, New York, New York, USA, 2005) pp. 663–670.

[24] P. A. N. Bosman and D. Thierens, *The balance between proximity and diversity in multiobjective evolutionary algorithms,* IEEE Transactions on Evolutionary Computation **7**, 174 (2003).

[25] P. A. N. Bosman, *The anticipated mean shift and cluster registration in mixture-based EDAs for multi-objective optimization,* in *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2010* (ACM Press, New York, New York, USA, 2010) pp. 351–358.

[26] J. Knowles and D. Corne, *Properties of an adaptive archiving algorithm for storing nondominated vectors,* IEEE Transactions on Evolutionary Computation **7**, 100 (2003).

[27] S. J. Darby and E. McKenna, *Social implications of residential demand response in cool temperate climates,* Energy Policy **49**, 759 (2012).

[28] J. J. Grainger and W. D. Stevenson, *Power system analysis* (McGraw-Hill, 1994).

[29] M. O. W. Grond, J. I. Pouw, J. Morren, and J. G. Slootweg, *Applicability of line outage distribution factors to evaluate distribution network expansion options,* in *2014 49th International Universities Power Engineering Conference (UPEC)* (IEEE, 2014) pp. 1–6.

[30] K. Deb, *An efficient constraint handling method for genetic algorithms,* Computer Methods in Applied Mechanics and Engineering **186**, 311 (2000).

[31] R. A. Verzijlbergh, M. O. W. Grond, Z. Lukszo, J. G. Slootweg, and M. D. Ilic, *Network impacts and cost savings of controlled EV charging,* IEEE Transactions on Smart Grid **3**, 1203 (2012).

[32] M. O. W. Grond, *Impact of Future Residential Loads on Medium Voltage Networks*, Master thesis, Delft University of Technology (2011).

[33] R. de Groot, J. Morren, and J. G. Slootweg, *Smart integration of distribution automation applications,* in *2012 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)* (IEEE, 2012) pp. 1–7.

7

Conclusion

Summa summarum.

Distribution network expansion planning (DNEP) has always been an important task that distribution network operators (DNOs) conduct in order to draw up strategic network reinforcement plans to handle the growth in power demands. DNEP can be formulated in various ways as optimization problems, for which the desired solutions satisfy a set of problem constraints (e.g., capacity constraints, reliability constraints) and also optimize some objective function (e.g., minimizing investment cost and/or energy loss, maximizing network reliability) regarding the forecast load growth during the planning period. Solving DNEP is non-trivial and involves considerable challenges in terms of problem modeling and computation. In this thesis, we pinpointed these challenges and how evolutionary algorithms (EAs) could be employed to tackle our DNEP problems. Rather than using EAs in an ad hoc manner, we proposed a systematic approach for designing robust EAs to apply in real-world optimization tasks like DNEP problems.

7.1. Distribution network expansion planning

In Chapter 1, we gave an overview of DNEP and described the related problem modeling and computational challenges. First, DNEP involves multiple complicated operational and engineering constraints as well as time-related decision variables, which are difficult to be modeled and to be efficiently handled. Second, transitions toward future sustainable energy systems raise the questions about how smart grid technologies (e.g., energy storage system, demand side management) can also be considered as network reinforcement options together with traditional assets (e.g., electric cables, transformers). Third, DNEP often involves multiple planning objectives which conflicts with each other such that a *utopian* solution optimizing all the objectives at the same time does not exist. In Chapters 4, 5, and 6, we presented how these issues could be properly addressed.

7.1.1. DNEP with AC power flow model and dynamic planning

The so-called *static* DNEP addresses the questions *where* on the network under concern reinforcement activities should be performed and *what* kinds of assets should be installed there. Solving this basic form of DNEP is not trivial because the globally optimal expansion plan has to be looked for in a vast combinatorial search space with many local optima due to complicated, non-linear, constraints. First, in Chapter 4, we showed how domain/expert knowledge can be used to narrow the search space by disregarding impractical expansion options, such as the downgrading of cable capacities or the connecting of distant substations. The DNEP problem has often been simplified by linearizing the non-linear AC power flow model or by omitting difficult constraints. We then argued that DNEP should be more accurately modeled, employing the true non-linear AC power flow model and retaining realistic constraints (e.g., radiality and reconfigurability). We described different evolutionary optimization algorithms and presented different ways of customizing them to be dedicated DNEP solvers, offering high-quality solutions within reasonable computing times. Our methodology offers the flexibility to be adopted by practitioners to solve their DNEP problem instances and other power system expansion planning problems as well.

The difficulty of DNEP increases if it involves the question *when* each reinforcement activity should be carried out during the planning period. Such so-called *dynamic* DNEP is often the case in practice since DNOs would normally like to have a detailed, for example, year-by-year asset installation schedule. The network configuration in each year must then be modeled and evaluated against the load profile in that year, substantially increasing the problem complexity and computing time. If a planning period covers 30 years, then the number of decision variables would be multiplied by 30. Even for medium-sized networks, such curse of dimensionality can make dynamic planning intractable. Therefore, it would be more practical and efficient if the problem models and the optimization algorithms that we have developed in Chapter 4 for static DNEP can also be employed to solve dynamic DNEP. To this end, in Chapter 5, we proposed a decomposition heuristic that can derive an acceptable cable installation schedule for each candidate network configuration, effectively transforming a static expansion plan into a dynamic expansion plan. With the obtained schedule, the network configuration in each year of an expansion plan can be easily determined and assessed against the peak load profile of that year. This decomposition heuristic can be used by any static DNEP solvers (e.g., the algorithms presented in Chapter 4) during the optimization process to obtain an installation schedule for each candidate solution and evaluate its objective function value and constraint violation accordingly, thereby solving dynamic planning in an efficient manner. Furthermore, in Chapters 5 and 6, we also showed that the decomposition heuristic can be customized to consider smart grid technologies (e.g., energy storage system, demand side management) in dynamic DNEP. The inclusion of these emerging technologies in DNEP offers DNOs a wider set of alternatives for network reinforcement but also poses additional challenges that need to be properly addressed.

