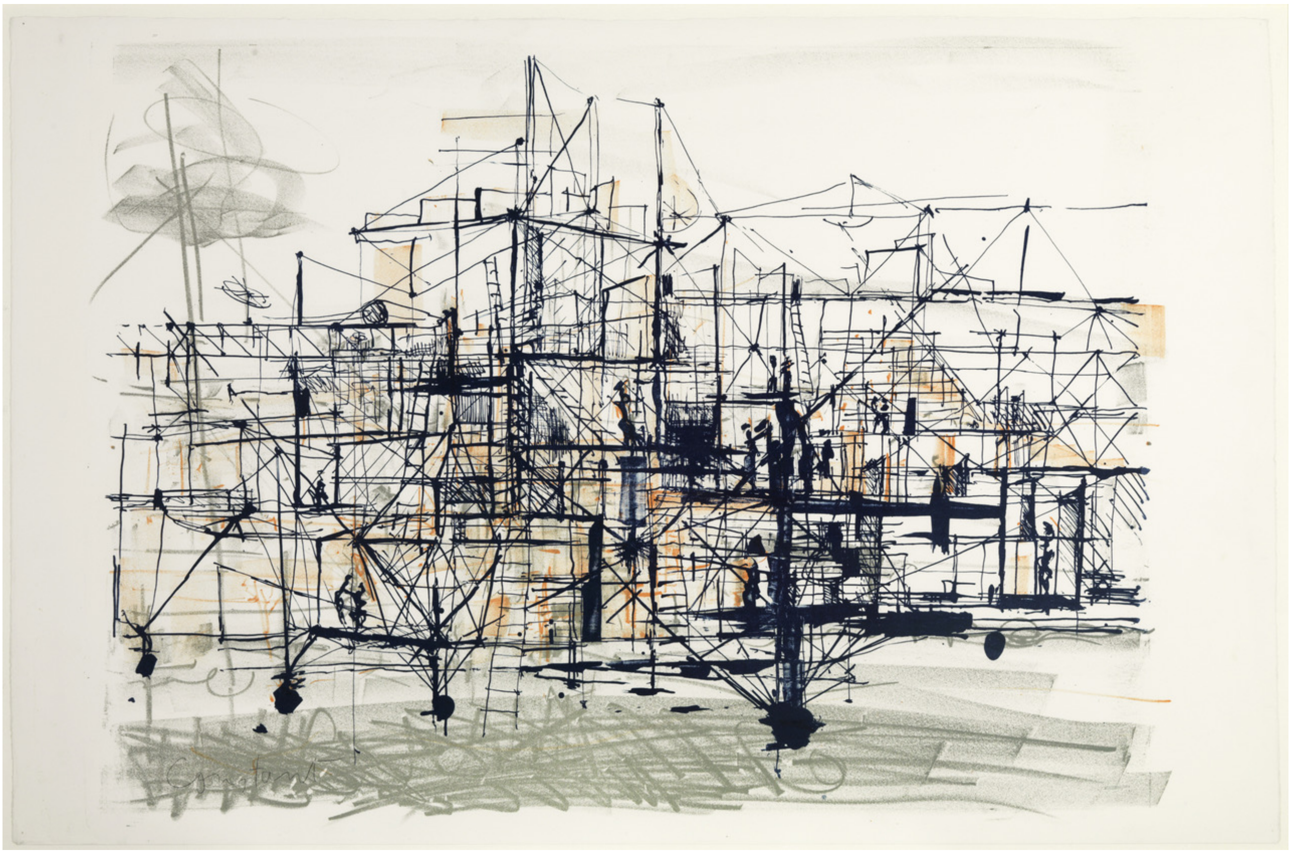
Applying Constraint Programming To Enterprise Modelling



Sytze P.E. Andringa

Applying Constraint Programming To Enterprise Modelling

THESIS

submitted in fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

Sytze P.E. Andringa

supervised by

dr. Neil Yorke-Smith

to be defended on Friday 2th July, 2021 at 13:30

Algorithmics Group
Department of Software Technology
Faculty of Electrical Engineering, Mathematics & Computer Science
Delft University Of Technology

Applying Constraint Programming To Enterprise Modelling

Author:	Sytze P.E. Andringa
Student id:	4377982

Abstract

Enterprise Modelling (EM) is the process of producing models, which in turn can be used to support understanding, analysis, (re)design, reasoning, control and learning about various aspects of an enterprise. Various EM techniques and languages exist, and are often supported by computational tools, in particular simulation. The goal of this thesis is to study the effects and advantages of applying constraint programming (CP) to EM. To the best of my knowledge, no previous study has explicitly combined EM and CP. On the topic of applying CP to EM, this thesis explains where it can be applied, as well as its requirements and advantages. Furthermore, it explains a possible approach where a neural network, trained on a simulation model that represents an enterprise model, is embedded into a constraint program. This approach is supported with experiments, that show typical business objectives can be embedded in a constraint program and find solutions to it in a multi-objective context. The main conclusion is that due to CP being a declarative programming technique, business constraints and goals can be effectively modelled into a constraint program, making the approach understandable and intuitive for business analysts to use. This thesis argues alternative approaches to apply CP to EM can also be realised. Some of these, as well as improvements over the proposed method, are also discussed.

Thesis Committee:

Title and Name	Function (in organisation)	Affilliation (section)
dr. Neil Yorke-Smith	Associate Professor	Algorithmics
dr. Theresia van Essen	Assistant Professor	Optimization
dr. Natalie van der Wal	Associate Professor	Multi-Actor Systems

Creditline cover image:

Constant (1920-2005)
Sketch for a Sector, 1966
Collection Fondation Constant
Photo: Tom Haartsen
©Constant / Fondation Constant c/o Pictoright Amsterdam 2021

Contents

Chapter 1

Introduction

Obtaining proper insight in the processes and routines of some business or organization is of great interest when making strategic decisions. It can be used to understand the present state and prepare for the future by providing insight into what strategic decisions will lead to what outcome.

Over the years business analysts have developed standardised tools and frameworks to assist in creating abstract models, aimed to represent enterprises effectively. The art of creating such models can be referred to as Enterprise Modelling (EM). More precisely, EM can be defined as the art of externalising knowledge in the form of models about the structure, functionality, behaviour, organisation, management, operations and maintenance of whole or part of an enterprise, or of an enterprise network, as well as the relationships with its environment [1]. EM methods are well-recognised for their value in describing complex, informal domains in an organised structure [2]. It is capable of answering questions that asses the structure of an enterprise, such as: [3]

- How does the enterprise work at present?
- Where do problems or challenges exist that necessitate changes in the enterprise?
- What are the requirements for these changes?
- What are the options for meeting these requirements?
- What criteria and arguments can be used to evaluate these options?

Technological advances in IT over the last few decades have had a large impact on how enterprise models are analysed. In particular simulation is used widely within the field of EM. By running a simulation model based on an enterprise model under various sets of parameters, one could get insight in how the enterprise would behave in unknown scenarios.

Applying simulation to EM is not straightforward. First of all, EM methods describe systems at a higher level of abstraction than software development does. EM is often used merely as a description and analysis tool. The result is that enterprise models have not always provided direct input for simulation models [2]. Next, although there are various studies that explain how to design a proper simulation model from an enterprise model (see Section 3.2.3), there is little standardisation on how to interpret the outcomes of the simulation to form strategic decisions. In a typical simulation approach, what parameters are fed into a simulation model is dependent on the problem to be solved. This is then often combined with data analysis, where data representations of the simulation outcomes provide the necessary input for a human to decide which strategy is best for the enterprise to take.

A promising programming technique which is studied very little in the context of EM is that of Constraint Programming (CP). CP is declarative and used to define discrete optimisation or satisfiability problems [4]. CP solves Constraint Satisfaction Problems (CSPs), which describe problems through variables and constraints. A solution to a CSP assigns a value to each variable such that every constraint is satisfied. A more exact definition can be found in Section 2.2. CP is seen by some as the closest approach to "the user states a problem, the computer solves it" [5]. CP allows the modelling of many real-world problems formulated in natural language to optimisation problems.

1.1 Motivation

There are various reasons to believe that it is beneficial to study how to effectively apply CP within EM.

First, CP solving is fundamentally different from simulation. Solving takes a theoretical approach by optimising parameters with respect to some objective function. On the contrary, simulation takes an empirical approach by repeatedly assigning random numbers to variables and compute the output of some function. Where simulation approaches aim to find the best possible solution by analyzing what inputs correspond to what outputs, CP approaches aim to find this by optimally solve some problem definition.

Second, CP is declarative and has an expression similar to natural language. A constraint program expresses *what* we want to solve, not *how*, as that is done by solvers. CP requires the programmer to focus on modelling the real-world problem accurately. Many EM leading frameworks are descriptive in nature [1] and can thus also be considered declarative. As such, it is reasonable to think that business goals and constraints can be expressed in CP programs naturally and intuitively.

Third, as far as my knowledge goes, there exists no previous study that explicitly combines CP and EM. Only closely related approaches, such as Enterprise Wide Optimisation, exist. We can see a trend over the last decade that more often optimisation techniques are being used to support EM. This raises intuition that although combining EM and CP is an unexplored area of research, many scientific gains are still to be achieved regarding how EM can be performed more effectively.

1.2 Research Questions

The main research question I aim to answer is *What are the effects of modelling an enterprise as a Constraint Satisfaction Problem (CSP)?*. This question is subdivided into three sub questions. The first two are *how* questions and aim to question *how* I intend to do parts of my research. The third *what* question is aimed on finding conclusions.

- How do we design a mapping that formulates EM aspects as a CSP?
- How do we decide whether an enterprise model is suitable for a conversion to CSP?
- What are the advantages of formulating EM as a CSP?

1.3 Structure

This thesis is structured as follows. Chapter 2 introduces EM and CP. Section 2.1 explains the current state of EM, with a focus on how EM is usually put in practice. Section 2.2 explains the current state of CP, with a focus on what can be done with this programming technique. Chapter 3 explains what it would actually mean if we were to combine the two fields, by explaining their differences and similarities between some closely related approaches towards combining them. Chapter 4 explains an approach that aims to use CP to help solving EM problems. This approach is put into practice in Chapter 5, where it is tested experimentally. Conclusive thoughts and recommendations for future work is presented in chapter 6.

The core answer to the research questions can be found in Chapter 3 and 4. Chapter 3 answers the research questions in a more general sense, whereas Chapter 4 does so for a particular approach. Fundamental background to understand the reasoning is provided in Chapter 2. Chapter 5 verifies the findings in those chapters experimentally. Summarised conclusions regarding the research questions can be found in Chapter 6.

Chapter 2

Background

2.1 Enterprise Modelling

Enterprise Modelling (EM) is the process of producing models, i.e. abstract representations, that can be used by humans or machines to support understanding, analysis, (re)design, reasoning, control and even learning about various aspects of interest of an enterprise [1]. It serves as a tool to support sense making of an enterprise's action system [6]. EM helps to capture the different elements and structures of an enterprise as well as to visualize the inter-dependencies between the elements. It focuses on addressing multiple perspectives of an enterprise in an integrated way and offers a set of practical guidelines for knowledge acquisition, modeling, and analysis.

More general information and sources about EM can be found in a recently published review by Vernadat [1]. An overview of the history of EM can be found in work by Vallespir and Ducq [7]. Throughout this section, I will refer to some figures that can be found in Appendix A. These figure originate from other literature. They were included to provide examples of the structure and common concepts found in EM.

2.1.1 EM Principles

EM frameworks started to develop somewhere in the 70's. At the end of the 80's, standards began to develop, resulting in less new frameworks and methodologies appearing (see Figure 2.1). Languages inspired each other which caused used concepts to reach a more relative stability. This transformed the field of EM from a brainstorm era to a convergence era [1]. EM principles began to appear and were documented in various ISO standards. Nowadays, it is possible to refer to basic principles which are believed EM languages and frameworks should rely on. These are captured in ISO Standard 19439. A good understanding of these principles is important if we want to develop a tool to support EM, as we do not want this tool to violate commonly used EM principles. Commonly used EM principles are [1]:

- *Plural nature of enterprise models.* There does not exist a single enterprise model that is able to express all aspects of some enterprise. As such, in order to get a good understanding, several models are necessary. The resulting enterprise model will then be a sum of these models.
- *Principle of minimalist ontology.* Any EM language must be defined by a minimal and non-redundant set of modelling constructs, each being made of a minimal set of attributes and properties describing the essential nature of the concept modelled.
- *Concept of modelling views.* Models are created based on some modeling view. An efficient EM framework should provide a minimal set of non-overlapping views to cover all aspects.
- *Concept of modelling levels.* A modelling level represents an abstraction level for a given population of users, in terms of the amount of details that should be taken into consideration.
- *Separation of enterprise behaviour and enterprise functionality.* Enterprise behaviour describes how the enterprise performs its work. Enterprise functionality describes the actual work to be performed.
- *Late binding of resources to process steps.* Resources are not a priori bound to process steps, since these are assigned to activities at run-time. This means that from a modelling perspective, one should first focus on *how* a process is performed, before it considers *what* the input of the process.
- *Fundamental types of flows.* EM languages should make a distinction between three types of flows: material/physical flows (representing physical objects), information/decision flows (information objects) and control flows (flows describing the logical sequencing of process steps).

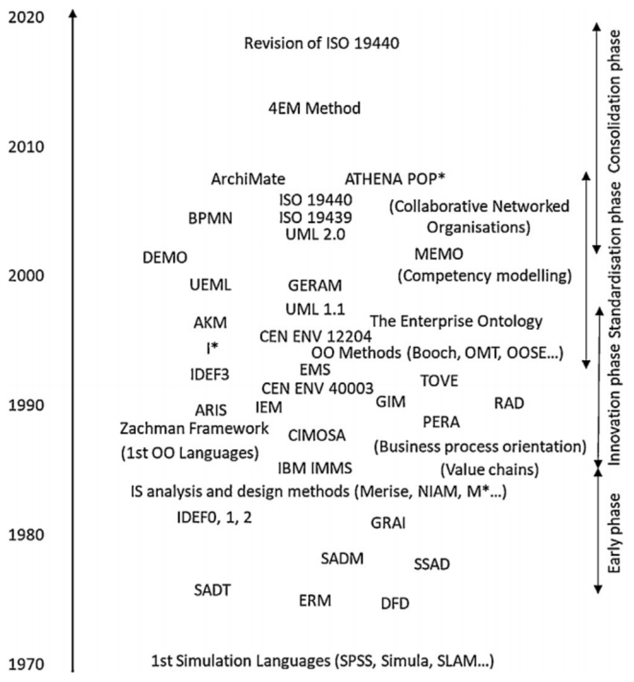


Figure 2.1: Chronological panorama of EM methods, languages and frameworks (not exhaustive) [1].

2.1.2 Frameworks

Many EM frameworks exist, varying widely in their perspective and abstractness. Some frameworks, such as the organizational modelling language OperA [8] or the social modelling language i* [9] take a Multi Agent System perspective. Such frameworks require little extra information to allow simulation. Others, such as MEMO OrgML or 4EM are more restricted to formal aspects and have a more conceptual and descriptive format. These have a closer business modeling perspective, putting more focus on high-level relationships and tend to be more descriptive. However, many of these approaches seem to lack lateral linkages (mechanisms to coordinate horizontally, for example how teams from the same hierarchical level cooperate), which can limit their information processing capabilities [6].

A comparison between some leading frameworks is presented in Table 2.1. The main similarity between them can be traced back to the main principles discussed in Section 2.1.1. Each framework uses common EM concepts, such as layers, views, elements, flows, behaviour and functionality. However, the frameworks differ on how the decomposition is performed, as they differ on what set of layers, flows and elements they use. As such, a proper EM framework should be picked that matches the application. For example, DEMO is based on Enterprise Ontology and can be useful for applications where it is crucial that communication is modelled accurately. ArchiMate is able to model large enterprises effectively, due to its rich set of tools, elements and layers. 4EM has a focus on how the modelling should be performed, and can therefore be particularly useful in a teaching context. MEMO emphasises the modelling of multiple perspectives, hence its name. It can be useful when perspectives of an enterprise differ, for example when a marketing department has vastly different perspective than its quality control department.

For a more thorough comparison I refer to existing literature. Ettema and Dietz [10] describe an approach that combines ArchiMate and DEMO. Kinderen and Kaczmarek [6] compare a few modelling languages and describes how these satisfy requirements found in business scholar literature. A comparison of Multi-Agent-System frameworks in the context of simulating organisations is presented by Merkt and Wagner [11].

Framework	Background	Key Concepts	Current status
DEMO	Development during early 90' at TU Delft. Introduced in 1996 by Dietz [12].	Focus on Enterprise Ontology, where the enterprise is a social system and all processes are described as being communicative transactions. Has therefore less focus on industry processes. Considers three different abstraction layers. From these layers, various diagrams are constructed (see Section 5.1.2).	Used widely by organizations and research. Maintenance of the framework is carried out by the Enterprise Engineering Institute. Has been studied in the context of simulation.
ArchiMate	Developed during 2002 - 2004 by the Telematica Research Institute and several partners from government, industry and academia.	Focus on enterprise architecture. The full framework considers 6 layers and 4 aspects used to classify enterprise elements (see Figure A.2).	Widely used by organizations and academia. Certificate held by The Open Group [13], which consists of several businesses. Richness of tool extensions. Simulation studies exist.
4EM (For Enterprise Modelling)	Introduced in 2014 by Sandkuhl et al. [3]. Based on its predecessor, Enterprise Knowledge Development (EKD).	High level and descriptive structure. Aims to synthesise the best principles and practices of existing frameworks. Output is most often in graphical notation. Considers three core elements (Figure A.3) and six different submodels (see Figure A.4). Includes methodology on how modelling should be done, namely by participatory discussions, involving all actors and stakeholders of the enterprise reengineering process.	Most work is in academic context. Relative new method, little tool support.
MEMO OrgML (Multi-Perspective Enterprise Modelling))	Introduced in 2002 by the University of Duisburg-Essen [14]. MEMO OrgML, used for organizational modeling, had a major revision in 2011 [15].	MEMO is a family of languages, each aimed at representing specific perspectives and aspects of an enterprise [16]. As such, it has a focus on modeling from multiple perspectives. Models remain descriptive and the output most often in graphical notation [1]. Complex structure of meta-models.	Latest method overview published in 2012 [17]. Comes with a variety of tools.

Table 2.1: Overview of a some leading EM frameworks.

2.1.3 Enterprise Modelling Problem

One central responsibility of managers is to deal with problems [18]. Their work involves detecting and gaining insight into problems in enterprises to in turn decide what the best course of action would be. In order to support the crucial problem structuring phase of this extensive problem solving processes, problem structuring methods have been developed. The basic idea of these methods is to use forms of conceptual modelling to facilitate communication among problem stakeholders. EM is often consulted to help solving some problem an enterprise is dealing with, as it gives insight in the functioning of an enterprise. However, many existing EM methods provide only tenuous concepts of 'problem', or they offer no explicit modelling concepts to delineate 'problems' at all. This problem is discussed in a study towards a more expressive problem structuring method within the context of EM, by Bock et al. [18].

CP solves decision problems. As such, it can be argued that effectively applying CP to EM should be done by translating the problem the EM is based upon – the Enterprise Modelling Problem (EMP) – to a CP model. This translation, from EMP to CP, is discussed in more detail in Section 3.1.1. This section focuses on the definition of an EMP, what format it can take and what we can expect from it. Bock et al. [18] discusses three conditions for something to be a problem.

- *Value-related unsatisfactoriness of the situation.* The current situation is unpreferred or "judged as negative" [19]. From another perspective, it can also be said there is "something to be desired".
- *Desirability of action.* Action to change the situation is desired. If no action would be desired, then "no one perceives [...] a need to determine an unknown" and "there is no perceived problem" [20].
- *Uncertainty as to possible and appropriate action.* There is a perceived uncertainty as to what course of action is possible and appropriate to make the situation more satisfactory.

Summarised, a problem can be defined as *a perceived situation that is composed of a number of interrelated factual, value, and contingent aspects, and which, by some individual, is perceived to be unsatisfactory, for which action to change it is regarded as desirable, and for which uncertainty is experienced as to possible and appropriate action to make it more satisfactory* [18].

The proposed metamodel by Bock et al. [18] is shown in Figure A.5. This forms a more detailed problem conceptualisation than methods found in EM frameworks. An example of it being put into practice for a middle-sized construction company is shown in Figure A.6. The presented approach has resemblance with concepts found in CP. First, it can be observed from Figure A.6 that the objective – analysis of the desired state – can be expressed as a multi-objective optimisation problem, as all its objectives correspond to minimising or maximising metrics, such as material request duration, stock levels, reputation and project delivery timeliness. Some CP techniques have been developed to effectively deal with multi-objective optimisation problems, for example through combination with evolutionary algorithms [4]. Second, the approach makes use of the concept constraint. Values of factual aspects are determined by whether their corresponding constraints are satisfied or not. Constraints are a core part of CP. Third, the metamodel in Figure A.5 describes that a solution is composed by courses of action. In CP, a solution is defined by the value of decision variables.

2.2 Constraint Programming

CP is a declarative programming technique. It allows one to describe many real-world problems through constraints, that is, statements which pose some relation among the problem's variables [21]. Constraints are defined using a logical formalism, in a way similar to functional or logic programming [22]. CP is used in a wide variety of applications. The first successes of CP were in areas of scheduling, transportation, personnel assignment and resource allocation [22]. Other applications are network management, controlling electro-mechanical systems, constraint based spreadsheets, interactive problem solving, graphical interfaces and solving overconstrained problems [21]. CP is a two stage process. First, a Constraint Satisfaction problem (CSP) is programmed. As a second step, a solver is executed on the CSP in order to find a solution to it. A Constraint Satisfaction problem (CSP) can be defined in the following compact manner [23]:

A constraint satisfaction problem (CSP) consists of a finite set of variables $V_1, ..., V_n$ of finite domains $D_{V_1}, ..., D_{V_n}$, and a finite set of constraints, where a constraint is a relation over the set of variables. Constraints can be extensional in the form of a set of compatible tuples or intentional in the form of a formula. A solution to a CSP is a complete assignment of values to variables satisfying all the constraints.

Unlike basic primitives of other programming languages, constraints do not specify computing operations, but rather the properties of a solution to be found [24]. This results in a clear distinction between "logic" and “control”, which is often lost in other declarative language implementations [22]. In CP, constraints have two facets, definition and behaviour. Once the definition is correct, the behaviour can be sorted out. The practical consequence is that the programmer can concentrate on modelling her problem and the problems with performance and termination can be ironed out afterwards [22].

An extension of CSP is that of *Constraint Optimisation Problem (COP)*. COP are similar to CSP, however, instead of finding a solution that satisfies all constraint, COP aim to find a solution that optimally solves some objective function that satisfies all constraints.

