Extending the Thick Level Set Approach

Plasticity, Parallel Computing and Cohesive Cracks

Taumaturgo Mororo, L.A.

DOI
10.4233/uuid:71338a17-78d9-44a9-96c5-15b4b841f6b8

Publication date
2023

Document Version
Final published version

Citation (APA)
Taumaturgo Mororo, L. A. (2023). *Extending the Thick Level Set Approach: Plasticity, Parallel Computing and Cohesive Cracks*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:71338a17-78d9-44a9-96c5-15b4b841f6b8

Extending the Thick Level Set Approach

Plasticity, Parallel Computing and Cohesive Cracks

EXTENDING THE THICK LEVEL SET APPROACH

PLASTICITY, PARALLEL COMPUTING AND COHESIVE CRACKS

Dissertation

for the purpose of obtaining the degree of doctor
at Deflt University of Technology
by the authority of the Rector Magnificus prof.dr.ir. T.H.J.J. van der Hagen
chair of the Board for Doctorates
to be defended publicly on
Wednesday 5 July 2023 at 10:00 o'clock

by

Luiz Antonio TAUMATURGO MORORÓ

Mestre em Engenharia Civil: Estruturas e Construção Civil
Universidade Federal do Ceará, Brazil
born in Fortaleza, Brazil

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Dr.ir. F.P. van der Meer	Delft University of Technology, promotor
Prof.dr.ir. L.J. Sluys	Delft University of Technology, promotor
Independent members:	
Prof.dr. F. Larsson	Chalmers University of Technology, Sweden
Prof.dr. N. Moës	Ecole Centrale de Nantes, France
Prof.dr.ir. H. Askes	University of Twente, Netherlands
Prof.dr.ir. M.A.N. Hendriks	Delft University of Technology
Dr.rer.nat. M. Möller	Delft University of Technology
Prof.dr.ir. J.G. Rots	Delft University of Technology, reserve member

Keywords: Thick Level Set, plasticity, parallel computing, skeleton curve, fracture mechanics
Printed by: Proefschriftspecialist

ISBN 978-94-6366-710-4

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

We cannot solve our problems
with the same thinking we used when we created them.

Albert Einstein

CONTENTS

SUMMARY

Developing accurate and robust numerical approaches that are capable of modeling fracture in solids has been a challenging undertaking in the computational mechanics community for decades. Models based on a continuous formulation or on a discontinuous one have been proposed by numerous authors, expanding upon abilities and disadvantages of these approaches. However, models attempting to bridge these two approaches have been less often encountered in the literature.

Over the last ten years, a new approach for modeling fracture in solids has been developed, coined the Thick Level Set (TLS) method, in which the damage evolution is linked to the movement of a damage front described with the level set method. This model offers an automatic transition from damage to fracture and deals with merging and branching cracks as well as crack initiation in a easy and robust manner. Furthermore, the TLS in its new (second) version, coined the TLSV2, is able to model explicitly the displacement discontinuity at the position of a crack.

These TLS features are very beneficial for the modeling of cusp crack patterns in resin-rich regions of fiber reinforced polymer composites under mode II loading. In this process, plasticity might occur prior to fracture, which begins with a series of inclined cracks that eventually merge to form what is at higher scale of observation understood as a single crack. When the crack reaches one of the boundaries of these resin-rich regions, the localized deformation in these parts is a sliding one, which is expected to be traction-free.

This *in situ* process has been reported as one of the reasons for the differences in terms of fracture energy between mode I and mode II crack growth since forming cusps requires the emergence of more crack surface than forming a single straight crack. Therefore, in order to simulate this fracture process under realistic boundary conditions, a model based on the TLS can be embedded in a 'macroscopic' setup, such as three-point bend end-notched flexure. A monolithic scheme with extreme refinement in a zone of interest is the most straightforward approach; however, for a specimen with realistic dimensions, this can be computationally unfeasible due to the computational resources needed to solve the large systems of equations involved in such problem. An ability to comprehensively model this microscopic process could help to achieve a better understanding of the mechanism behind the observed dependence of the fracture energy on the mode of fracture, which may in turn improve macroscale simulations.