7.1.2. DNEP with smart grid technologies

Apart from the annual growth in residential and industrial power demands, DNOs are challenged by the recent and increasing presence of distributed generation (DG) and electric vehicles (EVs). Without proper planning and management, unbalanced power injection/demand and uncontrolled EV charging can drastically raise the peak load on the network. We have assumed that it is possible for DNOs to achieve peak shaving by making investments in smart grid technologies, such as demand side management (DSM) and battery energy storage system (BESS). In Chapters 5 and 6, we modeled BESS and DSM as decision variables in DNEP problems, along with the traditional electric cable reinforcement. To handle increasing peak power demands, DNOs then have two strategies: 1) they can install new cables and transformers and thereby enhance the network capacity; or 2) they can invest in DSM and/or BESS to reduce the peak load on the network and thereby improve the efficiency of the current network infrastructure usage. The desirable expansion plans can be mixtures of both traditional asset installations and smart grid investments.

Because smart grid technologies is an on-going research domain, it is difficult to properly estimate their future construction and operation costs, which are important input data for solving DNEP as an optimization problem where the most economical expansion plan needs to be found. To deal with such uncertainty, in Chapter 5, we presented that the DNEP problem with BESS can be solved with the scenario-based approach. We proposed different scenarios of BESS implementation costs, and for each scenario, the DNEP problem is solved to obtain a high-quality expansion plan. This methodology can be straightforwardly applied for solving the DNEP problem with DSM as well as other smart grid technologies. DNOs can thereby investigate possible solutions if they would like to consider smart grid alternatives for DNEP in the presence of price uncertainty of these new technologies.

DNEP in general and the cases with smart grid technologies in particular involve many aspects and decision factors (e.g., investment cost versus network reliability, physical assets versus DSM) that exhibit some trade-off relationship which DNOs would normally like to investigate in their decision making process. It is therefore beneficial to be able to efficiently obtain such trade-off information in order to provide DNOs with more insights into the DNEP problem instance at hand.

7.1.3. DNEP with multi-objective optimization

DNEP can involve multiple conflicting criteria, such as investment costs, energy losses, or network reliability, and a *utopian* expansion plan optimizing all these terms at the same time hardly ever exists. Traditionally, all non-monetary factors of interest are monetized and then combined with the investment cost into a single objective function, for which available (single-objective) optimization algorithms can be used to find the solution with the least total cost. In Chapter 5, we demonstrated the limitations of such an aggregation approach in a case study, where energy losses were monetized as the operational cost. Experimental results showed that, within a certain planning period, the energy losses might dominate the cost function, pushing the optimization algorithm to opt for early installations of new cable connections using the thickest available cable type. In current DNEP practice, however, too early

capacity expansion is not a desirable solution because while energy loss is important, it might not always be the deciding factor for DNOs to choose *when* and *how* network enhancements should be carried out. The problem can be circumvented by weighting down the cost of energy losses, but finding proper scaling coefficients to express the relative importance between investment costs and energy losses is definitely not a trivial task. Besides, in the transition toward sustainable energy systems, the options of reducing energy losses should not be overlooked; that is, they should be considered as alternatives alongside the expansion options that minimize the investment cost. Therefore, instead of telling DNOs what is the optimal expansion plan for the network under concern, we could present to them what are the best available alternatives, reflecting the consequences for all objectives if one alternative is chosen over the others. The DNOs will then choose which alternative to carry out, regarding their own development/business strategies and specific situations.

In Chapter 6, we formulated the DNEP with DSM as a multi-objective optimization problem, where the involved objectives were kept separate. Instead of a single solution as in the case of single-objective optimization, the result of a multi-objective DNEP problem is a set of trade-offs that inform DNOs about the compromises on energy efficiency and network reliability if the investment cost is decreased, and also, the cost-effective solutions to reduce energy losses and customer minutes lost if DNOs would like to do so. This thesis therefore advocates the modeling of DNEP as a *true* multi-objective optimization problem. The advantages of such approach are manifold: 1) Non-financial criteria are not required to be capitalized; 2) DNOs and DNEP practitioners do not need to specify *a priori* the weights (i.e., the relative importance) of the involved criteria; 3) the optimization result is a rich set of trade-off solutions, offering valuable insights into different possibilities of network expansion strategy.

7.2. Robust evolutionary algorithm design and application

While solving DNEP with real-world constraints, especially its multi-objective formulations, provides useful information for DNOs to consider during their network expansion planning processes, the aforementioned computational challenges associated with DNEP must be properly addressed. In Chapter 1, we described the main reasons that evolutionary algorithms (EAs) are the appropriate solvers to tackle our DNEP problems. First, EAs evolve a population of candidate solutions, which are naturally well-suited for multi-objective optimization in the sense that a set of multiple solutions that approximate the trade-off relationship between the involved objectives can be obtained in one optimization run. Second, EAs offer a great flexibility in handling a wide range of optimization models, especially the ones with problem constraints and objective functions that are too complicated to be directly exploited in a white-box context like the AC power flow model-based operational constraints in our DNEP problems. Furthermore, properly-designed EAs allows the optimization model to be enlarged and extended (e.g., handling large distribution networks, tackling dynamic planning, or including smart grid technologies) without

the need of major modifications in the optimization algorithms. In this thesis, we proposed how EAs could be designed for successful applications in DNEP.