Classical CSP are composed of hard constraints, which implies that every constraint has to be satisfied. As such, classical CSP are not able to model constraints that are preferences rather than strict requirements or to provide a not-so-bad solution when the problem is overconstrained [24]. As a solution, many extensions of classical CSP exist that make use of soft constraints. A soft constraint is a condition that the user would prefer to see satisfied rather than not satisfied. The user is prepared to accept not satisfying it because of the cost of satisfying it, or because of conflicts with other constraints or goals. In case a user has multiple soft constraints, there is a need to determine which of the various constraints should take priority if there is a conflict between them or if it should prove costly to satisfy them [23]. Weighted CSP, probabilistic CSP and fuzzy CSP give a more specific interpretations to soft constraints, whereas semiring-based constraints and valued constraints are more generic extensions to soft constraints [24].

Chapter 3

Combining EM and CP

This chapter aims to describe how to effectively apply CP to EM. It discusses how to decide whether an enterprise model is suitable for conversion to a CSP by discussing the abstraction level EM and CP operate on. Next, it also discusses how a mapping – from EM to CP – can be realised, by explaining what difficulties can be encountered. This chapter does not discuss a particular approach for a pipeline that uses an EM to map an EMP (see Section 2.1.3) to some course of action that is best to take. Instead, it discusses on a higher level how CP and EM can be combined. A particular approach for a pipeline that realises such a mapping is discussed by Chapter 4.

3.1 Similarities and Differences Between EM and CP

Although EM and CP come from different sciences, various similarities can be found. For instance, both do problem solving by means of creating models, making them both interested in how to define problems and creating model representations that reflect the real world in some way. Where EM creates an enterprise model, CP creates an optimisation model. Similarly, it can be said that the EMP equivalent of CP is its objective function. In that sense, if we were to apply CP to EM, a natural way to do so would be to formulate aspects of an enterprise model to an optimisation model, and map enterprise modelling problems to objective functions. Table 3.1.2 shows an overview of some EM aspects and their CP equivalents.

3.1.1 Problem Statement

Enterprise modelling can be seen as a tool to represent an enterprise in some format in order to detect and tackle some underlying problem in an enterprise. EM is not a goal by itself, but rather a tool to provide knowledge about the impact of decisions being made. Often, an enterprise model is made with a goal in mind. As a result, which enterprise elements need to be designed in detail and which not necessarily have to, can vary on the purpose of the model. For example, if we want to optimise the output of our machines, we care less about modelling the customers and more about modelling the machines. Vice versa, if we want to optimise a marketing department, we would probably focus on modelling customers in great detail. This does not imply a new enterprise model should be made for each EMP, as many EM elements can be reused. Instead, it implies that models representing the same enterprise might differ based on their purpose.

CP is by some seen as the closest approach to "the user states a problem, the computer solves it" [5]. CP has a formulation which solves in the format of "considering we know X, what is the solution S that contains variables V that optimally satisfies objective Y". This same question can be used in an EM perspective, and could help inspecting the relationship between EM and CP.

Figure 3.1.2 shows that objective Y has a similar format both for EM and CP. EM deals with business goals, whereas CP deals with objective functions. Both can be expressed in a format that has close relationships to natural language. For example, a goal can be to maximise throughput, maximise the number of satisfied clients or to satisfy as many preferences of some list as possible. As such, their abstraction level is similar, which results to little information being discarded with respect to a formulation expressed in natural language.

The format of the solution S requires a bit more thought, but can usually be derived from natural language rather direct. Changing S should affect the outcome of the system. It requires knowledge of the parameters we can adjust. In EM, this can be referred to as courses of action or leverages. In CP, this corresponds to decision variables.

However, X – the model describing real world aspects – has a vastly different format in CP and EM. In CP, this has the form of constraints and parameters. In EM, X is represented by an enterprise model paired with historical data. The format of X differs because EM and CP have different evaluation methods to find S. Where CP is rather straightforward in how to find S, namely by using solvers that look for an optimal solution, EM

EM	CP
Enterprise Model	CP Model
Business Goal	Objective Function
Course Of Action	Decision Variable
Business Constraint	CP Constraint
Hypothetical Scenario	Set of Input Parameters

Figure 3.1: Overview of some EM terms and their CP equivalents

can have different approaches. A valid and commonly used way of evaluating an EM model that is very different in the sense that it does not use IT tools, would be to discuss the model in a meeting while taking notes with pen, paper and a whiteboard.

This shows that if we would like to use CP as our evaluation method, in other words use solving techniques on various parts of the EM, it requires X to be in some format that corresponds with CP. It requires us to somehow convert an enterprise model into a set of constraints. Such a conversion is not natural: EM and CP vary widely in abstraction level. Finding a suitable abstraction level is important, since we would like essential information to be kept intact but also do not want an overly complex model that is infeasible to solve by means of CP.

3.1.2 Model Abstraction Level

EM and CP vary widely on abstraction level. Enterprises are highly dynamic with lots of interacting subsystems. Modelling such a high dynamic and complex socio-technical system requires good understanding of what abstractions are reasonable to make. EM aims to capture enterprises both tractable and expressive: enough to allow processing of the model, and expressive enough such that the analysis makes sense. Expressiveness vs tractability is a trade-off [25] and to help finding a nice balance, standardised tools and methodologies have been developed. The most abstract models are designed to merely provide a high level descriptive notation. Such models are less prone to errors, but lack specificity due to their ambiguity. Apart from that such models are difficult to simulate, an unwanted side effect can be that someone consulting the model could have an interpretation of the enterprise that differs from the modeller's.

Recognizing a too abstract model is difficult. Humans can be ignorant, and are not always able to point out lack of detail in the model. If a modeller is unaware multiple interpretations are possible, a model can become inaccurate. Inaccurate or incomplete models are not always bad, as it allows humans to discuss and assess models without the need of implementing all the details. However, sometimes more details are required to allow proper discussion and analysis, and due to human ignorance this lack of details can stay unnoticed. This problem can be tackled by having experienced modellers. Due to their experience, professionals tend to be more aware of what information a model has and lacks. From a modellers side, this allows them to effectively add and remove details of the model. From a reader side, being experienced allows insight in knowing what type of analysis can and can not be performed given this model.

It is infeasible to assume models are exclusively made and read by experts. Domain experts often perceive the business in imprecise ways and may or may not have the expertise to capture their knowledge in a conceptual model. Professionally developed modelling frameworks can help tackling this problem. Frameworks require the modeller to provide the input in a particular way such that a particular type of analysis can be performed. Strict rules on the model results in a limited but accurate analysis, and loose rules result in a wide but less accurate analysis. However, too strict rules are also not desirable. Many current EM methods and tools tend to be formal and inflexible, and are often designed for automated model processing rather than for helping business professionals understand business situations, resulting in a state of EM where many enterprise models can only be understood by experts [26]. How to effectively relax modelling criteria to produce more usable EM methods is discussed in more detail by Bork and Alter [26].

Humans and computer differ in what level of detail they perform best on. Where humans excel at high level descriptive models, computers perform better on low level detailed models. Therefore, it would make sense that a model made to be analysed by computers is more detailed than one that should be analysed by human. Simulation deals with converting enterprise models, formulated in a descriptive format, to a model that can be analysed computationally. As such, it would make sense that the requirements for an enterprise model to be suitable for simulation are similar to that of optimisation.

A feasible mapping, from EMP to solution, should have proper abstraction levels at each stage. EM and CP operate on a different abstraction level, and as such, suitable abstraction methods should be consulted. This problem corresponds to the trade-off of expressiveness vs tractability.

3.2 Closely Related Approaches

As stated earlier, to the far of my knowledge, no previous study explicitly combines CP and EM, and only closely related approaches exist. This section explains a few of these approaches, as well as some work that is deemed relevant for the scope of this thesis.

3.2.1 CP and ML

As discussed in Section 3.1.2, there is a need to abstractify EM such that it becomes possible to solve (parts of it) through CP. Fields that are concerned in finding suitable abstractified models for data are Machine Learning (ML) and Data Mining (DM). Combining CP with ML and DM has recently developed itself into a hot topic of research. If combined effectively, it can be used to improve the solving of constraint problems, for example by feeding a constraint program with data, such that parameters can be predicted more accurately. For more articles on this topic I refer to an issue of *Artificial Intelligence* from 2017 [27].

Particularly relevant for this thesis is a work of Lombardi et al. [28], where two types of machine learning models, namely Artificial Neural Networks (ANN) (equivalent to NN) and Decision Trees (DT), and four Combinatorial Optimisation approaches, namely Local Search (LS), Mixed Integer Non-Linear Programming (MINLP), CP and SAT Modulo Theories (SMT) are combined and compared to each other. Except for ANN with SMT and DT with MINLP, each combination was tested. It is said here that CP has been shown to be particularly effective, due to the expressive modelling language and the filtering algorithms. This claim was based on experiments regarding a thermal-aware workload dispatching problem. The authors claimed that CP in combination with NN has good performance compared to other combinations of optimisation methods with machine learning techniques.

3.2.2 Enterprise Wide Optimisation

Enterprise models and supply chain models are related in the sense that they both model routines and processes while adhering to rules found in enterprises. Such enterprise routines, processes and rules are described in an enterprise architecture, which can be interpreted as a description of how organisational units interact with each other, i.e. exchange information and resources to collaboratively execute some task.

An area that highlights the relation between enterprises and supply chains is that of enterprise-wide optimisation (EWO), which aims to optimise the operations of supply, manufacturing and distribution in a company [29] by formulating the enterprise as an optimisation model. EWO has a significant overlap with supply chain management and lies at the interface of chemical engineering and operations research. Grossmann [29] explains the relation between EWO and CP in detail. More recent works of the same author discuss the advances and challenges in the application of mathematical programming in EWO [30, 31].

Although EM and EWO are different, studying EWO thoroughly might form a basis to formulate enterprise models as CSP. One EM principle, referred to as the *plural nature of enterprise models*, states that no single enterprise model is able to describe the enterprise as a whole, and thus multiple models should be considered (see Section 2.1.1). Therefore, we can argue that EM goes *further* than EWO by not focusing solely on the supply chain model by also taking non-industry models into account. This corresponds with the difference between EM and Business Process Re-engineering (BPR): BPR is considered to be one-dimensional whereas EM is considered to be n-dimensional. These extra dimensions could for example describe employee relations, information flow, management and hierarchy. Stated differently, EWO can be seen as a sub-problem of EM.

Several studies aim to add extra dimensions to EWO. For example, Sharma et al. [32] presents an integrated framework to optimise enterprise management, where three functionally dependent layers need to share information for a complete optimisation of the model. Each level optimises over a different time horizon with respect to a different objective.

Other examples aim to integrate ontology into EWO. For example, Muñoz et al. [33] explains a system where knowledge management is integrated into the system to detect whether production schemes are in the database. A limitation of this approach is that it considers the system to have only one "brain", in the sense that no information is transferred among individual processing units, since they use the same database. This assumes that communication within an enterprise happens instantaneously and error-free: no misunderstanding can be possible since each component always has same information available. A second example, where an ontological approach for planning and scheduling in primary steel production is explained, is presented by Dobrev et al. [34]. A more general study that explains a concept-object oriented modelling approach for EWO is presented by Munoz et al. [35], which has as its ultimate goal the development of intelligent agents for supporting EWO at a scheduling level.

3.2.3 Simulation

EM coupled with computer simulation can be a powerful system analysis and evaluation technique as it can help finding complex and emergent properties in a system. Simulation provides a descriptive model predicting how a system will behave under given conditions by adding a time dimension often missing in many EM approaches. Simulation engines have been incorporated in some EM tools, for example in ARIS ToolSet, IEM and FirstStep [1].

Creating simulation models of enterprise models is complex due to the large number of abstractions that have to be made. Simulation models need to encode both static and dynamic properties of systems, and their complexity grows rapidly as either the scope or depth of modelling increases. Enterprises are extremely complex systems and not all enterprise models are suitable for simulation. The practicability of a simulation model is dependent upon whether a suitable level of modelling abstraction can be made such that the problem being tackled can be modelled effectively.

Various studies exist that describe how simulation should be used in the context of EM. A survey with a focus on Multi-agent-system approaches is presented by Merkt and Wagner [36]. Methodology for simulating conceptual models, of which EM is a sub-category of, is discussed by Robinson, Robinson [37, 38]. A methodology for certification of modelling and simulation applications is presented by Balci [39].

An often used technique of designing simulation models is by using state-machine like structures, such as petri nets or system dynamics. This allows to simulate the flow of quantities through parts of a system over time in a structured manner. System Dynamics does this in a dynamic and continuous setting by using flows and stocks, whereas petri nets take a more discrete approach through the use of tokens. A comparison between the two is presented by Duggan [40]. Various studies exist that aim to translate enterprise models to a state-machine model [41, 42, 43, 44]. Such translations are very useful in the context of simulation since they provide structured and standardised ways of performing simulation experiments. However, such translations are less useful in the context of CP, as solving complex state machines through optimisation can become an infeasible process. To argue this statement further, I will go a little deeper into petri nets, explain the issues of formulating a petri net schema as a constraint program and demonstrate it by an experiment. This argues the need of abstraction if one would convert an enterprise model as a CSP.

Petri Nets

The concept of petri nets has been used in various EM methodologies. They have a general and simple control flow design, which allows intuitive mapping towards simulation models. Petri nets have an exact mathematical definition of their execution semantics. Two relative old methods, namely MERISE and M*, were both based on petri-net formalisms [1].

Vejrazkova and Meshkat [43] explains a mapping to a standard petri net. Here, various elements and common structures found in DEMO were put next to a petri net equivalent. Fatyani et al. [44] extends on this by introducing a mapping from DEMO to colored petri net. Colored petri nets assign colours to tokens, which are in turn used by the transactions in the model. The main benefit with respect to EM is that these colours can be assigned to properties to be measured in the simulation results.

Numerous studies exist on how petri nets can be useful to solve a form of optimisation problems namely linear programming problems. However, since methodologies exist that translate EM to petri nets, I am interested in the opposite direction: how to solve petri nets by means of linear programming. Such an approach is studied by Nakamura et al. [45]. Here, the authors provide a linear programming model for a autonomous (i.e. standard) petri net, a timed petri net and a colored petri net. The use case here is the solving of travelling salesman problem instances.

It is noteworthy that Nakamura et al. [45] focuses on deterministic petri nets, and does not consider ambiguity either on parameter, i.e. the input or solution, nor on token transitions. We previously discussed that modelling uncertainties is of importance in the scope of EM. Adding uncertainty would add another layer of computational complexity, and the resulting models might become infeasible to solve using optimisation techniques. Although adding uncertainty to input parameters can be done by for example considering the confidence constraint by Mercier-Aubin et al. [46], removing the deterministic solving (or even more complex: considering uncertainty in state transition probabilities) results in solving where tokens can be at several locations at the same time. Such uncertainties regarding transitions are no problem for simulation environments, as they can utilise Monte Carlo techniques to allow proper analysis of the model.

Petri Nets Experiments

To evaluate the usefulness of petri nets in the scope of combining CP and EM, I setup an experiment that aims to solve a petri net through CP. If CP would be able to solve complex petri nets within reasonable time, translating an EM to a petri net and solving the petri net through CP could be an effective way to solve an EM through CP.

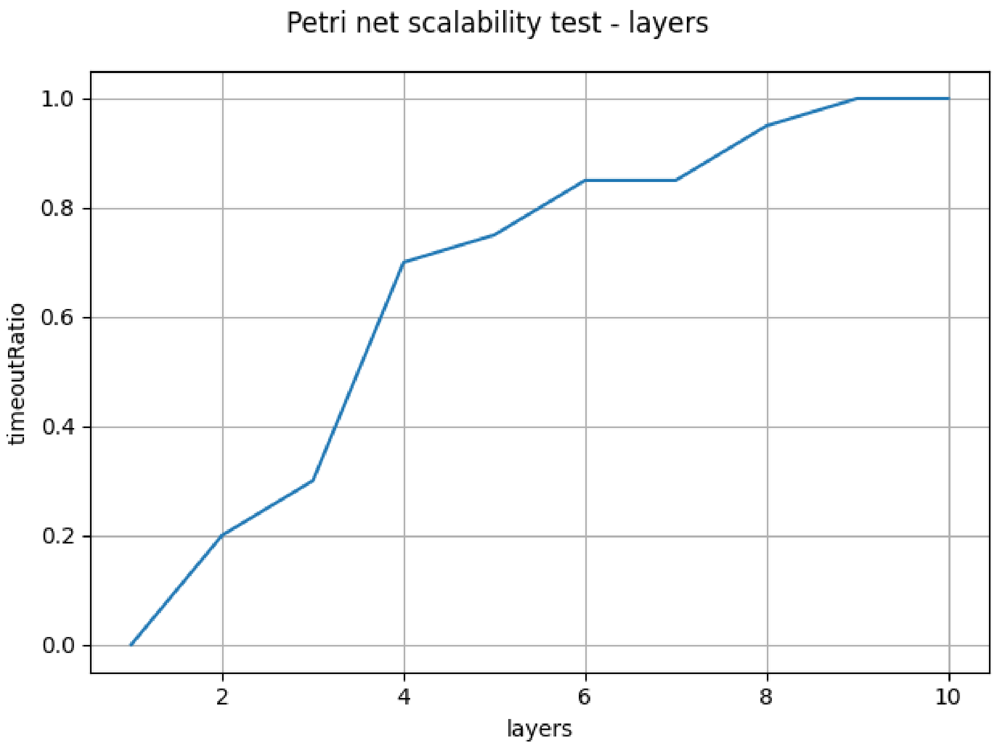


Figure 3.2: Scalability test for solving petri nets by means of CP. The percentage timeouts (set at 60s) was plotted against the number of petri-net layers. 20 petri nets were generated for each layer. The number of nodes and transactions per layer was randomly decided by drawing from a normal distribution with a mean and variance set at 3.

The experiment was setup as follows. I designed a CP program that is able to take a petri net formulation and solve it. Solving a petri net here refers to finding a stabilised state for which it is no longer possible to execute any further transactions. The petri nets were randomly created through a script through layers, such that the first layer passes on tokens to the second, the second to the third, etc. Each layer consists of a random number of nodes. The number of transactions between the layers were also created randomly, but to ensure solving is possible, a condition was added such that each node had to be connected to at least one transaction. The objective was set to maximise the amount of tokens at a random place in the last layer. Code used for this experiment can be found on GitHub [1].

Figure 3.2 shows how runtime scales with the size of the petri net. Here, each layer had three nodes. It shows that almost every perti net with more than 4 layers will occur in a timeout. Furthermore, it shows that timeouts also occur for small amount of layers.

There are multiple observations that can be made from Figure 3.2. First, solving time scales with the size of the petri net. Second, the large variance among the samples indicate the solving time is clearly dependent on the structure of the petri net. Third, the runtime is more dependent on the number of transactions than the number of nodes in the petri net. At last, many instances, even at small size, were unable to be solved within a reasonable time.

A petri net representation of an enterprise model can get very complex. The DEMO mappings by Vejrazkova and Meshkat, Fatyani et al. [43, 44] use multiple petri-nodes and petri-transactions per DEMO transaction. For example, if we would use the mapping of Vejrazkova and Meshkat [43], the petri-net equivalent of the standard transaction pattern, a DEMO-transaction that can be either accepted or rejected, would include 21 petri-nodes and 21 petri-transactions. We also have to take into account the complexity of the net is increased even further if we would including uncertainty in path traversal, which was not considered in the experiments but is part of the standard transaction pattern. All in all, I would argue that solving perti net representations of DEMO models by means of CP is unfeasible, due to its inefficient scaling with respect to the runtime. Therefore, in the approach presented in Chapter 4, the complexity of the enterprise model is reduced through an abstraction method – a neural network – to in turn allow feasible solving of the enterprise model.

[1] https://github.com/sytzeandr/em_to_cp

Chapter 4

A Function Estimation Based Approach

The previous chapter sketched ideas and possibilities of how CP can be applied to EM. This chapter discusses an approach, which is used for the experiments discussed in Chapter 5. This chapter discusses how a mapping that formulates EM aspects as a CSP can be designed, which refers to the first research question. Its advantages, which refers to the third research question, are discussed in Section 4.4. The approach can be considered rather general and leaves room for many improvements to be made, which is discussed in more detail in Section 6.2.

An overview of the approach is shown in Figure 4. First, a simulation model is designed based on the enterprise model. Then, simulation data is created, on which we train a simple Neural Network (NN). Next, an EMP (see Section 2.1.3) is setup that reflects business goals. The trained NN as well as the EMP are then consulted to create a constraint program. Then, by utilising soft constraints and various weight and threshold parameters for the various objectives functions, an approximation of the pareto front is formed.

This approach falls under the category of Function Estimation based Approach (FEA), which is a subcategory of Analytical Model Enhancement, a Simulation-Optimisation Approach [47]. AME uses simulation results to enhance some analytical model. FEA estimates a function that describe the relationship between particular input and output variables. In my case, the function trained is a NN. More about subcategories of simulation-optimisation is explained by Figueira and Almada-Lobo [47], and in Section 6.2.1.

Throughout this chapter, a simple restaurant will be used as a running example. This restaurant buys resources periodically according to some buying strategy and processes it to various types of food. Such a buying strategy can be represented by a float per resource, indicating the quantity of that resource being bought each period. There are some constraints on the possible buying strategies, for example, we cannot order fresh non-seasonal fruits or vegetables. Resources can become spoiled if stored for too long. Orders are predicted to come in according to some given probability distribution, based on historical data. There are two main goals to achieve: on one hand we do not want to buy too many resources to reduce spoilage and on the other hand we want to buy enough resources to process as many orders as possible. The restaurant is interested in the effects of their buying strategy on these two objectives.

4.1 The Simulation Model

As discussed in Section 3.2.3, simulation is widely used to analyse enterprise models. Effective use of simulation models requires them to represent the enterprise model as good as possible. Various simulation modelling techniques and studies exist that can be referred to (see section 3.2.3). As such, the rest of this chapter assumes that the simulation model is designed properly.

It should be noted that the EMP is not taken into account for choosing what inputs to send to the simulation model. In other words, the values that are sent to the simulation model are independent on what problem we aim to solve. Even when the structure of the EMP is known before executing the simulation model, I would not recommend choosing inputs that correspond to the EMP, as that would limit ourselves from solving various EMP using the same simulation data set.

After the model has been setup, it should be ran a sufficient large number of times under various inputs. How many executions is sufficient depends on the complexity of the solution space. As this is very case-dependent, it can best be determined experimentally by analysing where the performance of the resulting NN does not increase significantly by adding more data samples.

For our restaurant case, this means setting up a simulation model of the restaurant. Incoming orders can be modelled based on the known probability distribution. The input should consist of various buying strategies. The output should include at least two observables, namely the percentage of successful orders and the percentage of spoiled resources.