This work focuses on extending the TLS method in order to profit from its full capabilities to deal with simulations of failure in solids under quasi-static loading conditions. For this purpose, several original numerical and theoretical components are proposed for reaching qualitative agreement with experimental observations of cusp formation in polymer matrix. In this context, the primary application of this thesis relies on the experimental observations at the microscopic level of such process. However, it is worth mentioning that the numerical tools developed in this thesis are not limited to the prob-

lem of cusps; in fact, they can be either used or easily extended to simulate other problems, for instance crack growth through the microstructure of cementitious materials with different aggregates.

First, the TLS is combined with plasticity in order to deal with ductile fracture since polymers may behave plastically prior to failure, particularly when loaded in shear. To accommodate for plasticity, several changes to the TLS framework are introduced. A strength-based criterion for initiation of damage based on the ultimate yield surface of such plasticity model is proposed. A mapping operator for transferring plastic history is included if the integration scheme in a finite element changes due to the evolution of the level set field. Furthermore, a new loading scheme is devised in order to take into account permanent strain.

Next, a generalized framework for the TLSV2 is introduced. The TLSV2 couples continuous and discontinuous approaches within a single framework, where the continuum part allows for handling crack initiation, branching and merging, whereas the discontinuous part brings the capability to handle discrete cracks with large crack opening or sliding without heavily distorted elements, as well as the possibility to model stiffness recovery upon contact.

Two major issues with the TLSV2 method that have not been dealt with since its inception are addressed in this thesis, and solutions are proposed. Firstly, the method depends on identifying the location of the skeleton curve of the level set field, on which the discontinuity in the displacement field is evaluated. The problem of locating the skeleton curve can be a complicated task, even more so because topological events may emerge as the analysis progresses, such as crack branching. The skeleton curve is determined through a combination of ball-shrinking and graph-based algorithms and then mapped onto the finite element mesh. Secondly, the cohesive forces and displacement discontinuity of the TLSV2 are modeled using the phantom node method. Furthermore, a new approach to compute the non-local crack driving force is introduced, and model calibration is discussed. The degree of stiffness recovery under compression that is still needed for the continuum part is investigated.

The TLS can be a computationally demanding approach. Therefore, a domain decomposition strategy is introduced in order to obtain a parallel implementation of the TLS method. To handle the numerical components specific to the TLS analysis steps involving level set update, equilibrium solution, and damage front advance, a parallel strategy is introduced for each of them. The most demanding task in terms of computational cost, i.e., solving the linearized system of equations from the equilibrium problem, is performed with a parallel iterative method profiting from the adopted domain decomposition method. A communication strategy is provided to deal with enriched nodes and new nodes necessary for the phantom node method belonging to shared regions of subdomains. Collective communication strategies are also proposed to deal with operations related to the level set update, damage front advance, and skeleton curve.

Numerical experiments demonstrate the accuracy and efficiency of the proposed framework in handling simulations of failure analysis with complex crack patterns in a sequential and parallel context.

SAMENVATTING

Het ontwikkelen van nauwkeurige en robuuste numerieke methodes voor het modelleren van scheurvorming is al decennia lang een uitdaging in de numerieke mechanica. Modellen gebaseerd op continue of discontinue formulering zijn voorgesteld door verschillende onderzoekers, onder vermelding van de voor- en nadelen van deze benaderingen. Modellen die beide benaderingen proberen te verbinden zijn minder wijdverbreid in de wetenschappelijke literatuur.

In de afgelopen tien jaar is een nieuwe benadering voor het modelleren van scheurvorming ontwikkeld, de *Thick Level Set* (TLS) methode, waarin de ontwikkeling van schade gekoppeld is aan beweging van een schade-front dat beschreven wordt met de *level set* methode. Deze benadering biedt een automatische transitie van schade naar breuk en kan het vertakken en samengroeien van scheuren op een eenvoudige en robuuste manier beschrijven, evenals de initiatie van scheuren. De tweede versie van de methode (de TLSV2) biedt de mogelijkheid de discontinuïteit in verplaatsingen ter plaatse van de scheur expliciet te modelleren.