The performance results of ad hoc applications of EAs in DNEP are limited to the problem instances at hand and are difficult to generalize to other cases, especially when the problem sizes are enlarged or the problem models are extended. Therefore, rather than employing the ad hoc approach, in this thesis, we have worked on the fundamentals of designing robust EAs, in particular the following topics: eliminating parameter settings of EAs in real-world applications, exploiting domain knowledge and linkage information to enhance the efficiency of EAs, and scalability-oriented design of (multi-objective) EAs.

7.2.1. Eliminating parameter settings of evolutionary algorithms

While evolutionary algorithms (EAs) enjoy their popularity as the solvers of choice in a wide range of industrial optimization problems, they often receive the criticism that their (many) control parameters need to be properly set to obtain good performance. Finding the optimal combination of parameter settings itself is an optimization problem, and parameter tuning is also time-consuming. Indeed, DNEP practitioners, typically non-EA experts, can be discouraged by having to define unfamiliar algorithm parameters, for which the corresponding suitable values depend on each specific problem instance. We argue that parameter handling should be the burden of algorithm designers, not of the users. To this end, in this thesis, we presented EA solvers of different kinds and proposed how their parameter settings can be eliminated. The elimination of EA parameter settings improves the usability of the EA solvers such that DNEP practitioners can focus on problem modeling rather than tuning the parameters of the solvers.

In Chapter 2, we described the three typical population-based EAs, namely Genetic Algorithm (GA), Estimation-of-Distribution Algorithm (EDA), and Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA). We chose to describe specific implementations of these three EAs that only require users to define their population size, which can be considered as one, if not the most, crucial parameter. To eliminate the population size setting, we then employed the Harik-Lobo population-sizing-free scheme. In essence, the Harik-Lobo scheme operates an EA with an unbounded number of populations of increasing sizes in an interleaved fashion, where the smaller populations are always initialized before larger ones, and the larger populations have slower generational cycles than smaller ones. The Harik-Lobo scheme can be used as a framework for running any population-based EAs without the need of modifications in the working mechanism of the EA solvers.

The straightforward implementation of the Harik-Lobo scheme makes it a candidate platform to compare the performance of different EAs. Performance benchmarking and comparisons between population-based EAs are often conducted with optimization problems whose optimal solutions are known so that the bisection method can be used to find the minimally required population size and the corresponding number of evaluations. However, there are many problems, especially in real-world applications, where optimal solutions are unknown. The population size of each competing EA is then set based on some general guidelines, which can be

far from the optimal value depending on the particular problem instance at hand. Consequently, it is difficult to assess the true performance of EA solvers with such settings. Regarding the issue, we therefore emphasize two points in this thesis: 1) future EA designs should free users from the burden of control parameter settings, and 2) when benchmarking the performance of different EA solvers, all competing EAs must be put on an equal footing to ensure a fair comparison.

We note that the use of the Harik-Lobo scheme might not offer the best performance of EA solvers (i.e., when the optimal parameter settings for the problem instance at hand are known). Nevertheless, the elimination of parameter settings considerably improves the usability of EAs, and in practice, the goal is to solve the problem instance at hand rather than determining the optimal parameter settings, which is impossible without running the optimization with many different parameter settings. Furthermore, the performance of EAs, in the context of DNEP, typically depends on their capability in exploiting the DNEP problem-specific knowledge (e.g., the connectivity and radiality of the network topology) as well as the interactions between problem variables during the optimization process.

7.2.2. Exploiting linkage information and problem-specific knowledge

Because most EAs are general-purpose solvers, they contain little or no problem-specific information when just taken out-of-the-box. Different EAs have different capabilities to adapt themselves to the particular problem instance under concern. It is important that building blocks (i.e., good partial solutions) in the population are not frequently disrupted by EAs' stochastic operators and that they can be efficiently mixed to create better solutions. The classic variation operators of the simple GA, i.e., uniform crossover and mutation, are often very disruptive since all problem variables are (implicitly) assumed to be independent from each other. In the context of DNEP, for networks of small sizes, the simple GA can still function fairly well. However, for networks of large sizes, the probability that building blocks are disrupted considerably increases, leading to unacceptably poor performance.

Problem-specific knowledge, if available, can be employed to inform the stochastic operators of EA solvers to respect building blocks during the optimization process. If such domain knowledge is not available, as in black-box optimization, or cannot be straightforwardly exploited, building block information can be (partly) deduced by detecting dependency structures among problem variables. Dependency structures are often captured by learning linkage models from the working population of EAs. The learned models are then used to inform the variation operators which variables should be treated together during recombination. The complexity of the linkage model defines what kind of dependency the model can capture. For example, the marginal product model can match non-overlapping linkage-set structures and the linkage tree model can match hierarchical dependencies. Employing the linkage tree model, in this thesis, GOMEA exhibited superior performance on a wide ranges of problems. On small problem instances, GOMEA performs comparably with, or better than, other EAs. As the problem sizes increase, the performance gaps enlarge considerably: GOMEA obtains high-quality (or optimal) solu-

tions within available (and reasonable) computing budgets while other EAs would require much more computing time and resource to obtain solutions of the similar quality. The performance of GOMEA can be explained by: 1) the capability of the linkage tree model to match multi-level dependency structures, which is especially suitable for the DNEP problem; and 2) the efficiency of the Gene-pool Optimal Mixing (GOM) operator in exploiting the learned linkage models to construct new candidate solutions. Linkage learning requires additional computations, but the much better results obtained by GOMEA can easily offset its overhead, especially in cases where the solution evaluation functions dominate the total computing time, which are very typical in industrial optimization.