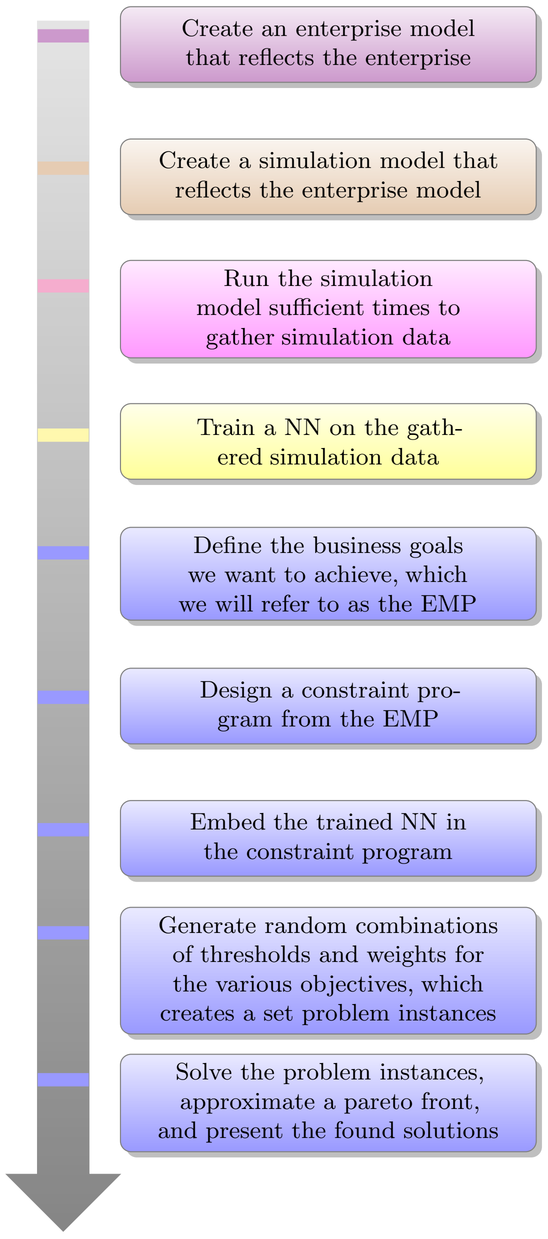

Create an enterprise model that reflects the enterprise
Create a simulation model that reflects the enterprise model
Run the simulation model sufficient times to gather simulation data
Train a NN on the gathered simulation data
Define the business goals we want to achieve, which we will refer to as the EMP
Design a constraint program from the EMP
Embed the trained NN in the constraint program
Generate random combinations of thresholds and weights for the various objectives, which creates a set problem instances
Solve the problem instances, approximate a pareto front, and present the found solutions

Figure 4.1: Diagram that shows the workflow of the presented approach.

4.2 The Neural Network

A Neural Network (NN) is able to capture relationships between data entries, to in turn predict the outcome. In my case, the NN is trained to predict behavior of the simulation model from simulation data. As discussed in Section 3.2.1, Lombardi et al. [28] claimed that CP in combination with NN has good performance compared to other combinations of optimisation methods with machine learning techniques.

A NN should be able to handle a large variety of simulation models for several reasons. First, NN are able to deal with various data encoding techniques found throughout machine learning, which helps us deal with many different formats the in- and output can have, such as applying one-hot encoding to deal with categorical data. Second, NN are able to learn behaviour of opaque or very complex systems, without requiring detailed knowledge of their components and interactions [48]. This allows the NN to learn complex properties found in simulation models, such as emergent and non-linear behavior.

A NN consists of several layers. The first layer is called the input layer and the last layer is called the output layer. The remaining layers are called the hidden layers. Each layer consists of several to many neurons. A single neuron is represented by a non-linear function with vector input $\bar{x}$ and scalar output y according to Equation 4.1. Here, x_i is a single component in $\bar{x}_i$, ϕ the neuron *activaty* (usually represented by some activity function), b its bias and w_i the weight. w_i and b are usually trained, in our case on simulation data.

$$y = \phi\left(b + \sum_i w_i x_i\right) \tag{4.1}$$

The NN architecture used in the presented approach can be described as follows. The NN consists three hidden layers, each with a width set at $\sqrt{N^2 + M^2} + a$, where N, M are considered to be the amount of input and output neurons. a can be seen as a parameter to deal with datasets that have few in- and outputs. Each hidden neuron consists of a weight parameter, a bias parameter and a ReLu activation function. For my experiments, I set a to 4, as it showed good performance. Further optimisation of a is possible, but since the goal is to provide a general design, optimising a was considered to be out of the scope for this thesis.

The output layer uses an activation function dependent on the format of the simulation output. This was set to be either ReLu (when the output is strictly larger than 0), Sigmoid (when the output is between 0 and 1), tanh (when the output is between -1 and +1) or softmax (when all outputs are between 0 and 1 and also sum to 1). No activation function was used on the output layer if the output had no upper or lower bound. Training was done in batches of 64 samples using AdamW [49], with a learning rate set at 10^{-5}. For example in our restaurant case, since both outputs are represented by percentages and can thus be represented as a rational number between 0 and 1, a sigmoid activation function is used on the last layer.

4.3 The Constraint Program

The goal of the CP model should be to outline what strategical decisions will lead to the preferred outcome. The CP model should therefore have close correspondence to the EMP. Decision variables should match with levers, constraints with the environment and objective functions with observables. The CP model was implemented in the MiniZinc language [50] with help of MiniBrass [51], an extension to MiniZinc to implement soft constraints.

4.3.1 Embedding NN into CP

Bartolini et al. [48] explains how a trained NN can be embedded into a classical combinatorial model, by defining a so called neuron constraint for each of the neuron in the trained NN. This is done by separately tackling the activity expression and activation function by decomposing Equation 4.1 such that we have:

$$A = b + \sum_i X_i w_i \tag{4.2}$$

$$Y = \phi(A) \tag{4.3}$$

Here, A is an artificially introduced activity variable. Equation 4.2 is linear and poses no embedding issues. For my experiments, I used four commonly found activation functions. This allows ϕ in Equation 4.3 to take four possible forms, namely:

- Linear: $\phi(A) = A$
- ReLu: $\phi(A) = \max(0, A)$
- Sigmoid: $\phi(A) = \dfrac{1}{1 + \exp(-A)}$

- tanh: $\phi(A) = \tanh(A)$

These activation functions can be implemented in a straightforward matter by using MiniZinc in combination with the JaCoP solver [52]. An example of such an embedding can be found in Appendix B.

4.3.2 Embedding EMP Into CP

Embedding the NN in the CP model allows definition of constraints over the values outputted by the NN. As stated in Section 2.1.3, there are three main aspects we can find in the concept of a problem, namely (1) an as-is scenario which is considered to be non-optimal, (2) uncertainty about what decision would lead to the preferred situation and (3) a preferred situation we would like to achieve. Embedding the EMP into the CP model requires us to map these three aspects to a format which that be used by our CP model.

(1) is formed by the simulation model, based on the as-is enterprise model. The simulation model used in our experiments captures the behavior of the enterprise as accurately as possible, which is an assumption made in section 4.1. In the CP model, the embedded NN functions as a representation of the simulation model. As such, how the system operates and what the output of the current decision is incorporated by the NN.

(2) is also incorporated by the NN. Training a NN is a method to create a model that is able to predict the output for inputs of which it does not know the output of. In other words, the NN is used to tackle the uncertainty involved in the EMP: it gives us an idea about the uncertainties involved and the correlation between various variables and parameters in the decision model.

(3) is incorporated by the objective function of the CP model. The objective function can be expressed in various ways. Often this distinction is used to distinguish various subclasses of CP. In this approach, the objective function is expressed by soft-weighted constraints. The main idea is that this helps in finding an approximation of the set of dominating solutions, also referred to as the pareto front. A pareto front of solutions can have a preference over a single solution in many EM applications, as it provides more insight to the solution space. Furthermore, outcomes of the CP model should not be trusted blindly. Instead, the proposed CP solutions should be interpreted as an outline of what decisions could have what impact. As no enterprise model is completely correct, no simulation model is able to capture a system with full accuracy. This is especially import for this approach, as it has an additional layer of uncertainty involved in terms of computational limits since it deals with training time and generating simulation samples.

A pareto front was approximated as follows. First, a set of objective-variables was defined. Typically, these correspond to output variables of the NN. For each objective, it was stated that it should either be maximised/minimised or that a certain variables should equal some value. Next, a collection of CP model variations were created, where each objective had a randomly assigned weight assigned to it. In addition, each optimisation objective had a randomly assigned threshold assigned to it. Next, each CP model was solved. The pareto optimal solutions among the solved CP models is regarded to form the output of the approach. This set of solutions forms the output of the system, and can be presented to a team or individual that needs to make organizational decisions.

4.3.3 Reducing Solving Time

There are several ways to reduce solving time of the model. One way would be to reduce the complexity of the NN. The trade-off here would be accuracy with respect to the simulation model. Another way would be to try various search strategies. Unfortunately, many search strategies were not implemented in JaCoP. Since JaCoP was the only solver I could find that had functionality implemented to solve sigmoid and tanh activation functions on float variables, experimenting with search strategies is left as future work. A third way we could reduce runtime, and which I used thoroughly in my experiments, is to discretize float-valued input variables by using a rounding procedure. For example, a float variable X_f can be discretized by defining a int variable X_I and state that $X_f = 10^{-k} X_I$ where $k \in \mathbf{N}$ is a parameter set beforehand. Decreasing k results in fewer possible intervals and thus in general should reduce solving time.

4.3.4 Embedding Business Constraints

As explained earlier, CP is often seen as the closest approach to “the user states a problem, the computer solves it”. CP is goal oriented, and specific interests formalized in natural language can often be directly expressed in terms of constraints. For example, instead of assigning a threshold at random, a manual one can be set, indicating that a particular objective is deemed to be more important. Alternatively, it can be stated that some relationship between variables should always be satisfied. For example, machine A should never produce more than machine B, or department C should always have more employees than department D.

A practical example can be derived from the constraint C1 in Figure A.6. It states that “all ordered construction materials must be in accordance with construction regulatory standard RS8000-02”. The effect on the systems outcome that such a regulatory standard should be set can be implemented in a simulation model,

```
% pseudocode for the restaurant case as a CP model

% NN_4layer represents the neural network and maps the correlation between buyingStrategy
  and spoilage, success
include NN_4layer

% parameters of the model
float[] buying_constraints = [c1,...,cn]

% decision variables
var[] buyingStrategy = [b1,...,bn]      % strategy to propose
var spoilage                  % spoilage percentage of resources
var succes                    % succes percentage of customer orders

% buying strategy should be positive and is limited by buying_constraints
constraint forall (i in 1..n) (0 <= bi <= ci)

% solve two weighted soft constraints, with some (randomly) assigned weight and threshold
solve (
  weighted-soft-constraint(spoilage <= spoilage_threshold, spoilage_weight),
  weighted-soft-constraint(succes >= succes_threshold, succes_weight)
)
```

Figure 4.2: Psuedocode for the restaurant CP model. It has two objectives, minimising spoilage and maximising success, and one decision parameter, indicated by `buyingStrategy`. `spoilage_threshold`, `spoilage_weight`, `success_threshold` and `success_weight` were randomly generated.

for example by including a parameter that specifies what standard the materials should be, with a few options to choose from. If we would include such a variable in the process of gathering data, the NN is trained on it as well, meaning we are able to use it in our CP program. We can then define a constraint in the CP model that the regularity standard should equal RS8000-02.

4.3.5 Restaurant CP Model

An example of the restaurant CP model in pseudocode can be found in figure 4.2. Here, `NN_layer` should have a structure similar to the embedded NN found in Appendix B. Additional constraints were put into practice by considering upper bounds for each resource regarding the buying strategy. Two soft constraints, one for each objective, were considered. By solving the CP model under various thresholds and weights for both objectives, it should be possible to obtain solutions that reflect different strategies.

4.4 Discussion

The presented approach has various advantages:

- The output is corresponds to a set of strategic decisions that are expected to not dominate each other, hence bring insight to what decisions might have what impact.
- The NN can be used for various EMP, which allows us alter and solve the CP model without executing simulations or retraining the NN.
- Embedding NN in CP is general and flexible and many existing deep learning techniques can be applied.
- It allows the use of simulation models to model the complex and emergent behaviour found in enterprises, which is a well studied approach to do so.
- Many business constraints can be expressed in terms of CP constraints in a rather intuitive way.

A limitation of this approach is the need of simulation data to train the NN, as generating many simulation samples can be computationally exhaustive. This problem can be tackled by using a more suitable simulation-optimisation technique, such that optimisation and simulation can be integrated more efficiently. This is discussed in more detail in Section 6.2.1.

Chapter 5

Experiments

The main purpose of this chapter is to verify the findings in Chapter 3 and 4. This is done through four experiments, each based on some problem that needs to be solved. Each problem was tackled using the approach described in Chapter 4. The findings of the experiments are summarised in Section 4.4.

5.1 Hotel Slaaplust

This experiment is based on a DEMO model presented by Dietz [12]. Here, Dietz explains the how a DEMO model can be derived from a descriptive story, more particular, one that explains Hotel 'Slaaplust'.

The main goal of this experiment is to show that every part of the suggested approach from the previous chapter can be realised and integrated next to each other to construct a full pipeline, with as input some enterprise model and an EMP, and as output more understanding on the best strategical decision. Therefore, we are interested in using an input that corresponds to how enterprise modelling is commonly approached. This requirement can considered to be satisfied with respect to the enterprise model. The model was originally created to provide an example of how one should use DEMO, and can therefore be considered a typical example of how one should create a DEMO model. Since no EMP was defined in the document, an EMP had to be designed for this experiment.

5.1.1 Problem Definition

The problem we aim to solve can be described as follows. We consider a hotel under the name of Slaaplust, which is described in detail according to Appendix C. The hotel has infinite rooms, and its capacity – the number of guests it can accommodate – solely depends on what the receptionists can handle. Receptionists can be busy with some task, and if so, they are not able to answer an incoming reservation or check-in. As a result, potential guests are often put in a queue. If these are put too long in a queue, they will become unsatisfied and post a negative review on the internet. If the queue takes even longer, he or she will leave, resulting in a lost client.

The management would like to optimise the workflow of receptionists, such that less clients are lost and less negative reviews are being posted. It has found out it is possible to affect the decrease duration of some tasks at the cost of others. For example, it can decide to not ask for some of the personal details when a client makes a reservation, but instead ask to do so during check-in. This would decrease the time it takes to make a reservation, at the cost of an increased check-in duration.

The management has access to historical data, which allows them to predict the upcoming season. This consists of the following:

1. Duration of the various transactions (see Section 5.1.2).
2. Number of receptionists that are employed.
3. How evenly spread through the day guests arrive at the hotel for a checkin, or make a reservation (a low spread means that peak hours are very crowded with respect to non-peak hours).
4. How long it takes for a guest to leave when it has to wait too long.
5. Proportion of guests that do not show up.

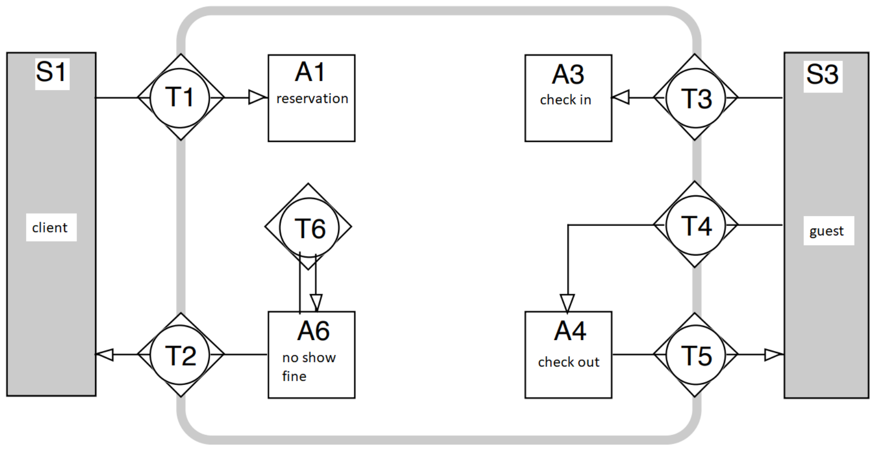


Figure 5.1: Communication diagram of Hotel Slaaplust. The receptionist, S2, is represented by the middle block [12].

5.1.2 Setup

The input consists of a descriptive story written (in Dutch) by Jan Dietz, the founding father of the DEMO framework, and can be found in Appendix C. From this, the various DEMO models, each one based on a different abstraction level, can be constructed. This mapping is discussed in detail by Dietz [12]. I will not go into detail on how these models are constructed, but rather put a focus on the resulting diagrams, since these are used in further parts of the pipeline.

Transactions

The resulting model considers 6 transactions that can be made.

- **T1: reservation.** The client calls to the reception to make a reservation.
- **T2: fine-payment.** The client pays a fine to the hotel for not showing up.
- **T3: check-in.** The client arrives at the hotel and communicates with the reception such that it can receive a room.
- **T4: check-out.** The client has finished its stay and communicates with the reception that it aims to leave the hotel.
- **T5: stay-payment.** The client pays for the stay at the hotel.
- **T6: internal-cancellation.** The reception decides to cancel a reservation.

The Communication Diagram

The communication diagram shows what actors are involved and how this relates to the various transactions. It is shown in Figure 5.1. A distinction can be made between internal and external actors, based on whether they are part of the enterprise and therefore some sense control of their behaviour. The hotel considers one type of internal actor, namely the receptionist, and two types of external actors, namely a client and a guest. A client is someone who makes a reservation. As soon as a client shows up at the hotel, it becomes a guest. The actor that is at the start of the arrow is called the initiator of the transaction. As such, T6 is a self initiating transaction, where the receptionist does not have to communicate with a client or guest.

The Process Diagram

The transaction flow is captured by the process diagram, as shown in Figure 5.2. This shows what transaction will cause other transactions to happen. This is essential information for creating the simulation model, as it models the flow of events.

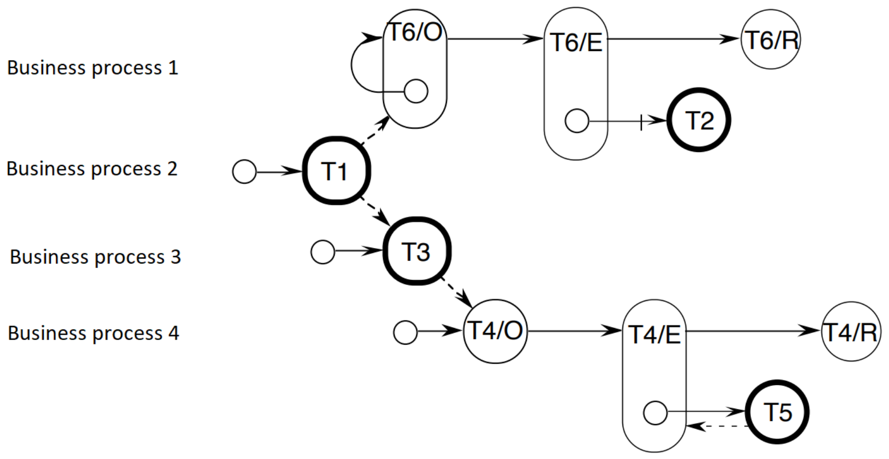


Figure 5.2: Process diagram of Hotel Slaaplust [12].

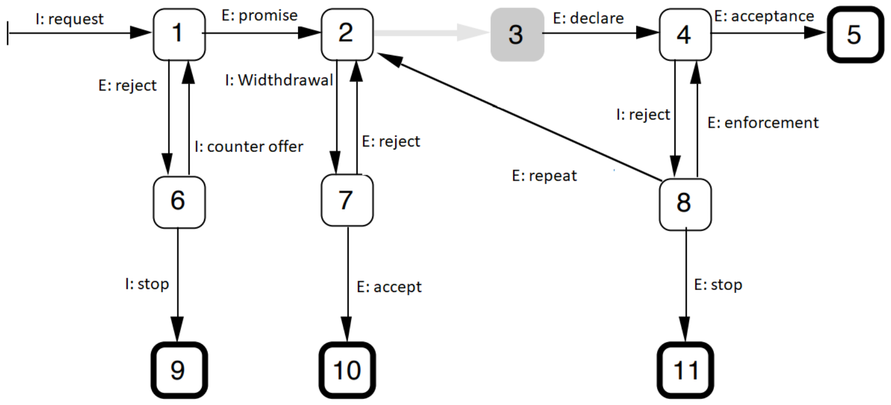


Figure 5.3: Transaction diagram of T1: making a reservation [12].

The Transaction Diagram

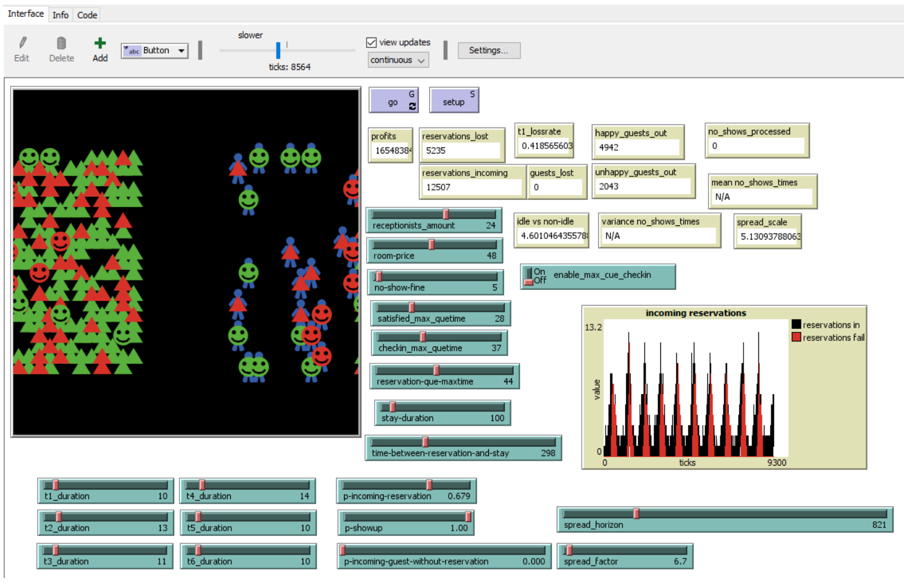
A more detailed model of transactions is given by a transaction diagram. This models the various outcomes of a single transaction and what causes them to happen. The transaction diagram of T1 is shown in Figure 5.3. Although this is useful information to construct a detailed simulation model, transaction diagrams were not consulted for my experiment. One reason is that only the transaction diagram of T1 was provided by Dietz. The other reason being that the hotel case is fictious, so the simulation model does not necessarily need to reflect realistic behaviour. If someone aims to create a more detailed simulation model, I would advise to use transaction diagrams.