Deze eigenschappen van de TLS zijn gunstig voor het modelleren van het patroon van *cusp*-vorming zoals dat voorkomt in de harsrijke zone in vezelversterkte kunststoffen onder *mode II* belasting. In dit proces kan plasticiteit aan scheurvorming voorafgaan, gevolgd door het ontstaan van een reeks schuine scheuren die uiteindelijk samengroeien tot wat op hogere schaal van observatie een enkele scheur is. Wanneer de scheur een van de randen van de harsrijke zone bereikt vindt gelokaliseerde vervorming plaats in de vorm van afschuiving zonder overbrenging van spanningen.

Dit *in situ* proces is genoemd als een van de oorzaken voor het verschil in breukenergie tussen mode I en mode II scheurgroei, omdat de cusp-vorming met een groter scheuroppervlak gepaard gaat dan een enkele rechte scheur. Om dit proces met realistische randvoorwaarden te simuleren, kan een model gebaseerd op de TLS ingebed worden in een macroscopisch model, bijvoorbeeld voor driepuntsbuiging met eindscheur. Een monolithisch schema met extreme verfijning in een specifieke zone is de eenvoudigste benadering, maar voor een proefstuk met realistische afmetingen is dit niet haalbaar vanwege de rekentijden die vereist zijn om de grote systemen van vergelijkingen van een dergelijk model op te lossen. Het microscopische proces volledig te modelleren zou kunnen helpen om de mechanismes achter de geobserveerde afhankelijkheid van breukenergie van het type scheurgroei te doorgronden, wat vervolgens macroscopische berekeningen nauwkeuriger zou kunnen maken.

Dit onderzoek is gericht op het uitbreiden van de TLS om de mogelijkheden van de methode volledig te benutten voor simulaties van scheurgroei onder quasi-statische belastingen. Om dit te bereiken worden verschillende oorspronkelijke numerieke en theoretische elementen voorgesteld met als doel kwalitatieve overeenkomst met experimentele waarnemingen van cusp-vorming. Hoewel de primaire motivatie voor de voorgestelde methodes ligt in microscopische experimentele waarnemingen van dit proces,

zijn de methodes niet uitsluitend geschikt voor het simuleren van cusp-vorming; ze kunnen ook gebruikt worden voor simulatie van andere problemen, bijvoorbeeld scheurgroei door de microstructuur van materialen gebaseerd op cement met verschillende toeslagmaterialen.

Ten eerste is de TLS gecombineerd met plasticiteit om ductiele breuk te kunnen beschrijven, omdat kunststoffen plastisch vervormen voordat ze bezwijken, zeker wanneer ze in afschuiving belast worden. Om plasticiteit te kunnen modelleren zijn verschillende wijzigingen toegepast in het TLS raamwerk. Een criterium voor initiatie gebaseerd op het vloeioppervlak van een plasticiteitsmodel is voorgesteld. Een operatie is toegevoegd voor het overdragen van geschiedenis wanneer het integratieschema in een element verandert. Bovendien is een nieuw belastingschema opgesteld om permanente vervorming in rekening te kunnen brengen.

Vervolgens is een algemeen raamwerk voor de TLSV2 geïntroduceerd. De TLSV2 koppelt continue en discontinue benaderingen in één raamwerk, waarbij het continue deel initiatie, vertakking en het samengroeien van scheuren verzorgt, terwijl het discontinue deel de mogelijkheid biedt om discrete scheuren met grote opening of afschuiving te modelleren zonder extreem vervormde elementen, met daarbij de mogelijkheid om stijfheid te herwinnen bij contact.

Twee belangrijke punten waar de TLSV2 eerder nog niet compleet was zijn in dit proefschrift aangepakt. Ten eerste, vereist de methode het bepalen van de locatie van de skelet-curve van het level set veld, waar de discontinuïteit in de verplaatsing geëvalueerd dient te worden. Deze locatiebepaling is een uitdaging, vooral omdat de topologie van de skelet-curve tijdens de analyse kan veranderen, bijvoorbeeld ten gevolge van vertakking van de scheur. De skelet-curve wordt vastgesteld door een combinatie van bal-krimp- en grafenalgoritmes en vervolgens op het elementen-net geprojecteerd. Ten tweede, worden de discontinuïteit in de verplaatsingen en de cohesieve krachten gemodelleerd met de *phantom node* methode. Daarbij is een nieuwe benadering om de niet-lokale drijvende kracht achter scheurgroei te bepalen geïntroduceerd en calibratie van het model verkend. De mate waarin het herwinnen van stijfheid onder druk nog toegepast dient te worden in het continuüm domein is onderzocht.