The dependency structure might however not be the only, or the most essential, type of structure in a problem. Successful applications of EAs in industrial optimization often require some adaptations of the solver's operators to the specific properties of the problem under concern. In Chapter 3, we pointed out that the connectivity structure is the most important property of DNEP solutions that should be exploited by EA solvers. We then studied the network connectivity and radiality knowledge to customize the operation of different optimization algorithms, creating specialized solvers for DNEP. However, for the purpose of performance analysis, it is useful to separate the contribution of domain knowledge from the uninformed optimization results so that each solver can be more precisely assessed. Therefore, in this thesis, we performed experiments for every EA in pairs of variants: the original implementation versus a problem-specific adapted version. The experimental results suggested that problem-specific knowledge is essential to the success of GA in real-world applications. The out-of-the-box simple GA is very inefficient in recombining and preserving feasible solutions, let alone finding the optimum. On the other hand, GOMEA is a more robust EA in the sense that even without specialization, GOMEA can perform linkage learning to capture the dependency structure and exploit the learned linkages to more efficiently create offspring solutions of better quality. Customization with domain knowledge (in this case, the network connectivity structure) does improve the performance of GOMEA, but the improvement gap is not as substantial as that of simple GA. The best EA configurations, therefore, are the ones that exploit both linkage information and problem-specific knowledge.

The capability to exploit linkage information and problem-specific knowledge is a crucial factor that contributes to the scalability of EAs solving DNEP problems. In this thesis, we pinpointed the key components for designing scalable EA solvers, especially in the multi-objective DNEP case.

7.2.3. Scalability-oriented design of evolutionary algorithms

While in many real-world applications, EAs are designed and adapted to be effective in solving the problem instances under concern, the scalability issue is often overlooked. Scalability is important because scalable solvers are *reusable* software, which can be straightforwardly employed to tackle different problem instances of bigger sizes without the need of re-implementing to fit new problem instances. Therefore, we recommend that the scalability-oriented perspective should be adopted in future EA designs and applications.

There are many factors that affect the scalability of an EA, including the adaptations of control parameters, or the exploitation of linkage information and domain knowledge, which were discussed in the previous sections. In Chapter 3, we focused on how to design scalable multi-objective evolutionary algorithms (MOEAs), but some general ideas can be transferred to the single-objective case as well. Given a computing budget, the goal of multi-objective optimization is to find a set of non-dominated solutions that well approximates the true Pareto-optimal front. We presented a diversity-based implementation of the elitist archive to keep track of the non-dominated solutions obtained during the optimization process. Elitist archiving in multi-objective optimization is necessary to prevent non-dominated solutions from being accidentally discarded as the size of the working population is normally limited compared to the numerous (or even infinite) number of non-dominated solutions. When the capacity of the elitist archive is exceeded, the archive will be trimmed such that the remaining elitist solutions are as diverse as possible. It is possible that solutions on different regions of the Pareto-optimal front have different characteristics, and it would be difficult for a single population to maintain equal search biases toward all those parts at the same time. Therefore, in each generation, the working population is partitioned into a number of equal-sized clusters, ensuring that all regions of the Pareto-optimal front are evenly approached. For every cluster, a separate linkage tree model is learned to capture the dependency structures among problem variables specific to candidate solutions in that cluster. Each existing (parent) solution is transformed into a new offspring by employing the Gene-pool Optimal Mixing operator to iteratively recombine the parent solution with other donor solutions in the same cluster that it belongs to, regarding the learned linkage tree of that cluster. It can be seen that while the elitist archive functions at the global level, keeping track of the overall status of the optimization process, the linkage learning and solution recombination operates at local levels, ensuring dedicated search bias is well-tuned toward each region. The resulting algorithm, named MO-GOMEA, has shown superior scalability over other typical MOEAs in a wide range of benchmarks (in Chapter 3) and also proved its applicability to the industrial multi-objective DNEP problem (in Chapter 6). The methodology that we developed here can be employed as a guideline to design scalable MOEAs for tackling other real-world multi-objective optimization problems as well.

7.3. Future research

This thesis has described the modeling and computational challenges in solving DNEP problems and presented how they can be systematically addressed by properly-designed (multi-objective) EAs with the scalability mindset. Our proposed EA methodology can serve as a DNEP framework and can be extended for future works in several ways as follows.

First, this thesis considered a "conservative" approach, in which candidate network configurations are evaluated against the peak power demand all over the network, which can be considered as the worst-case scenario. However, regarding the mobility of EVs and the volatility of DG, there might be multiple other worst-case scenarios with different occurrence times and locations of peak loads. If the peak

power demand and the power injection of EVs and DG can be properly modeled by some probability distribution, then Monte Carlo methods can be used to evaluate candidate expansion plans against the probable peak load profiles that can be sampled from the distribution. Efficient computing methods must be developed to perform such many solution evaluations within a reasonable amount of time.

Second, there are many perturbations in carrying out a multi-year distribution network expansion plan in practice. For example, some planned equipment might not be available in the future and would need to be changed to different devices, the installation time might be delayed, or the construction time might need to be extended. All these changes affect the actual objective values of the expansion plan and even its feasibility in practice. For future works, our methodology here can be enhanced by introducing the notion of *robustness* of candidate expansion plans in the presence of variable perturbations during the DNEP process.

Last but not least, many-objective optimization problems, i.e., problems having four or more objectives, are beyond the scope of this thesis. Multi-objective optimization tasks in current DNEP practice mostly involve two or three objectives, and higher-dimensional objectives can be transformed and collapsed into lower-dimensional representations. This approach is acceptable partly because DNOs are the primary, if not sole, stakeholder in maintaining and operating distribution networks. However, it can be foreseen that future DNEP and power system expansion planning in general would directly involve multiple stakeholders, e.g., network operators, energy suppliers, regulators, industrial customers, residential consumers, and electric vehicle owners, each with their own interest and different (conflicting) objectives to optimize. Therefore, it can be argued that solving a many-objective optimization problem in its true (non-collapsed) form would better provide all stakeholders a general and unbiased picture of the problem at hand. Analyzing the optimization results of such problems poses considerable challenges, e.g., how to visualize and efficiently traverse high-dimensional Pareto fronts, or how to perform decision makings when too many conflicting objectives must be taken into account. Furthermore, in higher-dimensional objective spaces, almost all solutions in the population can become non-dominated. A Pareto domination-based MOEA will then suffer from the loss of selection pressure, which deteriorates its convergence performance toward the Pareto-optimal front. Many-objective optimization requires novel techniques different from the multi-objective case to be developed. Building a scalable EA for many-objective optimization, therefore, invites further research.