The Simulation Model

Based on this demo model, a simulation model was created using NetLogo [53], as shown in Figure 5.4. The simulation model allows us to set input parameters, run it for various ticks, and observe several metrics. Through the use of Python scripts random inputs were send to the simulation model and its outputs were collected as a csv file. A NN with an architecture according to the one explained in Section 4.2 was trained on this.

5.1.3 Results

Modelling the relationship between task durations – that decreasing duration of one task increases the duration of some other task – was done so in the CP model. This was done according to Equation 5.1, where a and b are transactions, c_i how much the duration of transaction i was changed (i.e. $c_i = t_i^* - t_i$ where t_i^*, t_i are respectively the new and old duration times), $0 \leq w \leq 1$ a weight indicating the impact of the change (a high w indicates that change to the duration of a has a very large impact on the duration of b). This equation was put three times in the model according to values found in Table 5.1.



ticks: 8564
view updates ✓ continuous

profits 16548384
reservations_lost 5235
t1_lossrate 0.418565603
happy_guests_out 4942
no_shows_processed 0
reservations_incoming 12507
guests_lost 0
unhappy_guests_out 2043
mean no_shows_times N/A
receptionists_amount 24
room-price 48
no-show-fine 5
idle vs non-idle 4.601046435578
variance no_shows_times N/A
spread_scale 5.13093788063
On [✓ Off] enable_max_cue_checkin
satisfied_max_quetime 28
checkin_max_quetime 37
reservation-que-maxtime 44
stay-duration 100
time-between-reservation-and-stay 298
■ reservations in
t1_duration 10
t2_duration 13
t3_duration 11
t4_duration 14
t5_duration 10
t6_duration 10
p-incoming-reservation 0.679
p-showup 1.00
p-incoming-guest-without-reservation 0.000
spread_horizon 821
spread_factor 6.7

Figure 5.4: Netlogo model of Hotel Slaaplust. The model allows various input parameters to be set as well as observing various metrics. Here, triangles represent clients, people-figures represent receptionists and smileys represent guests. Colors indicate what stage the client or guests is currently in.

a	b	w	max relative change
t1	t3	0.5	0.1
t1	t6	0.5	0.1
t3	t4	0.5	0.1

Table 5.1: Values used for the hotel experiment to model the impact of duration changes.

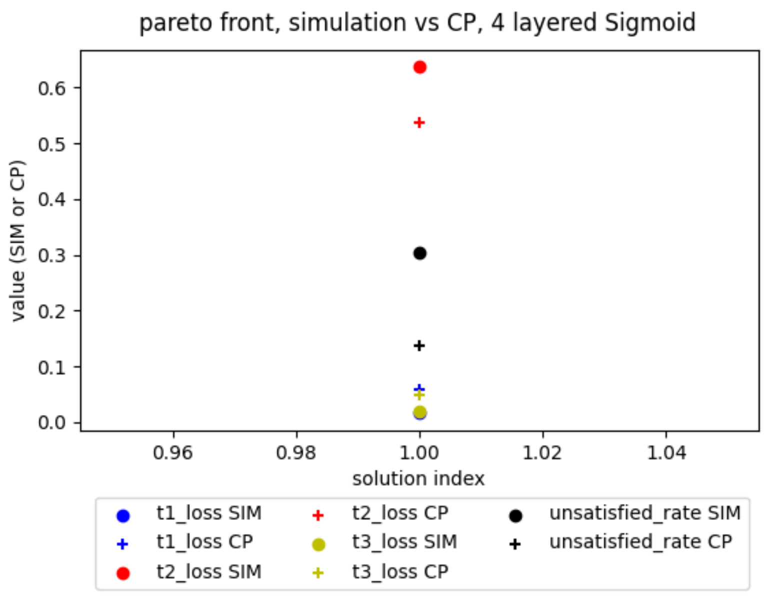
$$c_b = \begin{cases} -c_a * w, & \text{if } c_a \geq 0 \\ -c_a * w^{-1}, & \text{otherwise} \end{cases} \tag{5.1}$$

Results can be found in Figure 5.1.3. The outcome of the simulation is that it is beneficial to decrease some of the durations at the cost of others. More particularly, decrease t4 by increasing t3, decrease t3 by increasing t1, and decrease t6 by increasing t1. There is room for improvement in terms of accuracy. The CP model still has a large bias, in the sense that it predicts t1_loss and t3_loss too high and unsatisfied_rate and t2_loss too low. Furthermore, we can observe only one dominating solution presented. This makes sense if we consider the structure of the problem. All objectives are based on decreasing the workload of receptionists, as that is the major cause of worse objective values. As such, the solution that scores best on decreasing que times, is very likely to be the solution that dominates all others.

5.1.4 Conclusion

The experiment functions as a good example of how the complete pipeline can be realised. It shows step by step how one could get insight in how some decisions affect the systems functioning from a descriptive story accompanied by historical data.

There are however a couple of limitations that make this experiment an incomplete proof of concept. First, the accuracy of the CP solutions were not very accurate with respect to the simulation. This can be overcome by implementing some of the improvements discussed in Section 6.2 or by reducing the input complexity of the model, such that less training samples are necessary. Second, the experiment did not show multi-objective behaviour, in the sense that it tends to present a single dominating solution. This can be explained best as a result of the problem definition: the most efficient workflow that increases the availability of receptionists should in general decrease all queue times, and as a result, has a high chance of being the overall dominating solution. Third, the hotel model is fictious and simplified. Many details that are often present in models based on real-world scenarios were left out. A major interest of designing simulation models based on enterprise models is how to capture complex emergent properties, and a more realistic and detailed simulation model might have more of them.



```
solution 1:
    't1_duration': 20.0,
    't2_duration': 10,
    't3_duration': 12.5,
    't4_duration': 7.5,
    't5_duration': 10,
    't6_duration': 7.5

expectation 1:
    't1_loss': 0.0580633796888708,
    't2_loss': 0.5366184784804018,
    't3_loss': 0.0473613914878937,
    'unsatisfied_rate': 0.1367359165826422
```

Figure 5.5: Mean stock values and loss rates for various solutions of the hotel case. The x-axis corresponds to the solution index. The plus-signs represent values found by the CP model, i.e. the approximated value. The dots represent the actual values, that is, if we would send the proposed solution back to the simulation and observe its outcome. Accuracy of the CP model can be measured by means of the difference between the CP and simulation values. Below the plot, output values of the CP model are shown. `Solution 1` refers to the proposed strategy and `expectation 1` to what the CP model expects the output to be under that strategy.

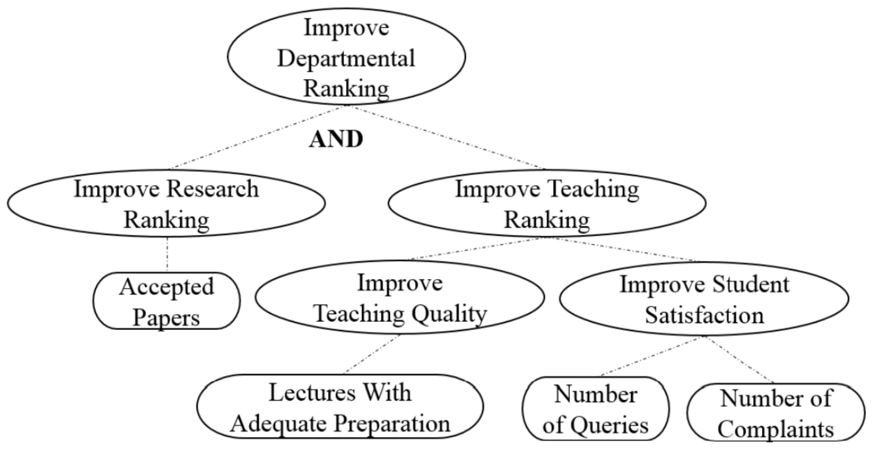


Figure 5.6: Goal decomposition of ABC university [54].

5.2 ABC University

This experiment is based on ABC university: a fictious university that has as goal to improve its ranking. The enterprise model, modelled in OrgML, as well as the corresponding simulation model, designed in the Enterprise Simulation Language framework, is described in detail by Barat [54]. The goal of this experiment is to show that the suggested approach explained in Chapter 4 is applicable to how EM is currently approached.

The preferred setting of this experiment was to reuse the simulation model. Unfortunately, due to organizational property reasons, I was unable to run the simulation model locally. Instead, the author helped me with understanding the underlying enterprise model and how he consulted the simulation model to solve a problem. This changed the scope of the experiment with what I had originally in mind. Instead of showing that my approach is applicable to an experiment by integrating it into the pipeline, I will show that if I would have access to a simulation model the problem could be solved using my approach. This experiment therefore lacks experimental data, but does provide a CP model that can in theory be used to tackle the problem stated by the author (see Appendix D). As a form of verification, I had contact with the author.

5.2.1 Problem Description

ABC University has as its main goal to improve its university ranking. This goal is decomposed into two main sub-goals, namely improving teaching ranking and research ranking. These are in turn accompanied by various metrics: the research ranking is based on the number of accepted papers. Teaching ranking on the number of lectures given with adequate preparation, number of queries and complaints raised. This decomposition of goals is shown in Figure 5.6. We consider the goal to be a multi-objective one, in the sense that the main goal - improve departmental ranking - cannot be expressed as a weighted combination of the subgoals. This means that the modelling goal is to obtain insight on how organizational decisions have impact on the various subgoals.

For this experiment, we consider access to a simulation model. This simulation model simulates micro events such that a macro understanding of the system can be observed. This is done in a state-machine like fashion, as shown in Figure 5.7. The simulation model considers five variation points. These form the input of our simulation model. These are:

- Number of full time academics.
- Number of part time academics.
- Distribution of academics (ratio of TeachingAcademics, ResearchAcademics and TeachingAndResearchAcademics).
- Number of students.
- Work priority of the academics (i.e. a workPriority per type of academic).

The number of students, full time academics and part time academics to be given a priori. The distribution of academics and the work priority of academics are considered to be our decision variables. The goal thus becomes to find the optimal distribution of academics and work priorities of academics to maximise university ranking.

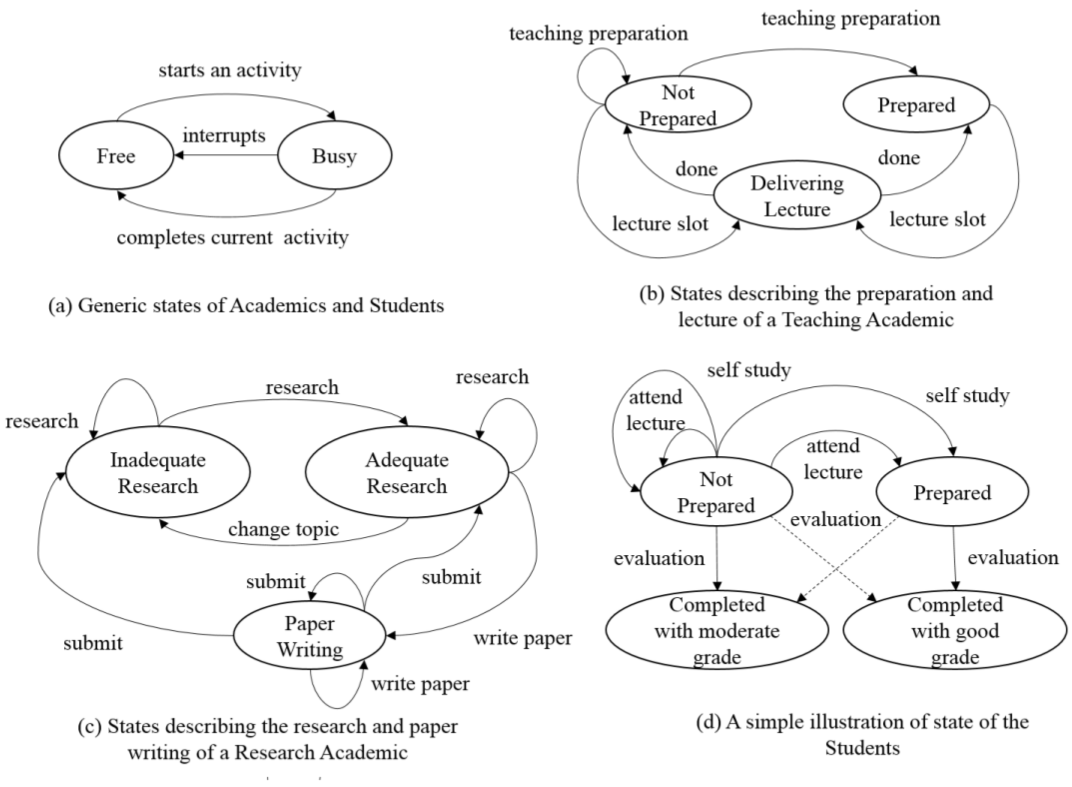


Figure 5.7: Behaviors of active elements of ABC University [54].

5.2.2 The CP Model

Although getting access to a simulation model was not possible, it is possible to design a CP model that aims to solve this. Similar to the other experiments, a pareto front is approximated by using soft constraints. This was done by using MiniZinc [50] and MiniBrass [51]. Its implementation is shown in appendix D.

5.2.3 Comparison With Original Work

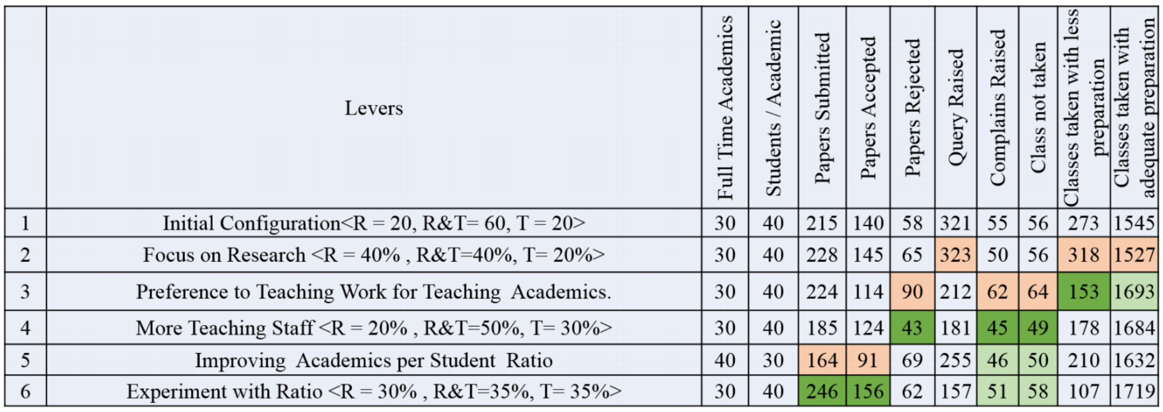
There are differences between the simulation approach presented in the original work by Barat [54] and my approach. The original approach is based on simulation, where various scenarios are fed into a simulation model. The various outcomes are evaluated and based on this, it is decided what levers, i.e. strategical decisions, should be made such that the university is able to fulfill its goal.

The simulation approach is based on exploring various scenarios and observing the corresponding simulation results. This is shown in Figure 5.8. Observations can be for example that adding more research academics may not lead to better research outcomes, that change in work priority can help improve teaching quality at the cost of student satisfaction, etc. According to the author, the most desirable outcome is that of row 6, and the next step would be to use that as a basis and explore further options based on this. My approach would skip this intermediary evaluation step. Instead, it proposes directly an approximation of a pareto front of solutions, based on the various objectives. No further simulation runs are required to explore the solution space any further, which can be a major advantage as it can help speed up the decision making process.

5.3 Supply Chain

Supply chain models are commonly found throughout EM. It explains the case of various manufacturing units that interact with each other by passing products and materials to each other, with some end product as a result. This experiment has as a goal to analyse the efficiency and accuracy of the proposed method in Chapter 5 on a simulation model that has a close correspondence to the functioning of an enterprise. A freely available NetLogo model, based on a simple supply chain was consulted for this experiment [55].

The supply chain model matches that of the Beer Game, introduced by Jay Forrester at MIT in the 60's [56]. The game represents a supply chain with a non-coordinated process. Each individual element of has control over its own part of the supply chain. Problems arise due to lack of information sharing, causing a bull-whip effect – an increase of variance of orders placed by each stage when we move from downstream stage to upstream stages



	Levers	Full Time Academics	Students / Academic	Papers Submitted	Papers Accepted	Papers Rejected	Query Raised	Complains Raised	Class not taken	Classes taken with less preparation	Classes taken with adequate preparation
1	Initial Configuration<R = 20, R&T= 60, T = 20>	30	40	215	140	58	321	55	56	273	1545
2	Focus on Research <R = 40% , R&T=40%, T= 20%>	30	40	228	145	65	323	50	56	318	1527
3	Preference to Teaching Work for Teaching Academics.	30	40	224	114	90	212	62	64	153	1693
4	More Teaching Staff <R = 20% , R&T=50%, T= 30%>	30	40	185	124	43	181	45	49	178	1684
5	Improving Academics per Student Ratio	40	30	164	91	69	255	46	50	210	1632
6	Experiment with Ratio <R = 30% , R&T=35%, T= 35%>	30	40	246	156	62	157	51	58	107	1719

Figure 5.8: Consolidated simulation results [54].

– resulting in inefficient inventory management [57]. The bullwhip effect can be tackled by regulation, such as procurement: finding and agreeing to terms on a between various parties involved in a supply chain.

5.3.1 Problem Definition

We consider the supply chain of a single product. The model considers factories, retailers, distributors and clients. Factories produce products, these are then bought by the retailers, which in turn are bought by the distributors and finally these are bought by the clients. We are able to set an inventory policy, which describes how the different units manage their inventories, and a costumers strategy, which explains how the customers buy their products. These policies reflect various forms of regulation that can be applied. More details regarding the simulation model, such as an explanation of the different strategies, can be found in Appendix E.1.

The management of the supply chain found out there are two issues regarding the current state. First, too many sales are being lost. Secondly, too many products are stored in the factories instead of the distributors and retailers. The number of clients and their demand is given. We are interested in finding the optimal number of factories, retailers and distributors, as well as the optimal inventory policy and costumers strategy, such that little sales get lost and the factories have fewer products stored in them.

5.3.2 Results

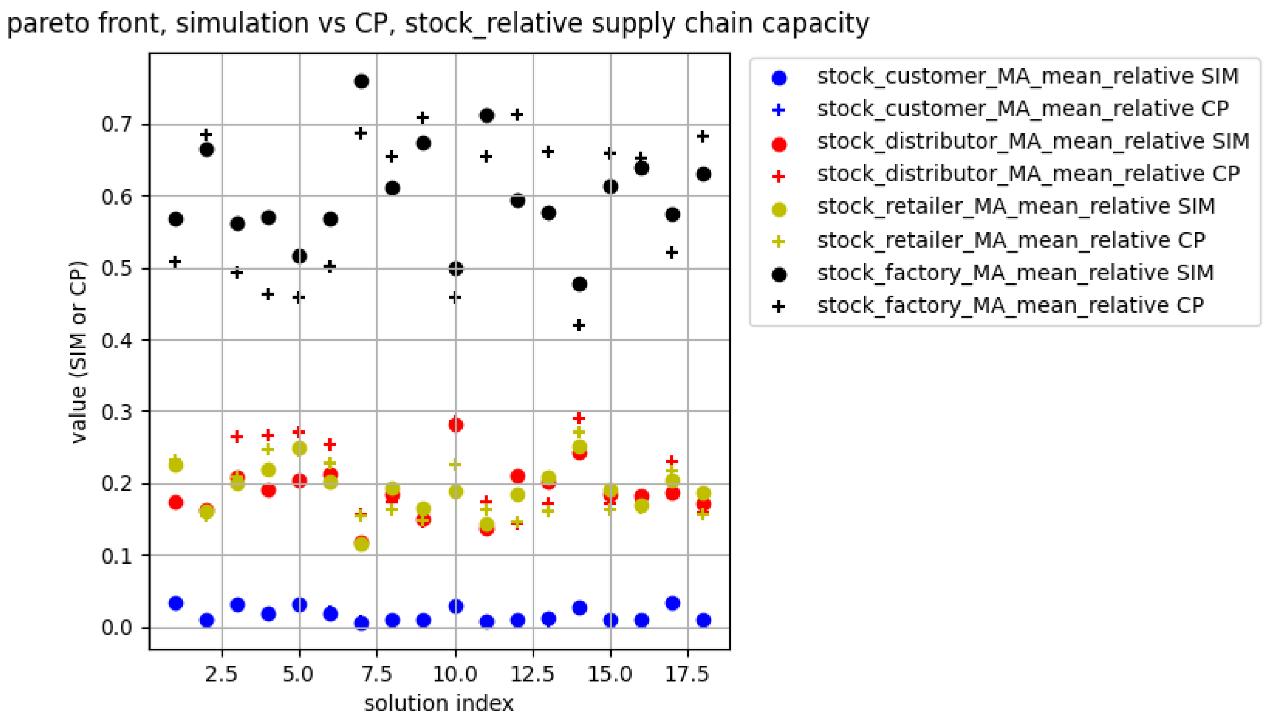
Figure 5.9 shows a plot of the various solutions found. The results confirm the approach is able to find a set of solutions that differ in terms of objective values. Considering there is room for improvement, it can be argued the results shown are decent enough for the sake of this experiment, as the predictions of the CP model are accurate enough such that the actual values - the simulation results - form an approximation of the pareto front. Each solutions proposes, both for the customers buying strategy and the inventory policy, a random policy. Also, the results indicate that it is best to have only a few factories. The number of retailers and distributors we need is dependent on how much we care about the loss of sales.

5.4 Food Wars

This experiment was based on the Fruit Wars [58] model, which is part of NetLogo's standard library of examples. The model is multi-agent based, and demonstrates how non-zero-sum economic environments can encourage cooperation and discourage violence. Its agents are fruit collectors. They wander the map looking for fruit bushes, can attack each other to steal their fruit and, if they survive, pass on their characteristics to offspring. These characteristics influence their behaviour. By adjusting system parameters, such as the bonus gained through collaborating or the maximum age foragers can achieve, it is possible to have impact on the traits of the population. More details on the model can be found in Appendix E.2. The goal of this experiment is to evaluate the accuracy of the proposed approach on a multi-agent based simulation model, where parameters of the model influence the behaviour of its agents.