Appendix

MV cable types. Type 12 is sub-transmission cable and is not considered in MV DNEP. Types 4-11 are currently existing cable types but are not used for new cable installations. (N/A: Not Available)

ID	Type	I_{nom}	R (Ω/km)	X (Ω/km)	C ($\mu F/km$)	Cost (EUR/km)
1	120 mm^2	215 A	0.257	0.085	0.38	50 000
2	150 mm^2	295 A	0.20858	0.09592	0.3833	59 000
3	240 mm^2	370 A	0.13517	0.10823	0.43553	62 000
4	400 mm^2	475 A	0.08077	0.09972	0.5344	N/A
5	630 mm^2	605 A	0.0511	0.09272	0.64103	N/A
6	N/A	135 A	0.53253	0.09777	0.27072	N/A
7	N/A	160 A	0.3737	0.09367	0.30671	N/A
8	N/A	195 A	0.26756	0.08995	0.34505	N/A
9	N/A	225 A	0.32829	0.10134	0.32667	N/A
10	N/A	320 A	0.13079	0.07757	0.53109	N/A
11	N/A	350 A	0.10193	0.08004	0.48485	N/A
12	N/A	N/A	N/A	N/A	N/A	N/A

Network 1 Data

Node Information				Cable Information				
ID	Load		Customers	Existing			Potential	
	P [kW]	Q [$kVAR$]	[#]	Branch	Length [m]	Type	Branch	Length [m]
1	0	0	0	1 - 2	654	1	1 - 3	1235
2	271	168	131	1 - 10	710	1	1 - 4	1259
3	924	573	1	2 - 3	610	1	1 - 5	1323
4	394	244	190	3 - 4	163	1	1 - 6	1711
5	409	253	197	4 - 5	511	1	1 - 7	1904
6	394	244	190	5 - 6	496	1	1 - 8	1976
7	370	229	179	6 - 7	420	1	1 - 9	1781
8	117	72	57	7 - 8	297	1		
9	259	160	125	8 - 9	336	1		
10	431	267	208	9 - 10	690	1		

Network 2 Data

Node Information				Cable Information				
ID	Load		Customers	Existing			Potential	
	P $[kW]$	Q $[kVAR]$	$[\#]$	Branch	Length $[m]$	Type	Branch	Length $[m]$
1	0	0	0	1 - 2	481	3	1 - 3	526
2	35	17	25	1 - 16	246	3	1 - 4	469
3	1113	539	1	1 - 31	761	3	1 - 5	989
4	348	216	1	2 - 3	96	2	1 - 6	2062
5	871	286	1	3 - 4	48	2	1 - 7	738
6	332	109	232	4 - 5	498	2	1 - 8	2227
7	132	82	92	5 - 6	86	2	1 - 9	2307
8	170	82	119	6 - 7	288	2	1 - 10	2633
9	22	14	15	7 - 8	935	2	1 - 11	3041
10	202	98	141	8 - 9	200	2	1 - 12	3041
11	120	0	84	9 - 10	470	2	1 - 13	1395
12	88	55	62	10 - 11	851	3	1 - 14	1194
13	284	137	199	10 - 17	736	2	1 - 15	923
14	219	136	153	11 - 12	220	3	1 - 17	2808
15	314	152	220	12 - 13	300	3	1 - 18	2760
16	185	90	130	13 - 14	284	3	1 - 19	2653
17	127	79	1	14 - 15	479	3	1 - 20	1275
18	17	8	12	15 - 16	846	3	1 - 21	1205
19	896	434	1	17 - 18	101	2	1 - 22	1136
20	314	152	220	18 - 19	154	2	1 - 23	1131
21	125	77	88	19 - 20	283	2	1 - 24	1041
22	248	120	174	20 - 21	308	2	1 - 25	950
23	85	41	60	21 - 22	133	2	1 - 26	966
24	123	76	86	22 - 23	132	2	1 - 27	900
25	209	130	146	23 - 24	138	2	1 - 28	804
26	566	274	1	24 - 25	140	2	1 - 29	677
27	266	129	186	25 - 26	103	2	1 - 30	801
28	126	61	88	26 - 27	215	2		
29	360	174	1	27 - 28	139	2		
30	273	169	191	28 - 29	218	2		
31	263	163	1	29 - 30	136	2		
				30 - 31	160	3		

Network 3 Data

Node Information				Cable Information						
ID	Load		Customers	Existing			Potential			
	P $[kW]$	Q $[kVAR]$	$[\#]$	Branch	Length $[m]$	Type	Branch	Length $[m]$	Branch	Length $[m]$
1	0	0	0	1 - 2	1	12	2 - 5	1335	8 - 38	1900
2	67	32	22	1 - 36	1	12	2 - 6	1357	8 - 40	2754
3	185	90	108	2 - 3	670	6	2 - 9	1256	8 - 41	3542
4	112	54	1	2 - 4	280	7	2 - 10	1040	8 - 42	3252
5	194	94	92	2 - 8	1	12	2 - 13	1840	8 - 43	2847
6	61	30	14	2 - 12	1820	7	2 - 14	1987	8 - 44	2696
7	152	73	216	3 - 5	570	6	2 - 15	2123	8 - 45	2510
8	158	76	143	4 - 35	380	9	2 - 16	1948	8 - 46	2148
9	282	136	317	5 - 17	570	6	2 - 17	1541	8 - 47	1900
10	193	94	153	6 - 7	1300	6	2 - 18	2507	8 - 48	1781
11	165	54	39	6 - 38	759	9	2 - 19	2437	8 - 49	1725
12	148	72	1	7 - 8	1421	6	2 - 20	3102	8 - 51	3405
13	91	44	47	7 - 9	610	7	2 - 21	2885	9 - 23	1840
14	119	57	1	8 - 11	360	7	2 - 28	3605	9 - 36	1693
15	311	150	186	8 - 24	570	8	2 - 29	3160	10 - 23	1981
16	314	152	295	8 - 27	570	8	2 - 30	3950	10 - 36	1890
17	333	161	245	9 - 10	250	7	2 - 31	3560	13 - 23	1496
18	351	170	397	10 - 11	340	7	2 - 32	3855	13 - 36	1189
19	236	114	167	12 - 13	320	7	2 - 33	3769	14 - 23	1822
20	297	144	351	13 - 14	380	7	2 - 34	2437	14 - 36	1388
21	253	122	264	14 - 15	150	7	2 - 35	372	15 - 23	1978
22	355	172	9	15 - 16	800	7	2 - 38	372	15 - 36	1496
23	492	238	208	15 - 18	510	7	2 - 40	4483	16 - 23	2726
24	152	74	34	16 - 17	570	7	2 - 41	5283	16 - 36	2277
25	156	75	1	18 - 19	280	7	2 - 42	4969	17 - 23	3063
26	186	90	41	19 - 34	510	7	2 - 43	4632	17 - 36	2704
27	310	150	211	20 - 21	300	7	2 - 44	4487	18 - 23	1887
28	292	141	4	20 - 34	510	7	2 - 45	4291	18 - 36	1321
29	11	6	1	21 - 22	530	7	2 - 46	3972	19 - 23	1524
30	230	111	8	22 - 28	955	8	2 - 47	3726	19 - 36	983
31	136	66	1	22 - 36	313	9	2 - 48	3593	20 - 23	1563
32	287	139	232	23 - 25	400	7	2 - 49	3546	20 - 36	920
33	298	144	186	23 - 26	350	7	2 - 51	5159	21 - 23	1219
34	174	84	167	23 - 27	350	7	5 - 8	1919	21 - 36	583
35	180	87	1	23 - 36	1	12	5 - 23	3038	23 - 28	1951
36	0	0	0	23 - 39	590	5	5 - 36	2718	23 - 29	2217
37	806	500	1	24 - 25	365	7	6 - 8	3086	23 - 30	2029
38	156	75	1	26 - 31	785	7	6 - 23	4250	23 - 31	900
39	259	126	134	28 - 29	465	8	6 - 36	4112	23 - 32	1194
40	281	136	76	29 - 30	740	8	8 - 9	671	23 - 33	1515
41	310	150	69	30 - 33	685	8	8 - 10	819	23 - 34	1524
42	217	105	2	31 - 32	272	7	8 - 13	530	23 - 35	3061
43	153	74	34	32 - 33	671	7	8 - 14	974	23 - 38	3061
44	137	66	1	35 - 38	426	9	8 - 15	1190	23 - 40	1593
45	259	126	62	36 - 37	94	9	8 - 16	1772	23 - 41	2374
46	261	126	121	36 - 50	450	7	8 - 17	1977	23 - 42	2092
47	226	110	50	39 - 40	640	7	8 - 18	1342	23 - 43	1678
48	269	130	139	40 - 51	1251	7	8 - 19	1036	23 - 44	1528
49	218	106	1	41 - 42	430	7	8 - 20	1557	23 - 45	1340
50	136	66	5	41 - 51	229	9	8 - 21	1231	23 - 46	1017
51	240	116	53	42 - 43	387	7	8 - 28	2088	23 - 47	778
				43 - 44	130	7	8 - 29	1947	23 - 48	629
				44 - 45	520	6	8 - 30	2354	23 - 49	592
				45 - 46	350	6	8 - 31	1731	23 - 51	2235
				46 - 47	233	6	8 - 32	2033	28 - 36	1328
				47 - 48	180	6	8 - 33	2039	29 - 36	1581
				48 - 49	150	7	8 - 34	1036	30 - 36	1456
				49 - 50	435	7	8 - 35	1900	31 - 36	632
									32 - 36	928
									33 - 36	1009
									34 - 36	983
									35 - 36	2945
									36 - 38	2945
									36 - 40	2006
									36 - 41	2721
									36 - 42	2489
									36 - 43	1973
									36 - 44	1818
									36 - 45	1672
									36 - 46	1204
									36 - 47	971
									36 - 48	936
									36 - 49	850
									36 - 51	2567

Acronyms

AC	Alternating Current
ACO	Ant Colony Optimization
BESS	Battery Energy Storage System
BX	Branch Exchanging
BX-M	Branch Exchanging with Mutation
CAPEX	Capital Expenditure
CCC	Combined Complexity Criterion
CML	Customer Minutes Lost
CPC	Compressed Population Complexity
CRB	Connectivity Repair by the Best solution
CRP	Connectivity Repair by Parent
DC	Direct Current
DE	Differential Evolution
DG	Distributed Generation
DNEP	Distribution Network Expansion Planning
DNO	Distribution Network Operator
DQ	Disconnectivity Quantification
DSM	Demand Side Management
EA	Evolutionary Algorithm
ECGA	Extended Compact Genetic Algorithm
EDA	Estimation-of-Distribution Algorithm
EV	Electric Vehicle
FI	Forced Improvement
FOS	Family Of Subsets
GA	Genetic Algorithm
GOM	Gene-pool Optimal Mixing
GOMEA	Gene-pool Optimal Mixing Evolutionary Algorithm
hBOA	hierarchical Bayesian Optimization Algorithm
HL	Harik-Lobo
HV	High Voltage
LLEA	Linkage Learning Evolutionary Algorithm
LODF	Line Outage Distribution Factor
LOTZ	Leading Ones Trailing Zeros
LT	Linkage Tree
LV	Low Voltage
MAMaLGaM	Multi-objective Adapted Maximum-Likelihood Gaussian Model
MBEA	Model-Based Evolutionary Algorithm
MC	Model Complexity