5.4.1 Problem Definition

The problem can be described as follows. Consider to have a governing position of some area consisting of fruit foragers. You want the fruit foragers to behave more peacefully, but are also interested in keeping a high overall population. That is, minimise strength, speed, murder rate and proactive aggression while maximising all other



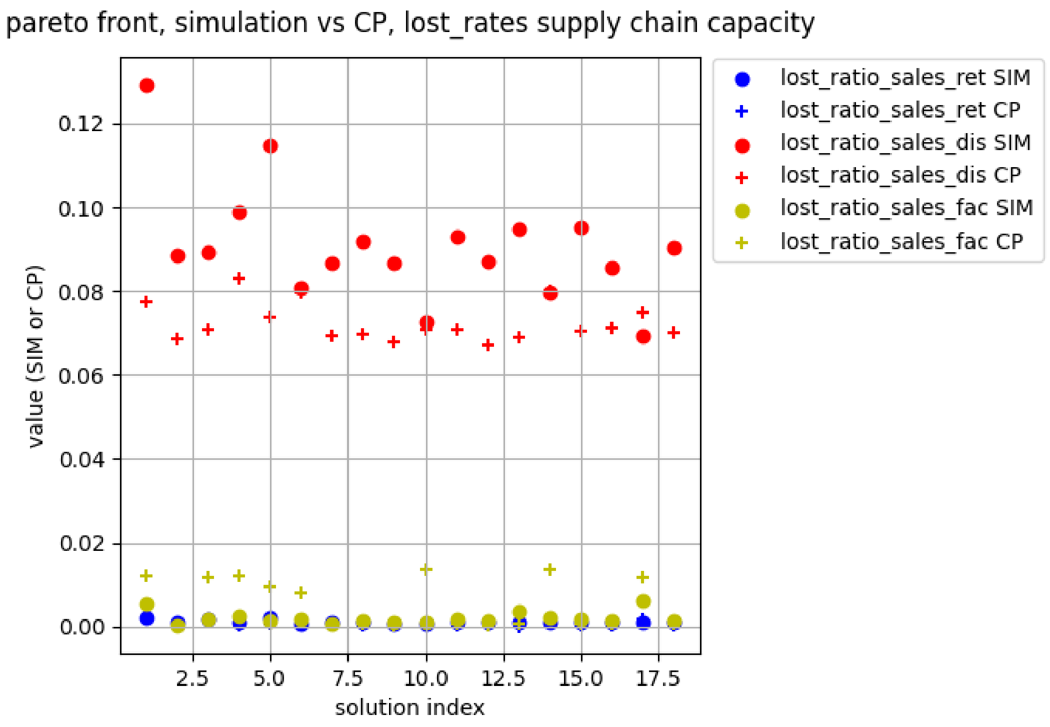


Figure 5.9: Mean stock values and loss rates for various solutions regarding the Supply Chain Experiment. The X axis corresponds to the solution index. The plus-signs represent values found by the CP model, i.e. the approximations. The dots represent the actual values, that is, if we would send the proposed solution back to the simulation and observe its outcome. Accuracy of the CP model can be measured by means of the difference between the CP and simulation values. Of the 21 generated thresholds and weights, 19 were non-dominating. Traning was done for 2 hours on 14288 data samples.

metrics. You can take action in two ways. First, by choosing how much healthcare you provide, reflected by the `max_age` parameter. Second, you are able to give some additional bonus for collaborating, represented by the `collaboration_bonus` parameter.

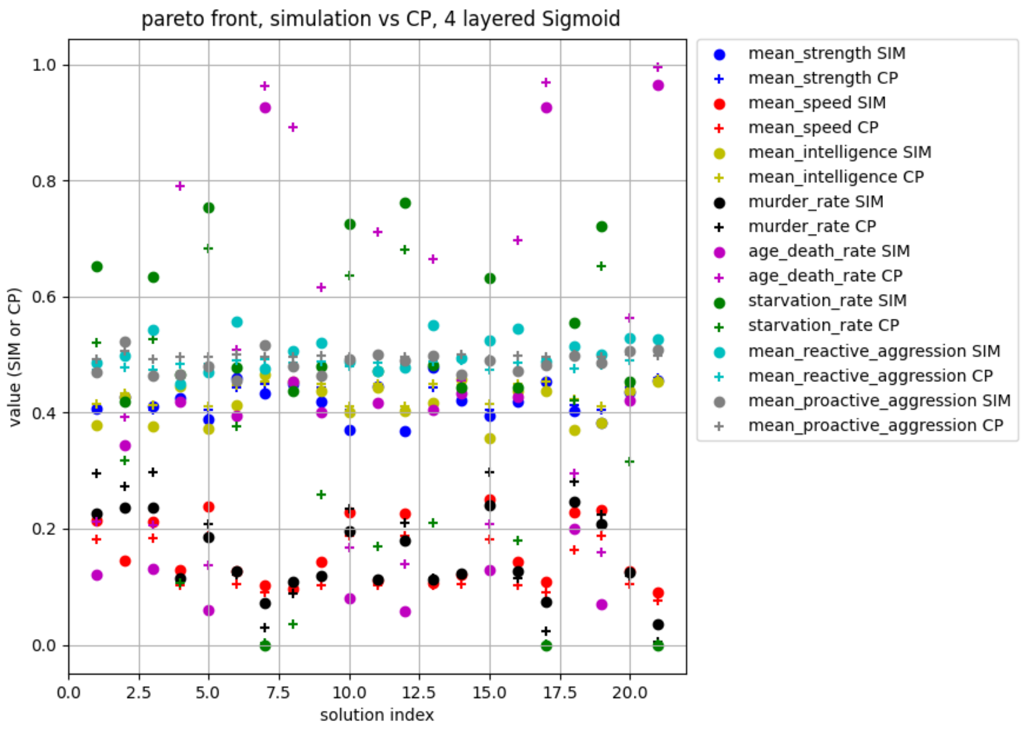
5.4.2 Results

Results are shown in Figure 5.10. Even though the results are not perfect, they do show the CP approach was in many cases able to make a reasonable guess of the simulation model. A second observation we can make is that it is possible to find a pareto front of solutions, which verifies the approach is applicable to a multi-objective case.

5.5 Summary

In this chapter various experiments were shown, using a combination of CP, NN and simulation. The first experiment showed that it is possible to realize this approach for a case where the enterprise model is derived from a narrative story. The second experiment showed that it can be applied as an alternative to a case study where simulation was used to tackle an EMP. The third experiment shows that this approach is able to approximate a pareto front in a setting that is relatable to EM, namely a supply chain. The fourth experiment shows that the approach can be applied effectively for agent-based simulation models.

The experiments have shown that the setup can be realized, is flexible and can be applied to many problems. CP allows flexible and intuitive embedding of EMP's and business constraints. The approach allows the tackling of various problems, based on the same enterprise model with the same simulation data. There is still room for improvement in terms of accuracy, especially regarding the first experiment. To overcome this issue, there are various techniques that can be applied, which is discussed in more detail in Section 6.2



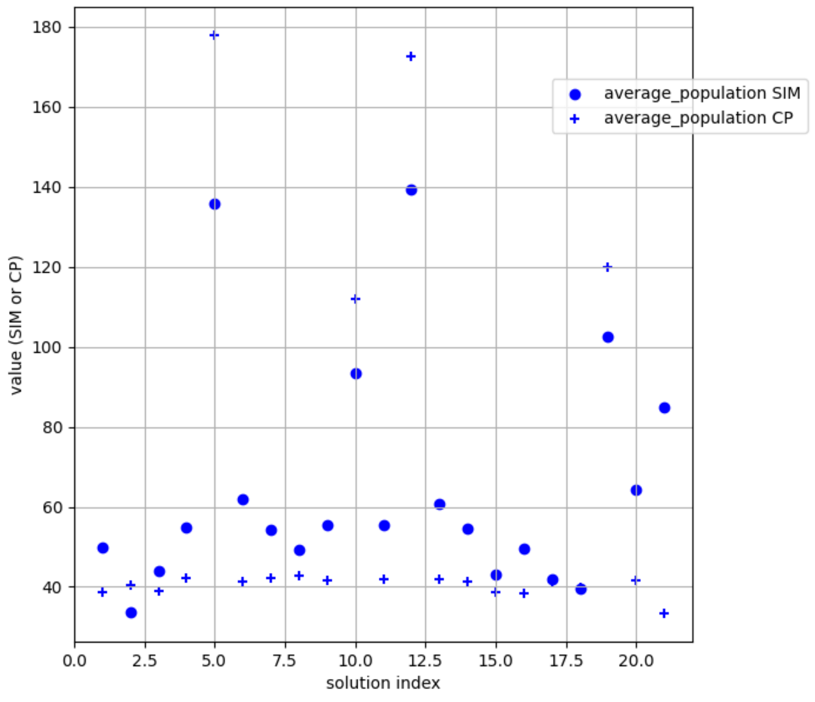


Figure 5.10: Values for the various objectives for the food wars experiment. The X axis corresponds to the solution index. Plus-signs represent values found by the CP model, i.e. the approximated value. Dots represent the actual values, that is, if we would send the proposed solution back to the simulation and observe its outcome. Accuracy of the CP model can be measured by means of the difference between the CP and simulation values. 35500 data points were used to train the NN, over a course of 4 hours on a GTX1070 GPU. 50 random assignments of thresholds and weights were generated and solved. Among the found solutions, 21 of them were non-dominating.

Chapter 6

Conclusion

EM and CP have many studies dedicated to them. Even though there are similarities to be found, as far as my knowledge goes, no previous study explicitly combined the two. The goal of this thesis is to give a good understanding of the similarities and differences between the two sciences and explain how they can be combined. This was done by providing literature, general thoughts, a possible approach supported with experiments and a collection of ideas for future work. Code used for this thesis can be found on GitHub [1].

6.1 Research Questions

Chapter 1 introduced the problem and formulated three research questions, namely:

1. How do we design a mapping that formulates EM aspects as a CSP?

2. How do we decide whether an enterprise model is suitable for a conversion to CSP?

3. What are the advantages of formulating EM as a CSP?

The first question is answered mainly by Chapter 3 and 4. Chapter 3 discusses how such a mapping should look like. Chapter 4 explains a possible approach, namely by training a neural network on a simulation model, embedding that into a constraint program and use soft-constraints to approximate a pareto set of solutions.

The second question is answered by Chapter 3. The main answer here is that it is of high importance to find a suitable abstraction level. On one hand, properties of the enterprise model may not get lost. On the other hand, the model should not get too complex, else it becomes infeasible to solve the optimisation model. This is expressed by the trade-off of tractability vs expressiveness. An example of a optimisation model that can considered to be too complex to solve by means of CP is that of petri net representations. Furthermore, data analytical tools can be helpful in finding a suitable abstraction level.

The third question is answered more generally by Chapter 3 and more specifically for the proposed method by Chapter 4. The main benefit is that CP is a declarative programming technique, and therefore has a close relationship to normal language. As a result, both business constraints found in enterprise models and business goals found in EMP's can be modelled effectively and intuitively in a constraint program. Furthermore, CP is well studied and many extensions on the classical CSP that deal with a large variety of modelling problems exist. For example, the current state of CP is able to deal with historical data, soft constraints, feedback loops and multiple objectives.

Chapter 2 functions as background, and provides the necessary information and sources to understand the rest of this thesis. Chapter 5 functions as verification for the approach described by Chapter 4, by showing that it can be put into practice.

6.2 Future Work

I expect many improvements can be made over the presented approach. This section should give an overview of what is, according to me, worthwhile to study next.Future work is divided into four categories, based on what it is expected to improve.

[1] https://github.com/SytzeAndr/EM_to_CP

6.2.1 Improved Learning

By optimising the learning procedure of the simulation model, the number of simulation calls necessary can be reduced. This should result in more accurate predictions as less simulation data is necessary in order to create an accurate model. Improving learning can be done in various ways.

Simulation-Optimisation

The overall concept of simulation–optimisation is to combine the benefits of both to solve a problem. It is considered to be a class of optimisation, in which the simulation represents the actual system. If a set of inputs is given to the model, the outputs should be minimised and matched with the actual system as much as possible, without loss of generality [59].

Simulation-optimisation can be categorised into three approaches: Solution Evaluation (SE), Solution Generation (SG) and Analytical Model Enhancement (AME). More details on these categories, including applications and subcategories, can be found in work by Figueira and Almada-Lobo [47].

- SE refers to first developing a comprehensive simulation model to represent the system, and use that model to evaluate solutions. The purpose of simulation here is to evaluate the performance of solutions, whereas the purpose of the optimisation is to decide what parameters should be picked. It generally consists of an iterative approach, where a feedback loop is used.

- SG is used to improve simulation variables, not to evaluate the solutions. The optimisation is executed as a mechanism of the simulation to determine some variables of the complete configuration. [59]. A major difference with respect to SE is that here simulation instead of optimisation is used to generate solutions.

- AME uses simulation results to enhance the analytical model. Feedback information is utilised to improve the construction of the solution evaluation approach.

The approach of this thesis falls best under the category of a Function Estimation based Approach (FEA), which is considered to be a sub-category of AME. FEA estimates functions that describe the relationship between particular input and output variables. In our case, the NN estimates the simulation model. FEA can also be used to model subsystems of the model. For example, a FEA approach to specify the relationship between the expected work-in-progress and the expected throughput is presented by Asmundsson et al. [60].

It can be interesting to compare various simulation-optimisation techniques when put in the context of combining CP and EM. Instead of considering simulation and optimisation as a two step process, they can be integrated, by predicting what parameters should be sent to the simulation model. For example, a promising technique for the scope of this work that can be worth investigating is that of constraint acquisition, where the learner (the cp model) is allowed to ask queries (send inputs to the simulation) [61].

Neural Network Design

The focus of this thesis is not on machine learning, and how the architecture depends on the simulation data can be considered future work. Therefore, I aimed to use an architecture that can be considered both simple and general. As such, I expect better performance can be achieved if one would experiment with different architectures. There are many design choices possible regarding the architecture of a NN, and designing the best architecture for the problem is one of the major concerns of deep learning specialists. Fundamental design choices are the amount of hidden layers, the size of each hidden layer, and what activation function is used at which place. Architectures where multiple NN interact or are part of each other, i.e. nested NN, are commonly used in practice.

A review on how to fix the number of hidden neurons is discussed by Sheela and Deepa [62]. How to choose the amount of hidden layers and hidden units is discussed in part 3, section 9 and 10 by Sarle [63]. The entire pipeline of a machine learning model can also be automatically generated. This is referred to as automated machine learning, of which a survey is presented by He et al. [64].

6.2.2 Improved Modelling

A good CP model accurately reflects the real world. Several improvements can be made on the presented approach that deal with modelling concepts, such that information obtained at a previous stage is used to create more accurate CP models.

Modelling Uncertainty

This thesis deals with many forms of uncertainty. Enterprises are dynamic and uncertain, which is represented by the stochastic behaviour of simulation models. The presented approach in Chapter 4 deals with another layer of uncertainty, namely that of training a NN such that it fits the simulation model. Outputs of the presented approach in Chapter 4 should not be expected to be fully accurate. It would therefore make sense to reflect this by including distributional properties of the various variables and parameters involved in the CP model. Distributional properties can be estimated from simulation or real world data, but also from training errors of the Neural Network. Such a setting allows to control the trade-off between optimising the objective function and ensuring the satisfiability of the solution under random parameters. An example of such a technique are confidence constraints [46]. Confidence constraints ensure with some probability that a set of variables are no smaller than random variables for which the probability distribution is given. This allows the solution to be robust to external random events. More about solving under uncertainty can be found in a survey by Verfaillie and Jussien [65].

Combining EWO and Simulation

EWO, as discussed in Section 3.2.2, studies how to optimise such a supply chain by using optimisation methods. Many EWO cases use no simulation model; parameters for the optimisation model are derived from historical data. However, simulation models can be useful to predict the behaviour of some enterprise elements. For example, the throughput of a machine might be dependent on weather circumstances, which are difficult to compute but can be modelled effectively by decomposing its functioning into micro events. A way to integrate simulation models into supply chain can be to embed a machine learning model - trained on some simulation model - into the optimisation model. The experiment from Section 5.3 experiment does not do such a decomposition, since the entire supply chain is modelled by means of a NN. It might be an interesting to study how a EWO and simulation models can be combined effectively.

6.2.3 Improved Solving

Various solving techniques from CP literature can be consulted in order to improve the solving procedure. Such techniques allow faster convergence to solutions and can reduce runtime significantly. How and which techniques can be effectively applied to the work in this thesis is left as future work.

Advanced Search Strategies

Due to limitations of JaCoP [52], as discussed in Section 4.3.3, advanced search strategies were not considered. Experimenting with different search strategies might provide insight in how solving time can be reduced on CP models that have NN embedded in them. Search strategies can also help solving in a multi-objective manner, which is for example done by Bennetto and van Vuuren [4] through evolutionary search. A review on constrained multi-objective optimisation algorithms is presented by Afshari et al. [66].

Semiring Soft Constraints

An alternative of using weighted soft-constraints would be to use semiring soft constraints. Instead of expressing soft constraint hierarchies by the use of weights, semiring soft constraints use a tree like structure. Instead of creating CP model variations by generating random weights (as done in our experiments), we can shuffle this tree randomly. It can very well be that using semiring CSP leads to faster convergence of the pareto front as it is more closely related to the principle of dominating solutions. However, this should be tested experimentally, and was out of the scope for this thesis.

6.2.4 Improved Diversity

Other Frameworks And Simulation Environments

It can be interesting to study how other frameworks would perform. The discussed frameworks in this thesis – DEMO, ArchiMate, MEMO and 4EM – are considered to be descriptive and can therefore be more difficult to simulate than other more IT focused frameworks, such as i* or OperA. Next, there are also frameworks with a high focus on problem statement, which can be interesting to study. A comparison of EM frameworks on their structure that assesses which ones have a natural conversion to a CP model, can be an interesting next topic of study.

A similar comparison regarding simulation environments can also be an interesting topic of study. Throughout this thesis, NetLogo was used. Although NetLogo is easy to setup, it is difficult to create large and complex simulation models due its limitations regarding inheritance of objects. Simulation frameworks that put more focus on inheritance could scale better towards the size of the enterprise model. An example of a promising actors- and message-based simulation environment that has good inheritance functionality is Akka [67].

6.3 Publication

As mentioned in Chapter 1, as far as my knowledge goes, the explicit combination of CP and EM has not been studied before. This thesis shows this combination can be realised and has several advantages. Furthermore, Section 6.2 shows that there are several interests with respect to future work. Therefore, I consider the work of this thesis to be a good contribution to EM and will work towards a publicated version of this thesis.

This publicated version should explain the main findings of this thesis. It will become more compact, by referring more to existing sources regarding background. Furthermore, I would like the published version to include several improvements mentioned in future work. I am in particular interested in the use of Simulation-Optimisation techniques that have a feedback loop with the simulation model (see Section 6.2.1), such that learning can be performed more effectively. This slightly changes the approach described in Chapter 4. Furthermore, I am interested in experimenting on a different case, for example by using a real-world application or different framework (see Section 6.2.4).

This thesis crosses a variety of sciences, and as such there are various options regarding where to publish my findings. Relevant are journals and conferences that study how engineering techniques can be applied to EM, for example:

- Enterprise & Organizational Modeling and Simulation (EOMAS) [2]
- Expert Systems with applications [3]
- Enterprise Engineering Working Conference (EEWC) [4]
- IFIP Working Conference on The Practice of Enterprise Modeling (PoEM) [5]

Others that can be considered relevant are those that study applications of CP, for example:

- International Joint Conferences on Artificial Intelligence [6]
- Engineering Applications of Artificial Intelligence [7]
- Principles and Practice of Constraint Programming [8]

[2] https://eomas-workshop.org
[3] https://journals.elsevier.com/expert-systems-with-applications
[4] https://link.springer.com/conference/eewc
[5] https://link.springer.com/conference/poem
[6] https://ijcai.org
[7] https://sciencedirect.com/journal/engineering-applications-of-artificial-intelligence/vol/94/suppl/C
[8] https://link.springer.com/book/10.1007/978-3-030-58475-7

Bibliography

[1] François Vernadat. Enterprise modelling: Research review and outlook. *Computers in Industry*, 122:103265, 2020. ISSN 0166-3615. doi: 10.1016/j.compind.2020.103265.

[2] Y. Chen-Burger, A. Tate, and D. Robertson. Enterprise modelling: A declarative approach for fbpml. 2002.

[3] Kurt Sandkuhl, Janis Stirna, Anne Persson, and Matthias Wißotzki. *Enterprise Modeling: Tackling Business Challenges with the 4EM Method*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[4] Robert Bennetto and Jan H van Vuuren. Multi-objective evolutionary search strategies in constraint programming. *Operations Research Perspectives*, 8:100177, 2021. ISSN 2214-7160. doi: 10.1016/j.orp.2020.100177.

[5] Céline Brouard, Simon de Givry, and Thomas Schiex. Pushing data into cp models using graphical model learning and solving. In Helmut Simonis, editor, *Principles and Practice of Constraint Programming*, pages 811–827, Cham, 2020. Springer International Publishing.

[6] Sybren Kinderen and Monika Kaczmarek. On model-based analysis of organizational structures: an assessment of current modeling approaches and application of multi-level modeling in support of design and analysis of organizational structures. *Software and Systems Modeling*, 19:313–343, 11 2019. doi: 10.1007/s10270-019-00767-4.

[7] Bruno Vallespir and Yves Ducq. Enterprise modelling: from early languages to models transformation. *International Journal of Production Research*, pages 1–19, 01 2018. doi: 10.1080/00207543.2017.1418985.

[8] Bart-Jan van Putten, Virginia Dignum, Maarten Sierhuis, and Shawn Wolfe. Opera and brahms: A symphony? 5386:257–271, 05 2008. doi: 10.1007/978-3-642-01338-6_19.

[9] Eric Yu. Social modeling and i*. pages 99–121, 01 2009. doi: 10.1007/978-3-642-02463-4_7.

[10] Roland Ettema and Jan Dietz. Archimate and demo - mates to date? volume 34, pages 172–186, 01 2009. ISBN 978-3-642-01914-2. doi: 10.1007/978-3-642-01915-9_13.

[11] Oana Merkt and Gerd Wagner. Modeling and simulating organisations. *Lecture Notes in Business Information Processing*, 88:45–62, 06 2011. doi: 10.1007/978-3-642-24175-8_4.

[12] Jan Dietz. *Introductie tot DEMO*. Samsom BedrijfsInformatie, 1996. ISBN 90-14-05327-4.

[13] The Open Group. The open group website, 2020. URL opengroup.org.

[14] Ulrich Frank. Multi-perspective enterprise modeling (memo) conceptual framework and modeling languages. *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, pages 1258–1267, 2002.

[15] Ulrich Frank. Memo organisation modelling language (1): Focus on organisational structure. (48), 2011.

[16] Ulrich Frank. Memo organisation modelling language (2): Focus on business processes. (49), 2011.

[17] Ulrich Frank. Multi-perspective enterprise modeling: Foundational concepts, prospects and future research challenges. *Software and Systems Modeling*, 13, 07 2012. doi: 10.1007/s10270-012-0273-9.

[18] Alexander C. Bock, Dmitry Kudryavtsev, and Miroslav Kubelskiy. Towards more expressive problem structuring: A theoretical conceptualization of 'problem' in the context of enterprise modeling. In *2018 IEEE 20th Conference on Business Informatics (CBI)*, volume 01, pages 30–39, 2018. doi: 10.1109/CBI.2018.00013.

[19] Maurice Landry. A note on the concept of 'problem'. *Organization Studies*, 16(2):315–343, 1995. doi: 10.1177/017084069501600206.

[20] David H. Jonassen and Woei Hung. *Problem Solving*, pages 2680–2683. Springer US, Boston, MA, 2012. ISBN 978-1-4419-1428-6. doi: 10.1007/978-1-4419-1428-6_208.

[21] Francesca Rossi. Constraint (logic) programming: A survey on research and applications. In Krzysztof R. Apt, Eric Monfroy, Antonis C. Kakas, and Francesca Rossi, editors, *New Trends in Constraints*, pages 40–74, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. ISBN 978-3-540-44654-5.

[22] M. Wallace. Practical applications of constraint programming. *Constraints*, 1(1-2):139–168, 1996. doi: 10.1007/BF00143881.

[23] Stefan Edelkamp and Stefan Schrödl. *Chapter 13 - Constraint Search*, pages 571–631. Morgan Kaufmann, San Francisco, 2012. ISBN 978-0-12-372512-7. doi: 10.1016/B978-0-12-372512-7.00013-4.

[24] Maria Grazia Buscemi and Ugo Montanari. A survey of constraint-based programming paradigms. *Computer Science Review*, 2(3):137–141, 2008. ISSN 1574-0137. doi: 10.1016/j.cosrev.2008.10.001.

[25] Ronald J. Brachman and Hector J. Levesque. Chapter 16 - the tradeoff between expressiveness and tractability. In *Knowledge Representation and Reasoning*, The Morgan Kaufmann Series in Artificial Intelligence, pages 327–348. Morgan Kaufmann, San Francisco, 2004. ISBN 978-1-55860-932-7. doi: 10.1016/B978-155860932-7/50101-1.

[26] Dominik Bork and Steven Alter. Relaxing modeling criteria to produce genuinely flexible, controllable, and usable enterprise modeling approaches. pages 46–50, May 2018. URL eprints.cs.univie.ac.at/5575.

[27] Combining constraint solving with mining and learning. *Artificial Intelligence*, 244, March 2017.

[28] Michele Lombardi, Michela Milano, and Andrea Bartolini. Empirical decision model learning. *Artificial Intelligence*, 244:343–367, 2017. ISSN 0004-3702. doi: 10.1016/j.artint.2016.01.005. Combining Constraint Solving with Mining and Learning.

[29] Ignacio Grossmann. Enterprise-wide optimization: A new frontier in process systems engineering. *AIChE Journal*, 51(7):1846–1857, 2005. doi: 10.1002/aic.10617.

[30] I.E. Grossmann. Advances in mathematical programming models for enterprise-wide optimization. *Computers and Chemical Engineering*, 47:2–18, 2012. doi: 10.1016/j.compchemeng.2012.06.038.

[31] Ignacio Grossmann. Challenges in the application of mathematical programming in the enterprise-wide optimization of process industries. *Theoretical Foundations of Chemical Engineering*, 48:555–573, 09 2014. doi: 10.1134/S0040579514050182.

[32] P. Sharma, B.R. Sarker, and J.A. Romagnoli. Integrated framework for enterprise management—a synergistic approach towards sustainable biorefineries. 28:1009–1014, 2010. ISSN 1570-7946. doi: 10.1016/S1570-7946(10)28169-5.

[33] Edrisi Muñoz, Elisabet Capón-García, Marta Moreno-Benito, Antonio Espuña, and Luis Puigjaner. Scheduling and control decision-making under an integrated information environment. *Computers and Chemical Engineering*, 35(5):774–786, 2011. ISSN 0098-1354. doi: 10.1016/j.compchemeng.2011.01.025. Selected Papers from ESCAPE-20 (European Symposium of Computer Aided Process Engineering - 20), 6-9 June 2010, Ischia, Italy.

[34] M. Dobrev, D. Gocheva, and I. Batchkova. An ontological approach for planning and scheduling in primary steel production. In *2008 4th International IEEE Conference Intelligent Systems*, volume 1, pages 6–14–6–19, 2008. doi: 10.1109/IS.2008.4670433.

[35] Edrisi Munoz, Elisabet Capon-Garcia, and Jose M. Lainez-Aguirre. Towards advanced enterprise wide optimization based on explicit concept-object oriented mathematical modeling. 40:2347–2352, 2017. ISSN 1570-7946. doi: 10.1016/B978-0-444-63965-3.50393-7.

[36] Oana Merkt and Gerd Wagner. Modeling and simulating organisations. *Lecture Notes in Business Information Processing*, 88:45–62, 06 2011. doi: 10.1007/978-3-642-24175-8_4.

[37] S. Robinson. Conceptual modeling for simulation. pages 377–388, 2013.

[38] Stewart Robinson. Conceptual modelling for simulation part i: Definition and requirements. *Journal of the Operational Research Society*, 59:278–290, 03 2008. doi: 10.1057/palgrave.jors.2602368.

[39] Osman Balci. A methodology for certification of modeling and simulation applications. *ACM Trans. Model. Comput. Simul.*, 11(4):352—377, October 2001. ISSN 1049-3301. doi: 10.1145/508366.508369.

[40] Jim Duggan. A comparison of petri net and system dynamics approaches for modelling dynamic feedback systems. 2006.

[41] Mamadou Seck and Joseph Barjis. An agent based approach for simulating demo enterprise models. *Procedia Computer Science*, 61:246–253, 12 2015. doi: 10.1016/j.procs.2015.09.206.

[42] Suman Roychoudhury, Sagar Sunkle, Hemant Rathod, and Vinay Kulkarni. Toward structured simulation of enterprise models. 09 2014. doi: 10.1109/EDOCW.2014.19.

[43] Zuzana Vejrazkova and Amir Meshkat. Translating demo models into petri net. In Joseph Barjis, Ashish Gupta, and Amir Meshkat, editors, *Enterprise and Organizational Modeling and Simulation*, pages 57–73, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-41638-5.

[44] Tarek Fatyani, Junich Iijima, and Jaehyun Park. Transformation of demo model into coloured petri net: Ontology based simulation. In *KEOD 2014 - Proceedings of the International Conference on Knowledge Engineering and Ontology Development*, pages 388–396. INSTICC Press, 2014. doi: 10.5220/0005137803880396.

[45] Morikazu Nakamura, Takeshi TENGAN, and Takeo YOSHIDA. A petri net approach to generate integer linear programming problems. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E102.A:389–398, 02 2019. doi: 10.1587/transfun.E102.A.389.

[46] Alexandre Mercier-Aubin, Ludwig Dumetz, Jonathan Gaudreault, and Claude-Guy Quimper. The confidence constraint: A step towards stochastic cp solvers. In Helmut Simonis, editor, *Principles and Practice of Constraint Programming*, pages 759–773, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58475-7.

[47] Gonçalo Figueira and Bernardo Almada-Lobo. Hybrid simulation–optimization methods: A taxonomy and discussion. *Simulation Modelling Practice and Theory*, 46, 08 2014. doi: 10.1016/j.simpat.2014.03.007.

[48] Andrea Bartolini, Michele Lombardi, Michela Milano, and Luca Benini. Neuron constraints to model complex real-world problems. In Jimmy Lee, editor, *Principles and Practice of Constraint Programming – CP 2011*, pages 115–129, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[49] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. 2019.

[50] Nicholas Nethercote, Peter J. Stuckey, Ralph Becket, Sebastian Brand, Gregory J. Duck, and Guido Tack. Minizinc: Towards a standard cp modelling language. In Christian Bessière, editor, *Principles and Practice of Constraint Programming – CP 2007*, pages 529–543, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-74970-7.

[51] A. Schiendorfer, A. Knapp, G. Anders, and W. Reif. Minibrass: Soft constraints for minizinc. *Constraints*, 23(4):403–450, 2018. doi: 10.1007/s10601-018-9289-2.

[52] K. Kuchcinski and R. Szymanek. Jacop - java constraint programming solver. In *CP 2013*, 2013.

[53] U. Wilensky. Netlogo. URL ccl.northwestern.edu/netlogo.

[54] Souvik Barat. Actor based behavioural simulation as an aid for organisational decision making, 2018.

[55] Alvaro Gil. Artificial supply chain, 2012. URL modelingcommons.org/browse/one_model/3378.

[56] Tina O'Donnell, L. Maguire, R. McIvor, and P. Humphreys. Minimizing the bullwhip effect in a supply chain using genetic algorithms. *International Journal of Production Research*, 44:1523 – 1543, 2006.

[57] T. Chinna Pamulety and V.M. Pillai. Performance analysis of supply chains under customer demand information sharing using role play game. *International Journal of Industrial Engineering Computations*, 3(3):337–346, 2012. doi: 10.5267/j.ijiec.2012.02.002. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84871327336&doi=10.5267%2fj.ijiec.2012.02.002&partnerID=40&md5=4ab778d85819e988d1a257cc2d415b52. cited By 2.

[58] L. Rasmussen and U. Wilensky. Netlogo fruit wars model. URL ccl.northwestern.edu/netlogo/models/FruitWars.

[59] Siravat Teerasoponpong and Apichat Sopadang. A simulation-optimization approach for adaptive manufacturing capacity planning in small and medium-sized enterprises. *Expert Systems with Applications*, 168:114451, 2021. ISSN 0957-4174. doi: 10.1016/j.eswa.2020.114451.

[60] J. Asmundsson, R.L. Rardin, and R. Uzsoy. Tractable nonlinear production planning models for semiconductor wafer fabrication facilities. *IEEE Transactions on Semiconductor Manufacturing*, 19(1):95–111, 2006. doi: 10.1109/TSM.2005.863214.

[61] Christian Bessiere, Frédéric Koriche, Nadjib Lazaar, and Barry O'Sullivan. Constraint acquisition. *Artificial Intelligence*, 244:315–342, 2017. ISSN 0004-3702. doi: 10.1016/j.artint.2015.08.001. Combining Constraint Solving with Mining and Learning.

[62] K. Sheela and S N Deepa. Review on methods to fix number of hidden neurons in neural networks. *Mathematical Problems in Engineering*, 2013, 06 2013. doi: 10.1155/2013/425740.

[63] Warren S. Sarle. comp.ai.neural-nets faq, 1997–2002. URL `faqs.org/faqs/ai-faq/neural-nets/part3`.

[64] Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622, 2021. ISSN 0950-7051. doi: 10.1016/j.knosys.2020.106622.

[65] Gérard Verfaillie and Narendra Jussien. Constraint solving in uncertain and dynamic environments: A survey. *Constraints*, 10(3):253—-281, July 2005. ISSN 1383–7133. doi: 10.1007/s10601-005-2239-9.

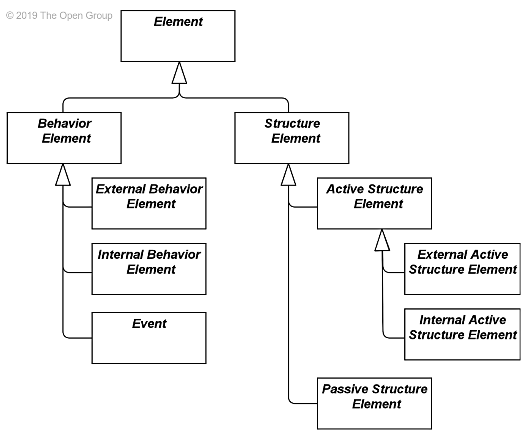
[66] H. Afshari, W. Hare, and S. Tesfamariam. Constrained multi-objective optimization algorithms: Review and comparison with application in reinforced concrete structures. *Applied Soft Computing Journal*, 83: 105631, 2019. doi: 10.1016/j.asoc.2019.105631.

[67] Lightbend. Akka, 2011–2021. URL `https://github.com/akka/akka`.

[68] The Open Group. Archimate specification, 2019. URL `pubs.opengroup.org/architecture/archimate3-doc`.

Appendix A

EM figures (from other literature)


© 2019 The Open Group
Element
Behavior Element
Structure Element
External Behavior Element
Internal Behavior Element
Event
Active Structure Element
External Active Structure Element
Internal Active Structure Element
Passive Structure Element

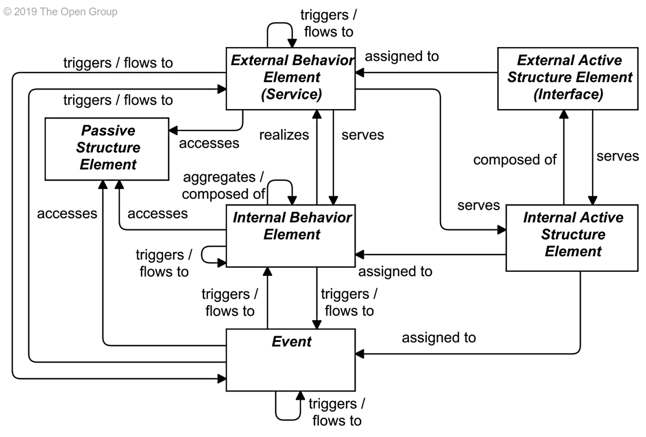


Figure A.1: Hierarchy (left) and metamodel (right) of Behaviour and Structure Elements found in ArchiMate. Structure elements can be subdivided into *active structure* elements and *passive structure* elements. Active structure elements are the subjects that can perform behavior. These can be further subdivided into *external* active structure elements (also called interfaces) and *internal* active structure elements. An internal active structure element represents an entity that is capable of performing behavior. An external active structure element, called an interface, represents a point of access where one or more services are provided to the environment. A passive structure element represents an element on which behavior is performed. Passive structure elements are often information or data objects, but they can also represent physical objects. Behavior elements can be subdivided into *internal behavior* elements, *external behavior* elements (also called services), and *events*. Behavior elements represent the dynamic aspects of the enterprise. An internal behavior element represents a unit of activity that can be performed by one or more active structure elements. An external behavior element, called a service, represents an explicitly defined exposed behavior. An event represents a state change [68].

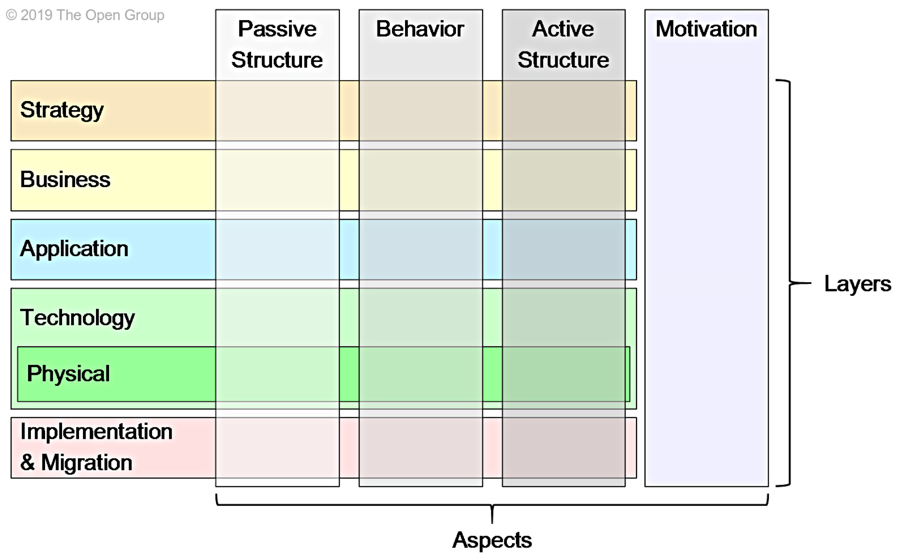


Figure A.2: ArchiMate Full Framework: layers and concepts. Each layer provides service to the one that is on top of it. The general structure of models within the different layers is similar. The same types of elements and relationships are used, although their exact nature and granularity differ. The Physical layer is considered to be part of the Technology layer. The Strategy layer, Implementation & Migration layer and the Motivation aspect were added in the full version and are not part of the initial ArchiMate framework [68].

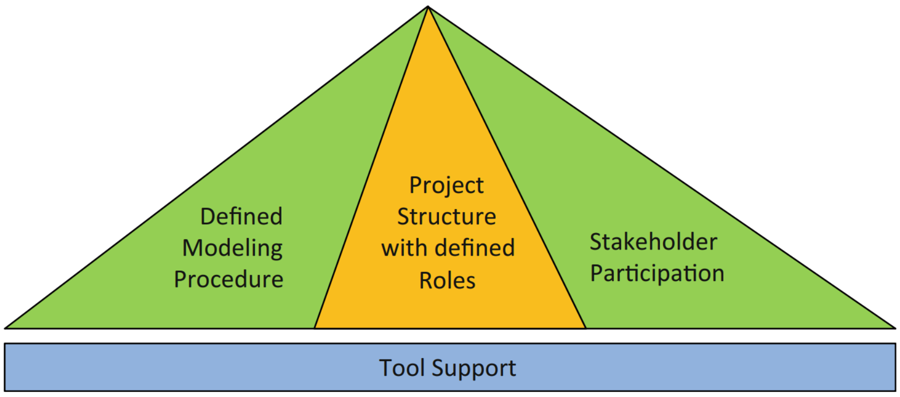

Defined Modeling Procedure
Project Structure with defined Roles
Stakeholder Participation
Tool Support

Figure A.3: Basic elements of the 4EM—framework. The *Defined Modeling Procedure* refers to a fixed notation used by 4EM to define procedures. The *Project structure with defined Roles* refers to evaluating the performance of EM in the form of a project with predetermined roles. The *Stakeholder Participation* involves enterprise stakeholders and domain experts. [3]

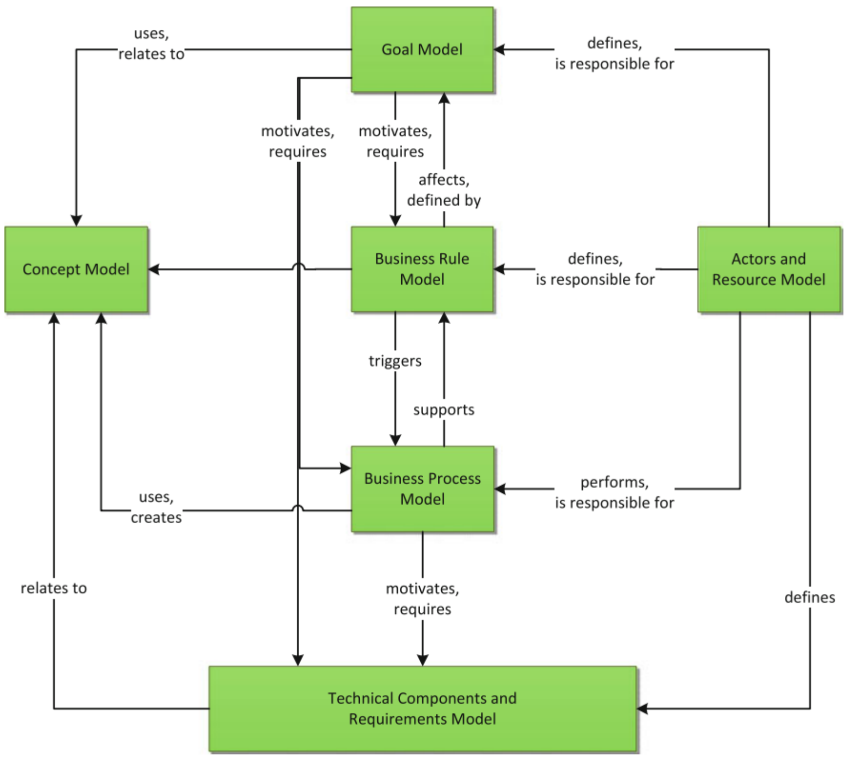
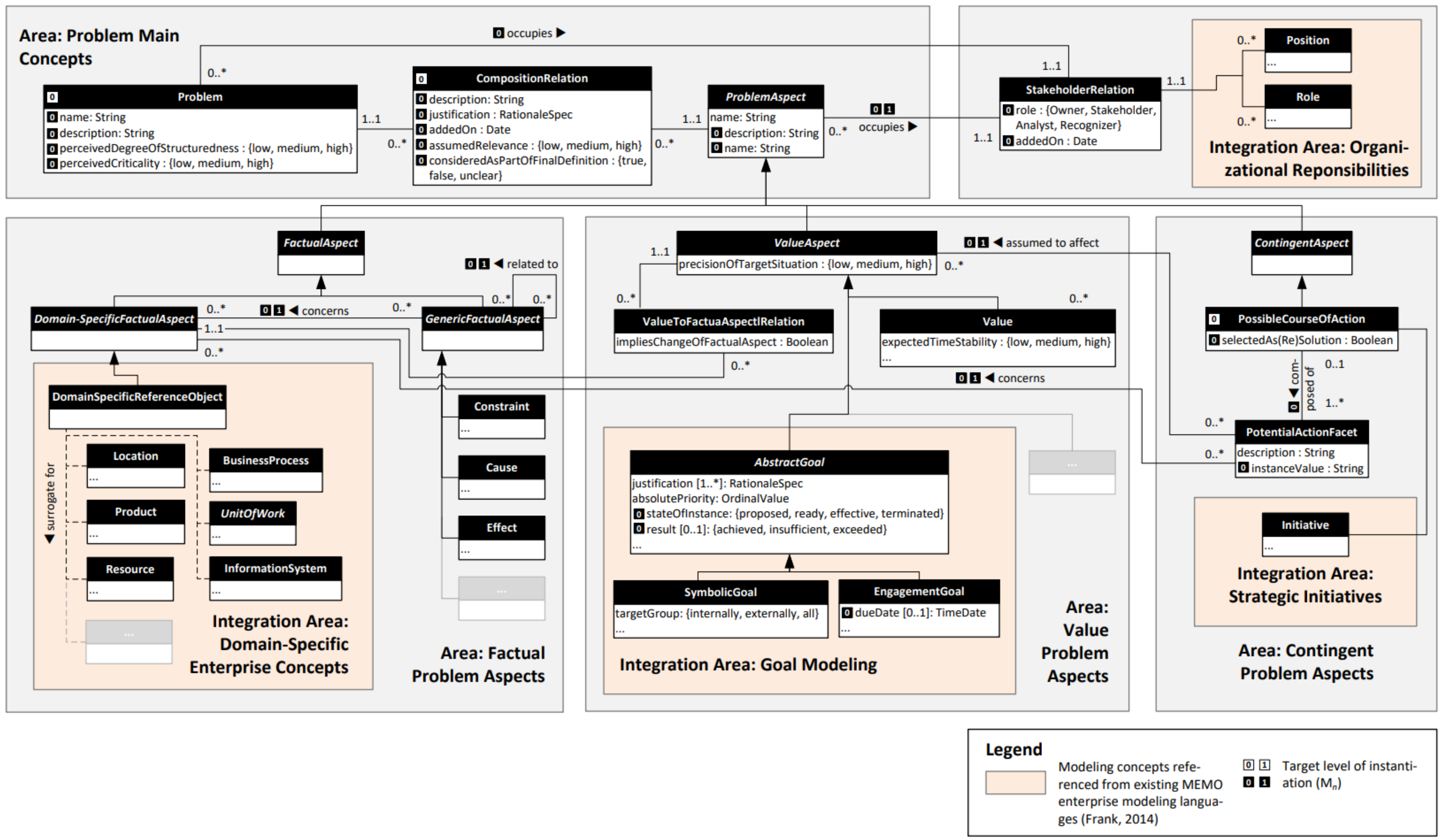
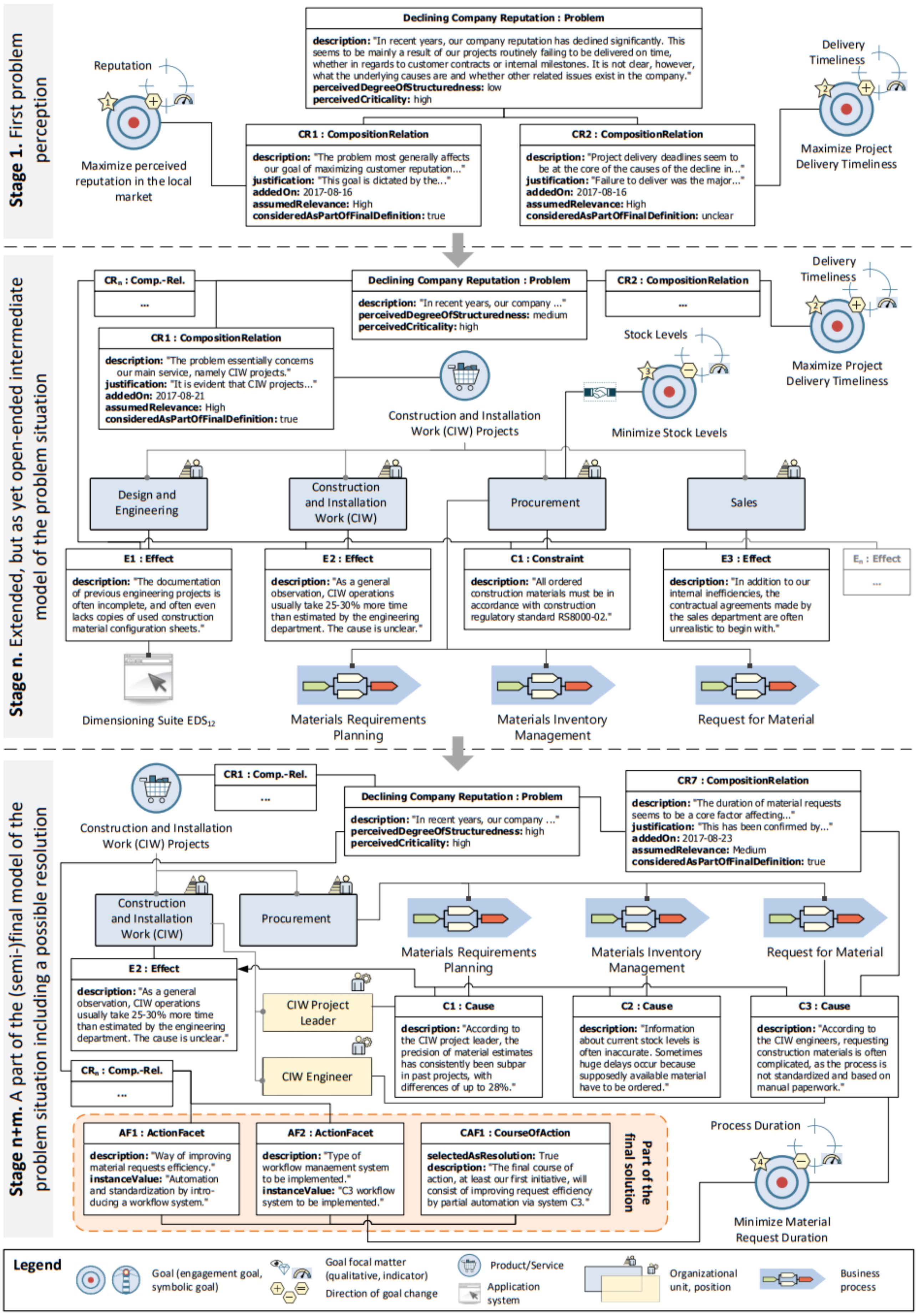