MI	Mutual Information
MILP	Mixed-Integer Linear Programming
MINLP	Mixed-Integer Non-Linear Programming
MO	Multi-Objective
MO-GOMEA	Multi-Objective Gene-pool Optimal Mixing Evolutionary Algorithm
MOEA	Multi-Objective Evolutionary Algorithm
MOEA/D	Multi-Objective Evolutionary Algorithm based on Decomposition
MOEDA	Multi-Objective Estimation-of-Distribution Algorithm
mohBOA	multi-objective hierarchical Bayesian Optimization Algorithm
MOOP	Multi-Objective Optimization Problem
MP	Marginal Product
MV	Medium Voltage
MV-D	Medium Voltage Distribution
NIS	No Improvement Stretch
NOP	Normally Open Point
NPV	Net Present Value
NSGA	Non-dominated Sorting Genetic Algorithm
OPEX	Operational Expenditure
PFC	Power Flow Calculation
PSK	Problem-Specific Knowledge
PSO	Particle Swarm Optimization
PV	Photovoltaics
SAIDI	System Average Interruption Duration Index
SO	Single-Objective
SPEA	Strength Pareto Evolutionary Algorithm
TNEP	Transmission Network Expansion Planning
UF	Univariate Factorization
UMDA	Univariate Marginal Distribution Algorithm
VO	Variation Operator

Acknowledgements

First, I would like to thank my promotor and advisor Professor Han La Poutré for his supervision and invaluable support during the time I conducted my research in the Intelligent & Autonomous Systems (IAS) group (formerly the IS, and SEN4 group) at Centrum Wiskunde & Informatica (CWI). I always appreciate his efforts in the role of the group leader and a management team member of CWI to establish a supportive environment for all students and scientists to conduct our research. I am grateful for his supports for me to attend many conferences to broaden my professional networks and meaningful summer schools to acquire valuable knowledge and experience. During my research, Han helped me strengthen my research questions and arguments as well as pointed out important issues that I had overlooked. Had it not been for his research guidance and his conscientious proofreading of my writings, I would not have managed to publish at prestigious conferences and journals. The completion of my PhD research and this dissertation owes a considerable part to Han. Last but not least, I thank Han for enlightening me about the conjecture that, for similar prices, the probability of having a good white wine is considerably larger than that of having a good red wine, which is in line with my drinking experience.

Next, I would like to thank my promotor and daily supervisor Peter Bosman for his tremendous contributions to my research during my PhD time in the IAS group and my postdoctoral time in the Life Sciences & Health (LSH) group at CWI. The Earth could not be any rounder since we competed against each other for the Best Paper Award at GECCO 2010 (where Peter won the award with the most outstanding presentation that I have ever witnessed) until we co-authored and won the same prize together at GECCO 2015. Somehow, in my experience with Peter, the borders between advisors and students and the borders between colleagues and friends do not seem to exist. I can walk into his office without appointment for discussing about interesting ideas, reporting new results, or simply showing him a graph that I have just printed. I especially enjoy and appreciate every moment that we brainstorm over algorithm designing (on the whiteboards, papers, tablets, or WhatsApp), that when we code, test, and debug together, and that when Peter unconsciously makes some humming tunes each time he shows his magical skills in awk and gnuplot. Had it not been for this close teamwork with Peter, I would not have been able to shape my research career and to find the answers to my research questions. As we speak the same Evolutionary Algorithms language and share the "singleton" philosophy in writing C source codes, I believe in and look forward to our continuous collaborations in the future. Also, I would like to thank Tanja Alderliesten at the Academic Medical Center (AMC) in the same paragraph with Peter. Tanja and Peter offered me the precious permission to finish this dissertation during my postdoctoral time. Had it not been for their support, I would not have found enough space and time to complete my writings.

Next, I would like to thank our project partners Marinus Grond, Johan Morren, and Han Slootweg at Technische Universiteit Eindhoven and Enexis B.V. for the fruitful collaboration in the COCAPLEN project. I am grateful to Johan Morren and Han Slootweg for sharing their expertise and practical advice in electrical engineering. I especially thank Marinus Grond for his preparing of real distribution network data that I used in this dissertation and other publications. Coming from the computer science background, I owe my understanding about power systems in a large part to many meaningful discussions and meetings with Marinus Grond, Johan Morren, and Han Slootweg. Also, I would like to thank Gabriël Bloemhof at DNV GL, an invaluable partner in our COCAPLEN project. I appreciate his exploratory mindset in asking novel questions and transcending conventional constraints when doing scientific research for real-world applications.

Moreover, I would like to thank my colleagues and friends at CWI and AMC for bringing so much joy and happiness to my life in the Netherlands. I especially thank Jasper Hoogland for helping me struggle with the notoriously difficult Dutch grammar and pronunciation. It has been a great pleasure to share my office with Sara Ramezani and Majid Khoshrou in the IAS group, and later with Kalia Orphanou and Alejandro Rincon in the LSH group. During my time in the Netherlands, I am grateful for having the opportunities to work in the same places with many talented and helpful people: Eric Pauwels, Nicolas Höning, Felix Claessen, Joeri van Eekelen, Bart Liefers, Michael Kaisers, Georgios Methenitis, and Chau Do (IAS group, CWI); Sílvio Rodrigues (Delft University of Technology); Dirk Thierens (Utrecht University); Krzysztof Sadowski, Marco Virgolin, Anton Bouter, Jasmijn Baaijens, Alexander Schönhuth, Marleen Balvert, Gunnar Klau, and Leen Stougie (LSH group, CWI); Stef Maree, Marjolein van der Meer, Kleopatra Pirpinia, and Arjan Bel (AMC). Had it not been for their warm presence, a tropical soul like me would not have survived the Dutch 11-month-long winter weather (a.k.a. *lekker weertje*). I thank Sílvio Rodrigues for helping me with the experiment of the linear population-sizing-free scheme in Section 3.6.2 of Chapter 3 of this dissertation. Also, my special thanks are to Stef Maree and Peter Bosman for translating the summary and the proposition list of this dissertation into Dutch.

Furthermore, I would like the send my thanks to the brothers and sisters (which is the way Vietnamese people calling each other as we consider that all of us belong to the same family) in the Vòng Tay Amsterdam community for being my first friends when I just came to the Netherlands, especially Thuong Bui, So Pham, the families Tâm-Kiêt, Vân-Canh, and Dũng-Tú. My sincere thank is to the family Duc-Vân for their friendship and generous help in my daily life. Especially, I heartily thank Nghĩa for having always been by my side, no matter when you are here or in Vietnam, during the last many years. Thank you from the bottom of my heart for being the endless source of love and support. Lastly, I owe my gratitude to my parents. Thank you for your invariable encouragement and unconditional love in my whole life. Had it not been for your support, I would not have been able to complete this journey.

Curriculum Vitæ

Ngoc Hoang Luong was born in Ho Chi Minh City (Vietnam) on September 24th, 1986. He attended the Pho Thong Nang Khieu high school from 2001–2004. He obtained his bachelor degree in Information Technology from Ho Chi Minh City University of Science (Vietnam) in 2008. From 2009–2011, he conducted his master research project in the Sungkyunkwan Evolutionary Algorithm Laboratory, and received a Master of Science degree in Electrical and Computer Engineering from Sungkyunkwan University (South Korea) in 2011. In December 2011, he started his doctoral research at Centrum Wiskunde & Informatica (CWI, The Netherlands) in the Intelligent & Autonomous Systems (IAS) group. Since 2016, he has worked as a researcher in the Life Sciences & Health (LSH) group, also at CWI.

His research is concerned with designing scalable evolutionary algorithms and their applications in real-world optimization. In the IAS group, he works on solving electricity distribution network expansion planning problems. In the LSH group, his research involves designing efficient multi-objective evolutionary algorithms to assist radiation oncologists in brachytherapy planning for cancer treatment.

Ngoc Hoang Luong won the Best Paper Award in the Real-World Applications track at the Genetic and Evolutionary Computation Conference GECCO 2015 (Madrid, Spain). His papers were also nominated for the Best Paper Award in the Estimation-of-Distribution Algorithms track at GECCO 2010 (Portland, USA), and in the Genetic Algorithms track at GECCO 2016 (Denver, USA).

List of Publications

10. **N.H. Luong**, P.A.N. Bosman, M.O.W. Grond, H. La Poutré (2018). Evolutionary Multi-objective Dynamic Distribution Network Expansion Planning with Demand Side Management. In Book: *Application of Modern Heuristic Optimization Methods in Power and Energy Systems*. IEEE/Wiley (accepted for publication).

9. **N.H. Luong**, H. La Poutré, P.A.N. Bosman (2018). Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm with the Interleaved Multi-start Scheme. *Swarm and Evolutionary Computation*, vol. 40, pp. 238-254. Elsevier.

8. **N.H. Luong**, H. La Poutré, P.A.N. Bosman (2018). Exploiting Linkage Information and Problem-Specific Knowledge in Evolutionary Distribution Network Expansion Planning. *Evolutionary Computation*, vol. 26, no. 3. MIT Press.

7. **N.H. Luong**, H. La Poutré, P.A.N. Bosman (2015). Exploiting Linkage Information and Problem-Specific Knowledge in Evolutionary Distribution Network Expansion Planning. In: *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15)*, pp. 1231-1238. ACM.

6. **N.H. Luong**, H. La Poutré, P.A.N. Bosman (2015). Scalable and Practical Multi-objective Distribution Network Expansion Planning. In: *Proceedings of Power & Energy Society General Meeting (PES-GM '15)*, pp. 1-5. IEEE.

5. M.O.W. Grond, **N.H. Luong**, J. Morren, P.A.N. Bosman, J.G. Slootweg, and H. La Poutré (2014). Practice-oriented Optimization of Distribution Network Planning using Metaheuristic Algorithms. In: *Power Systems Computation Conference (PSCC '14)*, pp. 1-8. IEEE.

4. **N.H. Luong**, H. La Poutré, P.A.N. Bosman (2014). Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithms. In: *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*, pp. 357-364. ACM.

3. **N.H. Luong**, M.O.W. Grond, H. La Poutré, P.A.N. Bosman (2014). Efficiency Enhancements for Evolutionary Capacity Planning in Distribution Grids. In: *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*, pp. 1189-1196. ACM.

2. **N.H. Luong**, M.O.W. Grond, P.A.N. Bosman, and H. La Poutré (2013). Medium-Voltage Distribution Network Expansion Planning with Gene-pool Optimal Mixing Evolutionary Algorithms. In: *Artificial Evolution 2013*, pp. 93-105. Springer.

1. **N.H. Luong**, P.A.N. Bosman (2012). Elitist Archiving for Multi-objective Evolutionary Algorithms: To Adapt or Not to Adapt. In: *Proceedings of the 12th International Conference on Parallel Problem Solving from Nature (PPSN '12)*, pp. 72-81. Springer.