Figure A.4: Sub-models of the 4EM approach and their inter-model relationships [3].



Area: Problem Main Concepts
occupies
0..*
Problem
name: String
description: String
perceivedDegreeOfStructuredness : {low, medium, high}
perceivedCriticality : {low, medium, high}
1..1
0..*
CompositionRelation
description: String
justification : RationaleSpec
addedOn : Date
assumedRelevance : {low, medium, high}
consideredAsPartOfFinalDefinition : {true, false, unclear}
1..1
0..*
ProblemAspect
name: String
description: String
name: String
0..*
occupies
1..1
StakeholderRelation
role : {Owner, Stakeholder, Analyst, Recognizer}
addedOn : Date
1..1
1..1
0..*
Position
...
Role
...
0..*
Integration Area: Organizational Reponsibilities
FactualAspect
related to
Domain-SpecificFactualAspect
0..*
concerns
0..*
GenericFactualAspect
0..*
0..*
1..1
0..*
DomainSpecificReferenceObject
surrogate for
Location
...
BusinessProcess
...
Product
...
UnitOfWork
...
Resource
...
InformationSystem
...
...
Integration Area: Domain-Specific Enterprise Concepts
Constraint
...
Cause
...
Effect
...
...
Area: Factual Problem Aspects
ValueAspect
precisionOfTargetSituation : {low, medium, high}
1..1
assumed to affect
0..*
0..*
ValueToFactuaAspectlRelation
impliesChangeOfFactualAspect : Boolean
0..*
0..*
Value
expectedTimeStability : {low, medium, high}
...
concerns
AbstractGoal
justification [1..*]: RationaleSpec
absolutePriority: OrdinalValue
stateOfInstance: {proposed, ready, effective, terminated}
result [0..1]: {achieved, insufficient, exceeded}
...
SymbolicGoal
targetGroup: {internally, externally, all}
...
EngagementGoal
dueDate [0..1]: TimeDate
...
...
Integration Area: Goal Modeling
Area: Value Problem Aspects
ContingentAspect
PossibleCourseOfAction
selectedAs(Re)Solution : Boolean
composed of
0..1
1..*
0..*
0..*
PotentialActionFacet
description : String
instanceValue : String
Initiative
...
Integration Area: Strategic Initiatives
Area: Contingent Problem Aspects
Legend
Modeling concepts referenced from existing MEMO enterprise modeling languages (Frank, 2014)
Target level of instantiation (M_n)

Figure A.5: Meta model of the proposed problem conceptualization in the context of EM by Bock et al. [18].



Stage 1. First problem perception
Declining Company Reputation : Problem
description: "In recent years, our company reputation has declined significantly. This seems to be mainly a result of our projects routinely failing to be delivered on time, whether in regards to customer contracts or internal milestones. It is not clear, however, what the underlying causes are and whether other related issues exist in the company."
perceivedDegreeOfStructuredness: low
perceivedCriticality: high
Reputation
1
Maximize perceived reputation in the local market
CR1 : CompositionRelation
description: "The problem most generally affects our goal of maximizing customer reputation..."
justification: "This goal is dictated by the..."
addedOn: 2017-08-16
assumedRelevance: High
consideredAsPartOfFinalDefinition: true
CR2 : CompositionRelation
description: "Project delivery deadlines seem to be at the core of the causes of the decline in..."
justification: "Failure to deliver was the major..."
addedOn: 2017-08-16
assumedRelevance: High
consideredAsPartOfFinalDefinition: unclear
Delivery Timeliness
2
Maximize Project Delivery Timeliness
Stage n. Extended, but as yet open-ended intermediate model of the problem situation
CRn : Comp.-Rel.
...
Declining Company Reputation : Problem
description: "In recent years, our company ..."
perceivedDegreeOfStructuredness: medium
perceivedCriticality: high
CR2 : CompositionRelation
...
Delivery Timeliness
2
Maximize Project Delivery Timeliness
CR1 : CompositionRelation
description: "The problem essentially concerns our main service, namely CIW projects."
justification: "It is evident that CIW projects..."
addedOn: 2017-08-21
assumedRelevance: High
consideredAsPartOfFinalDefinition: true
Construction and Installation Work (CIW) Projects
Stock Levels
3
Minimize Stock Levels
Design and Engineering
Construction and Installation Work (CIW)
Procurement
Sales
E1 : Effect
description: "The documentation of previous engineering projects is often incomplete, and often even lacks copies of used construction material configuration sheets."
E2 : Effect
description: "As a general observation, CIW operations usually take 25-30% more time than estimated by the engineering department. The cause is unclear."
C1 : Constraint
description: "All ordered construction materials must be in accordance with construction regulatory standard RS8000-02."
E3 : Effect
description: "In addition to our internal inefficiencies, the contractual agreements made by the sales department are often unrealistic to begin with."
En : Effect
...
Dimensioning Suite EDS12
Materials Requirements Planning
Materials Inventory Management
Request for Material
Stage n+m. A part of the (semi-)final model of the problem situation including a possible resolution
Construction and Installation Work (CIW) Projects
CR1 : Comp.-Rel.
...
Declining Company Reputation : Problem
description: "In recent years, our company ..."
perceivedDegreeOfStructuredness: high
perceivedCriticality: high
CR7 : CompositionRelation
description: "The duration of material requests seems to be a core factor affecting..."
justification: "This has been confirmed by..."
addedOn: 2017-08-23
assumedRelevance: Medium
consideredAsPartOfFinalDefinition: true
Construction and Installation Work (CIW)
Procurement
Materials Requirements Planning
Materials Inventory Management
Request for Material
E2 : Effect
description: "As a general observation, CIW operations usually take 25-30% more time than estimated by the engineering department. The cause is unclear."
CIW Project Leader
CIW Engineer
C1 : Cause
description: "According to the CIW project leader, the precision of material estimates has consistently been subpar in past projects, with differences of up to 28%."
C2 : Cause
description: "Information about current stock levels is often inaccurate. Sometimes huge delays occur because supposedly available material have to be ordered."
C3 : Cause
description: "According to the CIW engineers, requesting construction materials is often complicated, as the process is not standardized and based on manual paperwork."
CRn : Comp.-Rel.
...
AF1 : ActionFacet
description: "Way of improving material requests efficiency."
instanceValue: "Automation and standardization by introducing a workflow system."
AF2 : ActionFacet
description: "Type of workflow management system to be implemented."
instanceValue: "C3 workflow system to be implemented."
CAF1 : CourseOfAction
selectedAsResolution: True
description: "The final course of action, at least our first initiative, will consist of improving request efficiency by partial automation via system C3."
Part of the final solution
Process Duration
4
Minimize Material Request Duration
Legend
Goal (engagement goal, symbolic goal)
Goal focal matter (qualitative, indicator)
Direction of goal change
Product/Service
Application system
Organizational unit, position
Business process

Figure A.6: Example Illustration of the Intended Use of the Meta Model from Figure A.5 [18].

Appendix B

NN embedding in MiniZinc Example (Hotel Slaaplust)

```
% auto generated file, representing a neural network
% features in: ['receptionists_amount', 'p_incoming_t1', 'p_incoming_t3', 'p_showup', '
    spread_factor', 'satisfied_max_quetime', 'checkin_max_quetime', 'reservation_max_quetime',
     'noshow_max_time', 't1_duration', 't2_duration', 't3_duration', 't4_duration', '
    t5_duration', 't6_duration', 'spread_horizon', 'stay_duration', 'time_between_t1_and_t3']
% features out: ['t1_loss', 't2_loss', 't3_loss', 'unsatisfied_rate']
% layerCount: 2
% layerWidth: 8

% use these to access output parameters
array[Time] of var float: t1_loss;
array[Time] of var float: t2_loss;
array[Time] of var float: t3_loss;
array[Time] of var float: unsatisfied_rate;


% Node constraints
% Layer 0
array[Time] of var float: n_0_0_ratio;
constraint forall(t in Time) (n_0_0_ratio[t] = max((-0.005552266724407673 *
    receptionists_amount[t] + 3.378122568130493 * p_incoming_t1[t] + 3.0774800777435303 *
    p_incoming_t3[t] + 1.715173125267O288 * p_showup[t] + -3.274698257446289 * spread_factor[t
    ] + -0.02187654748558998 * satisfied_max_quetime[t] + 0.00501234969124198 *
    checkin_max_quetime[t] + 0.03480386361479759 * reservation_max_quetime[t] +
    -0.009201295673847198 * noshow_max_time[t] + 0.002582360291853547 * t1_duration[t] +
    -0.005686505697667599 * t2_duration[t] + 0.019368739798665047 * t3_duration[t] +
    0.01745329611003399 * t4_duration[t] + 0.016309792175889015 * t5_duration[t] +
    0.03850631043314934 * t6_duration[t] + -5.689562749466859e-05 * spread_horizon[t] +
    0.0009009161149151623 * stay_duration[t] + -0.00052812066860497 * time_between_t1_and_t3[t
    ] + 0.030590936541557312), 0));
array[Time] of var float: n_0_1_ratio;
constraint forall(t in Time) (n_0_1_ratio[t] = max((0.010291551239788532 *
    receptionists_amount[t] + -1.351742148399353 * p_incoming_t1[t] + -2.4481403827667236 *
    p_incoming_t3[t] + -0.7043818831443787 * p_showup[t] + 3.800250291824341 * spread_factor[t
    ] + 0.013074466958642006 * satisfied_max_quetime[t] + -0.030260436236858368 *
    checkin_max_quetime[t] + 0.0022348202764987946 * reservation_max_quetime[t] +
    -0.003074843203648925 * noshow_max_time[t] + -0.012048905715346336 * t1_duration[t] +
    -0.003822901751846075 * t2_duration[t] + -0.018087496981024742 * t3_duration[t] +
    -0.019248997792601585 * t4_duration[t] + -0.023421267047524452 * t5_duration[t] +
    0.002232118509709835 * t6_duration[t] + -0.002202181378379464 * spread_horizon[t] +
    -0.00605540769174695 * stay_duration[t] + -0.005998273845762014 * time_between_t1_and_t3[t
    ] + 1.4154807329177856), 0));
array[Time] of var float: n_0_2_ratio;
constraint forall(t in Time) (n_0_2_ratio[t] = max((-0.03684801235795021 *
    receptionists_amount[t] + 0.05957389250397682 * p_incoming_t1[t] + -0.12520712614059448 *
    p_incoming_t3[t] + -0.042958687990903854 * p_showup[t] + -0.12237079441547394 *
    spread_factor[t] + -0.06495683640241623 * satisfied_max_quetime[t] + 0.030673811212182045
    * checkin_max_quetime[t] + -0.033833667635917664 * reservation_max_quetime[t] +
    0.004254186991602182 * noshow_max_time[t] + -0.02025914192199707 * t1_duration[t] +
    -0.015918176621198654 * t2_duration[t] + -0.11611270159482956 * t3_duration[t] +
    -0.032978355884552 * t4_duration[t] + -0.09965482354164124 * t5_duration[t] +
    0.056212928146123886 * t6_duration[t] + -0.10288435220718384 * spread_horizon[t] +
    -0.059540633112192154 * stay_duration[t] + -0.055290017277002335 * time_between_t1_and_t3[
    t] + -0.01879778690636158), 0));
```

```
array[Time] of var float: n_0_3_ratio;
constraint forall(t in Time) (n_0_3_ratio[t] = max((-0.03186827152967453 *
    receptionists_amount[t] + 4.217875957489014 * p_incoming_t1[t] + 5.286959171295166 *
    p_incoming_t3[t] + 0.6190303564071655 * p_showup[t] + -0.04575213044881821 * spread_factor
    [t] + -0.0021041170693933964 * satisfied_max_quetime[t] + 0.025881601497530937 *
    checkin_max_quetime[t] + 0.01810804009437561 * reservation_max_quetime[t] +
    -0.00018785281281452626 * noshow_max_time[t] + 0.014143881388008595 * t1_duration[t] +
    0.006068131886422634 * t2_duration[t] + 0.020353976637125015 * t3_duration[t] +
    0.024426260963082314 * t4_duration[t] + 0.02486073412001133 * t5_duration[t] +
    0.019518844783306122 * t6_duration[t] + -0.00027341899112798274 * spread_horizon[t] +
    -0.0001343473413726315 * stay_duration[t] + 0.0001390925608575344 * time_between_t1_and_t3
    [t] + -2.2312560081481934), 0));
array[Time] of var float: n_0_4_ratio;
constraint forall(t in Time) (n_0_4_ratio[t] = max((-0.0597444586455822 * receptionists_amount
    [t] + 5.391207695007324 * p_incoming_t1[t] + 6.350437164306641 * p_incoming_t3[t] +
    0.11786403506994247 * p_showup[t] + 0.1710611879825592 * spread_factor[t] +
    -6.86878920532763e-05 * satisfied_max_quetime[t] + 0.00917316135019064 *
    checkin_max_quetime[t] + 0.02067706361413002 * reservation_max_quetime[t] +
    -0.00020089604367967695 * noshow_max_time[t] + 0.017801247537136078 * t1_duration[t] +
    0.0042007663287222385 * t2_duration[t] + 0.017808249220252037 * t3_duration[t] +
    0.024792427197098732 * t4_duration[t] + 0.0230131093412637 7 * t5_duration[t] +
    0.028185555711388588 * t6_duration[t] + -0.00029834324959665537 * spread_horizon[t] +
    -0.000199031550437212 * stay_duration[t] + 0.0003771412593778223 * time_between_t1_and_t3[
    t] + -1.2267895936965942), 0));
array[Time] of var float: n_0_5_ratio;
constraint forall(t in Time) (n_0_5_ratio[t] = max((-0.054250624030828476 *
    receptionists_amount[t] + -0.0994448810815811 2 * p_incoming_t1[t] + -0.015479475259780884
    * p_incoming_t3[t] + 0.0537140592932701 1 * p_showup[t] + -0.0348622128367424 *
    spread_factor[t] + 0.06607885658740997 * satisfied_max_quetime[t] + 0.0394648648798 4657 *
    checkin_max_quetime[t] + 0.07296678423881531 * reservation_max_quetime[t] +
    -0.07333448529243469 * noshow_max_time[t] + 0.028902588412165642 * t1_duration[t] +
    0.016017474234104156 * t2_duration[t] + -0.060940150171518326 * t3_duration[t] +
    -0.07674882560968399 * t4_duration[t] + -0.06709527224302292 * t5_duration[t] +
    -0.05910588428378105 * t6_duration[t] + -0.06185930594801903 * spread_horizon[t] +
    0.035853080451488495 * stay_duration[t] + -0.04734676331281662 * time_between_t1_and_t3[t]
     + -0.05862151458859444), 0));
array[Time] of var float: n_0_6_ratio;
constraint forall(t in Time) (n_0_6_ratio[t] = max((0.14158278703689575 * receptionists_amount
    [t] + -0.7393486499786377 * p_incoming_t1[t] + -7.296354293823242 * p_incoming_t3[t] +
    -4.997591495513916 * p_showup[t] + 4.318524360656738 * spread_factor[t] +
    0.14878331124782562 * satisfied_max_quetime[t] + 0.05598339810967445 4 *
    checkin_max_quetime[t] + 0.1853535920381546 * reservation_max_quetime[t] +
    -0.0005493673961609602 * noshow_max_time[t] + -0.015184195712208748 * t1_duration[t] +
    0.03735168650746345 5 * t2_duration[t] + -0.00047234963858500123 * t3_duration[t] +
    0.0005559903802350163 * t4_duration[t] + -0.016614699736237526 * t5_duration[t] +
    0.1526528149843216 * t6_duration[t] + -0.00046501983888447285 * spread_horizon[t] +
    0.0010347227798774838 * stay_duration[t] + 0.00234037777408957 5 * time_between_t1_and_t3[t
    ] + 0.7485445141792297), 0));
array[Time] of var float: n_0_7_ratio;
constraint forall(t in Time) (n_0_7_ratio[t] = max((0.019266281276941 3 * receptionists_amount[
    t] + -1.2240017652511597 * p_incoming_t1[t] + -3.169340610504150 4 * p_incoming_t3[t] +
    -1.104755163192749 * p_showup[t] + 5.627672672271728 5 * spread_factor[t] +
    0.02762724086642265 3 * satisfied_max_quetime[t] + -0.007977697066962719 *
    checkin_max_quetime[t] + 0.01703265123069286 3 * reservation_max_quetime[t] +
    -6.562373891938478e-05 * noshow_max_time[t] + -0.007069487124681473 * t1_duration[t] +
    0.00218272116035223 * t2_duration[t] + -0.007785466033965349 * t3_duration[t] +
    -0.0092034125700593 * t4_duration[t] + -0.011222709901630878 * t5_duration[t] +
    0.013470616191625595 * t6_duration[t] + -0.00023711241374257952 * spread_horizon[t] +
    6.558325549121946e-05 * stay_duration[t] + 0.0004069800488650799 * time_between_t1_and_t3[
    t] + 2.952331781387329), 0));
% Layer 1
array[Time] of var float: n_1_0_ratio;
constraint forall(t in Time) (n_1_0_ratio[t] = max((-0.0021118773147463 8 * n_0_0_ratio[t] +
    0.27164512872695923 * n_0_1_ratio[t] + -0.08921993523836136 * n_0_2_ratio[t] +
    0.2458434402942657 5 * n_0_3_ratio[t] + 0.292856425046920 8 * n_0_4_ratio[t] +
    0.10125173628330231 * n_0_5_ratio[t] + -0.16388389468193054 * n_0_6_ratio[t] +
    0.9632182121276855 * n_0_7_ratio[t] + -0.0006364168948493898), 0));
array[Time] of var float: n_1_1_ratio;
constraint forall(t in Time) (n_1_1_ratio[t] = max((0.41024819016456604 * n_0_0_ratio[t] +
    0.37775570154190063 * n_0_1_ratio[t] + -0.12252442538738251 * n_0_2_ratio[t] +
    0.12611645460128784 * n_0_3_ratio[t] + 0.31531932950019836 * n_0_4_ratio[t] +
    0.12667495012283325 * n_0_5_ratio[t] + 0.019637376070022583 * n_0_6_ratio[t] +
    0.3881857097148895 * n_0_7_ratio[t] + -1.2816541194915771), 0));
array[Time] of var float: n_1_2_ratio;
constraint forall(t in Time) (n_1_2_ratio[t] = max((-0.1407649070024490 4 * n_0_0_ratio[t] +
    -1.0054539442062378 * n_0_1_ratio[t] + -0.079989492893219 * n_0_2_ratio[t] +
    -0.08161623775959015 * n_0_3_ratio[t] + -0.07487081736326218 * n_0_4_ratio[t] +
    -0.0747442469008163 * n_0_5_ratio[t] + 0.13608750700950623 * n_0_6_ratio[t] +
    -4.54027795791626 * n_0_7_ratio[t] + -0.9716749787330627), 0));
array[Time] of var float: n_1_3_ratio;
constraint forall(t in Time) (n_1_3_ratio[t] = max((-0.18497268855571747 * n_0_0_ratio[t] +
    0.03334296122193365 * n_0_1_ratio[t] + 0.04991967976093292 * n_0_2_ratio[t] +
    0.7971280813217163 * n_0_3_ratio[t] + -0.7146216630935669 * n_0_4_ratio[t] +
    -0.11048527806997299 * n_0_5_ratio[t] + 0.008181902579963207 * n_0_6_ratio[t] +
    0.05119796097278595 * n_0_7_ratio[t] + -1.469698429107666), 0));
array[Time] of var float: n_1_4_ratio;
constraint forall(t in Time) (n_1_4_ratio[t] = max((-0.057326991111040115 * n_0_0_ratio[t] +
    -0.6056956648826599 * n_0_1_ratio[t] + -0.06932283192873001 * n_0_2_ratio[t] +
    0.019254066050052643 * n_0_3_ratio[t] + 0.07904529571533203 * n_0_4_ratio[t] +
    -0.0068346536718308926 * n_0_5_ratio[t] + 0.022476140409708023 * n_0_6_ratio[t] +
    0.7605360746383667 * n_0_7_ratio[t] + -1.7248374223709106), 0));
array[Time] of var float: n_1_5_ratio;
constraint forall(t in Time) (n_1_5_ratio[t] = max((-0.31671708822250366 * n_0_0_ratio[t] +
    0.4486914873123169 * n_0_1_ratio[t] + 0.044689375907182693 * n_0_2_ratio[t] +
    0.19285517930984497 * n_0_3_ratio[t] + -0.6415403485298157 * n_0_4_ratio[t] +
    0.09387854486703873 * n_0_5_ratio[t] + 0.06474684923887253 * n_0_6_ratio[t] +
    0.2265612632036209 * n_0_7_ratio[t] + 1.1635560989379883), 0));
array[Time] of var float: n_1_6_ratio;
constraint forall(t in Time) (n_1_6_ratio[t] = max((0.2993699908256531 * n_0_0_ratio[t] +
    -0.4600647985935211 * n_0_1_ratio[t] + 0.01806819438934326 * n_0_2_ratio[t] +
    -0.1321253627538681 * n_0_3_ratio[t] + 0.579810619354248 * n_0_4_ratio[t] +
    -0.004477897193282843 * n_0_5_ratio[t] + 0.3054755628108978 * n_0_6_ratio[t] +
    0.4273982048034668 * n_0_7_ratio[t] + 0.00012885835894849151), 0));
array[Time] of var float: n_1_7_ratio;
constraint forall(t in Time) (n_1_7_ratio[t] = max((-0.0971827581524849 * n_0_0_ratio[t] +
    0.3970702588558197 * n_0_1_ratio[t] + 0.013660607859492302 * n_0_2_ratio[t] +
    0.33407458662986755 * n_0_3_ratio[t] + -0.28272804617881775 * n_0_4_ratio[t] +
    0.15030862390995026 * n_0_5_ratio[t] + -0.009278697893023491 * n_0_6_ratio[t] +
    0.14677061140537262 * n_0_7_ratio[t] + -0.4624299705028534), 0));
% Layer 2
array[Time] of var float: n_2_0_ratio;
constraint forall(t in Time) (n_2_0_ratio[t] = (0.508807361125946 * n_1_0_ratio[t] +
    -0.3979931175708771 * n_1_1_ratio[t] + -0.8692365884780884 * n_1_2_ratio[t] +
    0.6767562031745911 * n_1_3_ratio[t] + -0.8868420124053955 * n_1_4_ratio[t] +
    -0.14117871224880219 * n_1_5_ratio[t] + 0.3153606951236725 * n_1_6_ratio[t] +
    -0.7875701189041138 * n_1_7_ratio[t] + -1.6333240270614624));
array[Time] of var float: n_2_1_ratio;
constraint forall(t in Time) (n_2_1_ratio[t] = (-0.013616766780614853 * n_1_0_ratio[t] +
    0.5143668055534363 * n_1_1_ratio[t] + 0.025865338742733 * n_1_2_ratio[t] +
    1.3853068351745605 * n_1_3_ratio[t] + -0.05062967166304588 * n_1_4_ratio[t] +
    -1.1495575904846191 * n_1_5_ratio[t] + 0.190445214509964 * n_1_6_ratio[t] +
    0.04320884123444557 * n_1_7_ratio[t] + -1.1651642322540283));
array[Time] of var float: n_2_2_ratio;
constraint forall(t in Time) (n_2_2_ratio[t] = (0.5581948757171631 * n_1_0_ratio[t] +
    -0.4298996031284332 * n_1_1_ratio[t] + -0.9501732587814331 * n_1_2_ratio[t] +
    0.6974130272865295 * n_1_3_ratio[t] + -0.992718517780304 * n_1_4_ratio[t] +
    -0.11991114169359207 * n_1_5_ratio[t] + 0.34147703647613525 * n_1_6_ratio[t] +
    -0.8093789219856262 * n_1_7_ratio[t] + -1.731464147567749));
array[Time] of var float: n_2_3_ratio;
constraint forall(t in Time) (n_2_3_ratio[t] = (0.803183376789093 * n_1_0_ratio[t] +
    -0.6490669846534729 * n_1_1_ratio[t] + -1.3723446130752563 * n_1_2_ratio[t] +
    0.7747349143028259 * n_1_3_ratio[t] + -1.5042433738708496 * n_1_4_ratio[t] +
    -0.12861618399620056 * n_1_5_ratio[t] + 0.539939820766449 * n_1_6_ratio[t] +
    -0.44107893109321594 * n_1_7_ratio[t] + -1.1965217590332031));
% Add activation function
constraint forall(t in Time) (t1_loss[t] * (1 + exp(-n_2_0_ratio[t])) = 1.0);
constraint forall(t in Time) (t2_loss[t] * (1 + exp(-n_2_1_ratio[t])) = 1.0);
constraint forall(t in Time) (t3_loss[t] * (1 + exp(-n_2_2_ratio[t])) = 1.0);
constraint forall(t in Time) (unsatisfied_rate[t] * (1 + exp(-n_2_3_ratio[t])) = 1.0);
```

Appendix C

Hotel Slaaplust

De belangrijkste dienstverlening van hotel ”Slaaplust” is het verhuren van kamers. De verhuring van een kamer van een bepaald kamertype aan een klant gedurende een aantal aaneengesloten dagen heet een hotelverblijf. De eerste dag heet de aankomstdag en de laatste de vertrekdag. Met de verblijfsperiode worden de dagen bedoeld van (en inclusief) de aankomstdag tot (en exclusief) de vertrekdag. Tijdens hun verblijf kunnen de gasten ook van andere faciliteiten gebruik maken, zoals de bar, het restaurant, de sauna en het zwembad.

De directie bepaalt welke kamertypes er zijn en wat de bijbehorende prijzen zijn. Ook stelt de directie vast welke kamers op welke dagen beschikbaar zijn (bij niet-beschikbaarheid moet men bijvoorbeeld denken aan onderhoud). Hotelverblijven worden meestal vooraf gereserveerd, maar men kan ook op de bonnefooi het hotel binnenstappen en een kamer krijgen. Zo'n geval wordt opgevat als het vlak na elkaar reserveren en inboeken. Het reserveren gebeurt doorgaans niet door de gasten zelf maar door bijvoorbeeld de firma waarvoor zij werken of door een reisbureau. Natuurlijke personen en rechtspersonen die reserveringen plaatsen, worden klanten genoemd. Van hen worden enige tijd namen en adressen bewaard voor public-relations-doeleinden. Aanvragen voor reserveringen kunnen binnenkomen per brief, per telefoon of per fax. De volgende gegevens worden daarbij steeds vastgelegd: naam en adres van de klant, de datum van reservering, de aankomstdag, de verblijfsduur en het kamertype (een- of tweepersoons of suite). Deze gegevens worden genoteerd in het reserveringenboek. Voor elke reservering wordt een aparte regel gebruikt. De regels zijn doorlopend genummerd, waardoor men altijd gemakkelijk kan verwijzen naar een bepaald hotelverblijf. Indien een klant twee of meer kamers wil reserveren, al of niet van hetzelfde kamertype, wordt dat door het hotel opgevat als evenzoveel reserveringen; een reservering betreft dus steeds één hotelverblijf. Reserveringsaanvragen worden in volgorde van binnenkomst afgehandeld. Indien de klant te boek staat als onbetrouwbaar, wordt de reservering vriendelijk doch beslist geweigerd. De directie stelt regelmatig een lijst op van onbetrouwbare klanten. Een andere voorwaarde voor het accepteren van een reservering is uiteraard het vrij zijn van een kamer van het gewenste type in de verblijfsperiode. Deze informatie is bevat in het bezettingsoverzicht, dat is een tabel waarin voor een jaar vooruit het aantal vrije kamers per kamertype wordt bijgehouden. Indien een aanvraag wordt gehonoreerd, ontvangt de klant daarvan een schriftelijke bevestiging. Afwijzingen worden ook medegedeeld.

Van de gasten die tezamen op één kamer verblijven, wordt er één aangemerkt als de hoofdgast. Alleen deze persoon kan nota's paraferen van de bar, het restaurant etc.. In het vervolg wordt met ”gast” steeds de hoofdgast bedoeld. Op de dag van aankomst meldt de gast zich bij de receptie om in te boeken. Dan pas wordt een bepaalde kamer, geïdentificeerd door een kamernummer, toegewezen. De inboeking wordt genoteerd in het gastenboek. Dit boek bevat per dag een bladzijde waarop per kamer een regel beschikbaar is. Het kamernummer is daarop al ingevuld. Van elke dag die binnen de verblijfsperiode van een hotelverblijf valt, wordt op de regel van de toegewezen kamer, de naam van de gast en het verblijfsnummer (uit het reserveringenboek) genoteerd. Als een hotelverblijf op de aankomstdag vóór 18.00 uur nog niet is ingeboekt, vervalt de reservering en komt de kamer weer vrij. Zo'n interne annulering wordt aangetekend in het reserveringenboek. In alle gevallen wordt er een boetefactuur gestuurd naar de klant, die per bank of giro dient te worden betaald. Op de dag van vertrek meldt de gast zich bij de receptie om uit te boeken. Hij of zij ontvangt dan de verblijfsfactuur, waarop alle gemaakte kosten staan vermeld. Het totaalbedrag betaalt de gast vervolgens bij de kassier (die in een beveiligd glazen hokje naast de receptiebalie zit). Na betaling krijgt de gast de factuur, die nu als betaald is gestempeld, van de receptie terug. Het verblijf van de gast is nu definitief afgesloten [12].

Appendix D

ABC University MiniZinc and MiniBrass implementation

abc_university.mzn

```
% choose from a few calendars
enum Calendar;
Calendar: calendarUsed;

%%% decision Variables corresponds with NN input

% 1. how many academics of each catagory we hire is a strategic decision
% these three should sum to 1
var 0.0..1.0: teachingAcademicRatio;
var 0.0..1.0: researchAcademicRatio;
var 0.0..1.0: hybridAcademicRatio;
constraint (teachingAcademicRatio + researchAcademicRatio + hybridAcademicRatio = 0);

% 2. work priority for each category of academics is also a strategic decision
enum workPriorityTeachingOptions;
enum workPriorityHybridOptions;
enum workPriorityResearchOptions;

var workPriorityTeachingOptions: workPriorityTeaching;
var workPriorityHybridOptions: workPriorityHybrid;
var workPriorityResearchOptions: workPriorityResearch;

%%% observables correspond to output of the NN, and are used to define goals
% using ratios instead of integers has better performance, as it is easier to learn for the NN
var 0.0..1.0: papers_acceptance_ratio;
var 0.0..1.0: class_not_taken_ratio;
var 0.0..1.0: class_adequate_preparation_ratio;
var 0.0..1.0: complaints_per_module;
var 0.0..1.0: queries_per_module;

% papers submitted is a positive integer
var 0..infinity: papers_submitted;

% some other observables that does not necessarily match the goal, but we want to keep intact
var 0.0..1.0: free_time_student;
var 0.0..1.0: free_time_academic;

% take a soft constraint approach
solve minimize topLevelObjective;
```

abc_university.mbr

```
include "defs.mbr";

% we can set certaint goals based on thresholds to get a more multi-objective kind of solution
% decomposition of goals
PVS: improve_research_ranking = new WeightedCsp("improve_research_ranking") {
  soft-constraint papers_submitted: 'papers_submitted >= papers_submitted_th' :: weights('
    papers_submitted_w');
  soft-constraint papers_acceptance_ratio: 'papers_acceptance_ratio >=
    papers_acceptance_ratio_th' :: weights('papers_acceptance_ratio_w');
};

PVS: improve_teaching_ranking = new WeightedCsp("improve_teaching_ranking") {
  soft-constraint class_not_taken_ratio: 'class_not_taken_ratio >= class_not_taken_ratio_th'
    :: weights('class_not_taken_ratio_w');
  soft-constraint class_adequate_preparation_ratio: 'class_adequate_preparation_ratio <=
    class_adequate_preparation_ratio_th' :: weights('class_adequate_preparation_ratio_w');
  soft-constraint complaints_per_module: 'complaints_per_module >= complaints_per_module_th'
    :: weights('complaints_per_module_w');
  soft-constraint queries_per_module: 'queries_per_module >= queries_per_module_th' :: weights
    ('queries_per_module_w');
};

% condorcet voting among our two main objectives to improve overall ranking
solve vote([improve_research_ranking, improve_teaching_ranking], condorcet);
```

Appendix E

NetLogo Models Info

E.1 Artificial Supply Chain

WHAT IS IT?

This model is an artificial market with four types of participants. The first are the costumers who have a daily demand and according to their strategy can purchase daily or periodically if their stock is below some specific level. The second are the retailers who sell products to costumers and update their demand forecast by considering the sales and the stock rupture (lost sales). The third are the distributors (who follow a similar procedure) to send products to retailers. Finally, there are the factories who start producing once their inventory level is lower than some reorder point. The model can help students and professionals to understand better the supply chain with a single product, and how simple changes as the promotions, can affect the stocks levels and the demand calculation with a considerable amplitude, which is know as the bullwhip effect.

HOW IT WORKS

The customers demand is normally distributed with a given standard deviation. Both the mean and the variability are user parameters. The inventory values (product cost, holding costs and order costs), are also parameters from the model. The user can also modify the number of agents. Finally, the clients' strategy and the retailers and distributors strategy are also parameters. At every day, the costumers can score the purchase experience (if there are some), so the next period the customers will choose the best supplier. At the beginning, the score is given by taking into account only the distance from retailers (the nearest retailer will be the supplier), but with the time, the agents could change this perspective by punishing the supplier if the sale is incomplete. As to suppliers (retailers and distributors), they use linear regression for update their forecast and calculate the reorder point. If the stock is lower than this point, they place an order to their own supplier. If their strategy is (s,Q), they will only ask for the difference between the actual stock and the Economic Order Quantity (EOQ). If the strategy is (s,S), they will also consider the daily demand during the lead time. The last option is to repeat this last strategy but periodically. The factories does the same estimation but for starting the production in a fix daily rate calculated at the beginning of the simulation.

HOW TO USE IT

The user should fix some values and play the model (I recommend at least 720 periods for having some stable results), then collect statistics and play again. Different values will result in different costs and stock levels which can be compared in the analysis and conclusions phase.

THINGS TO NOTICE

See how the stock levels for each type of participant are so different but at the same time they follow the same patron?, this is because the inventory theory, no matter what strategy the agents, it will always happen.

THINGS TO TRY

Try to create promotions at some moment. See how they affect the stock levels and the inventory costs. See also how some retailer can increase the clients' number and hence the sales with these activities.

EXTENDING THE MODEL

It would be nice to improve the model with:

1. More sophisticated forecasting models
2. Add some operational costs to the suppliers, so the user could evaluate the profit of every business.
3. Add more products to the models. This will require some elasticity concept because the products could or could not be substitutes among them, so the availability and/or the price will affect the demand.
4. Finally, we can think in smarter suppliers, who update their strategy and also improve it, to compete and to increase the earnings.

CREDITS AND REFERENCES

Developed by Alvaro Gil at the Polytechnic School of Motreal alvaro.gil@polymtl.ca

If you mention this model in an academic publication, we ask that you include these citations for the model itself:

Gil, Alvaro (2012). Artificial supply chain. École Polytechnique de Montréal. alvaro.gil@polymtl.ca

About the NetLogo software - Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

COPYRIGHT NOTICE

E.2 Fruit Wars

WHAT IS IT?

The Fruit Wars model is intended to demonstrate how non-zero-sum economic environments can encourage cooperation and discourage violence. The foragers wander the map looking for fruit bushes. When they arrive at a fruit bush they gain energy through foraging until the fruit bush is exhausted of resources. These animals reproduce and pass on their characteristics to their offspring after gathering a certain amount of energy. They also make decisions based on heritable parameters about how to interact with other foraging animals. The foragers can choose to cooperate, threaten, fight or flee under different circumcstances.

In the model, foragers can either cooperate or fight based on heritable aggression attributes. The COLLABORATION-BONUS parameter in the model controls how beneficial cooperation is in terms of foraging and so one should expect to see the evolutionary equilbrium of the system move toward foragers with less violent tendencies when the model is run with higher COLLABORATION-BONUS settings.

HOW IT WORKS

Foragers in the model have 240-bit binary genomes:

- The sum of the first 120 bits determines the *strength*, *speed* and *intelligence* of the *foragers*. These three stats can have any values between 0 and 60, but their sum must always be 60.
- The sum of bits 120 to 180 determine the *forager*'s *reactive-aggression*, which determines the tendency to threaten when arriving at a bush and the tendency to threaten new arrivals when already at a bush. The sum of bits 180 to 240 determine the forager's *proactive-aggression*, which determines the tendency to fight back when threatened.

Each Tick in the Model

:

For each Forager:

- If no bushes are very nearby or my status is *fleeing*, move randomly based on the *speed* attribute. Consume 1 energy.
- If there is a bush nearby, move toward it a distance based on the *speed* attribute. Consume 1 energy.
- If I am very close to a bush, do the *arrival* procedure.
- If my status is *foraging*, gain some energy and take that energy away from the bush. If the bush runs out of energy, make it die, and remove the *foraging* status from all other *foragers* at that bush.
- If my energy is above 200, reproduce, lose 100 energy.
- If out of energy or older than the max-age parameter, die.

The arrival procedure:

- If there are no other *foragers* at the fruit bush, set status to *foraging.*
- If there are other *foragers* at the bush, the arriving *forager* chooses to threaten or collaborate probabalistically based on its *reactive-aggression* value.
- If my choice is *threaten*, other *foragers* at the bush choose to *flee* or *fight back* according to their *proactive-aggression* parameter.
- If any *fight back*, the arriving *forager* wins with a probability based on the difference of the arriving *forager*'s *strength* and the aggregate *strength* of all the other *foragers* choosing to *fight back.* If the arriving forager wins all of the *foragers* choosing to *fight back* die, otherwise the arriving *forager* dies.
- If all the other *foragers flee*, the arriving *forager* sets its status to *foraging.*
- If the arriving *forager* chooses to *collaborate* and there is one other present *forager*, that *forager* chooses to either allow the arriving *forager* to join or to *threaten* causing the arriving *forager* to flee.
- If there is only one *forager* present and it doesn't *threaten*, or there are multiple *foragers* present, the status of the arriving *forager* is set to *foraging.*

Reproduction:

Each *forager* has a binary genome which encodes the values of its parameters. When the *foragers* reproduce (asexually), the genome is passed on to the offspring and each position in the genome has a *rate-of-mutation* chance to flip values.

HOW TO USE IT

1. Set the *parameters* as desired.
2. Click SETUP.
3. Click GO.

Setup Parameters:

- INITIAL-FRUIT-BUSHES - The number of fruit bushes at the start.
- INITIAL-FORAGERS - The number of foragers at the start.

Visualization Parameters:

- SHOW-ENERGY? - displays the current energy of each individual forager in the view if enabled.
- VISUALIZATION - scales the color of the agent depending on the selection based on the value of that parameter with lighter colors representing a higher value.

Runtime Parameters:

- BUSH-GROWTH-CHANCE - The chance for a new fruit bush to grow at each turn.
- TICKS-TO-FLEE - When a forager flees, this is the number of ticks it must move randomly before it can seek a new fruit bush.
- RATE-OF-MUTATION - The probability that an individual bit will flip during the passage of bit genome from parent to offspring.
- COLLABORATION-BONUS - A factor used to determine the collaborative rate of foraging. Higher collaboration bonus means higher foraging rates for multiple foragers at a single bush.
- MAX-AGE - The maximum number of ticks an individual forager can live.

THINGS TO NOTICE

Observe the behavior of the foraging animals as they move around the map. Do you notice any of them fleeing from the fruit bushes? Do you notice many fatalities due to violent encounters (marked in the view by a red X?) How large of population can be sustained by the environment?

The model also has many graphs to observe:

- A histogram of different forager attributes at each tick.
- A standard line graph of the forager attributes at each tick.
- A graph of the ratio of homicides to other deaths.
- A graph of the average foraging rate per tick.

THINGS TO TRY

Try the different VISUALIZATION modes and observe the changes in the agent population at different model speeds and parameter settings. What sorts of patterns emerge?

Try adjusting the COLLABORATION-BONUS parameter. Oberve the behaviors and graphs discussed in the previous section. How do they vary with the COLLABORATION-BONUS parameter? How long does it generally take the model to reach a somewhat stable state, if ever?

EXTENDING THE MODEL

Extend the model such that foragers can fight, but not necessarily die. Also, add a mechanic so that if a forager is defeated in a fight, the victorious forager can take all or most of its resources. Additionally, one might incorporate some mechanism such that the foragers' likelihood to fight is determined by some function of their hunger.

NETLOGO FEATURES

This model makes extensive use of mutable NetLogo lists to implement bit genomes for agent evolution. It also uses custom turtle shapes designed in the *Turtle Shapes Editor*.

COPYRIGHT AND LICENSE