De TLS kan veel rekenkracht vereisen. Daarom is een strategie van domein-decompositie geïntroduceerd die een parallelle implementatie van de TLS mogelijk maakt. Een parallelle strategie is geïntroduceerd voor de TLS-specifieke taken van update van het level set veld, berekening van evenwicht en ontwikkeling van het schadefront. De meest veeleisende taak, het oplossen van het gelineariseerde systeem van vergelijkingen voor de evenwichtsberekening is uitgevoerd met een parallelle iteratieve methode gebruik makend van de domein-decompositie. Strategieën voor communicatie rondom verrijkte knopen en toegevoegde knopen, alsmede voor operaties betreffende de level-set update en bepaling van de skelet-curve zijn ontwikkeld.

Numerieke experimenten tonen de nauwkeurigheid en efficiëntie van het voorgestelde raamwerk voor het simuleren van bezwijken met complexe scheurpatronen in sequentiële en parallelle context.

1

INTRODUCTION

1.1. BACKGROUND

The development of numerical tools capable of capturing complex mechanisms present in fracture in solids is still one of the most lively fields in computational mechanics, in spite of its vastness of publication in the literature and long history. Modeling fracture can be achieved in different manners, which can be categorized into two approaches [1]: discontinuous and continuous. In the former approach, cracks are explicitly inserted into the solution basis by adding a discontinuity in the displacement field. Optionally, cohesive forces are applied on the crack surface in order to control the amount of energy that is dissipated. In the latter approach, cracks are modeled by an internal damage variable that degrades the stiffness of the material as a consequence of accumulative micro-crack emergence.

Advantages and drawbacks of both families of models have been extensively reported in the literature. Cohesive elements, which fall into the category of discontinuous approaches, have been widely used to model fracture in solids where a discontinuity is considered between finite element edges, and cohesive tractions are defined connecting the adjacent elements. Complex behavior at the crack surface, such as frictional contact and mixed mode fracture, can readily be modeled. These models, however, tend to suffer from mesh bias due to the fact that discontinuity (and consequently the crack path) has to follow the finite element mesh orientation. Numerical methods have been proposed in order to overcome this mesh dependency by allowing the crack to propagate across finite elements, such as XFEM [2, 3] and the phantom node method [4, 5]. Although these methods allow for representing arbitrarily positioned discontinuities, proper administration of the evolving discontinuity in presence of branching and merging is challenging, as well as for general 3D cases.

At the other end of the spectrum, continuum damage models can readily handle damage onset and complex damage distributions, even when multiple merging events are taking place as the simulation progresses. However, these models are well known for experiencing spurious localization in the strain field resulting in pathological mesh

dependency. Several models have been proposed to avoid this problem, well-known examples being the non-local integral models [6, 7] and gradient damage models [8, 9]. All these models rely on the concept of introducing, implicitly or not, an intrinsic material length scale in order to obtain a non-local description. In line with these approaches, two new frameworks have been proposed recently, the phase-field method [10–12] and the Thick Level Set (TLS) method [13–15]. Both methods introduce a superposed field to describe the material degradation, giving rise to a transition region from sane to fully degraded material.

The TLS method is the approach of choice for modeling fracture in this thesis. The TLS has been proposed as an approach that couples both fracture and damage in a single regularized framework. The location of fronts of damaged zones are implicitly represented as the zero level set of an auxiliary field, the level set field. All the topological events with respect to damage fronts are handled with the level set technology [16, 17]. Damage is constrained to follow a given profile within a thick band of material behind the damage front, the transition zone with fixed width, where the damage variable depends directly on the level set values and gradually increases until a zero-stiffness state is obtained, i.e., when macro-cracks emerge. The introduction of a material length scale characterized by the width of the transition zone introduces a non-local feature in the formulation that prevents spurious localization in the strain field. Furthermore, damage growth is dictated by non-local configurational forces that are obtained by integrating local values of energy release rate over damaged regions. The non-local configurational force can be related to the energy dissipation upon crack growth and, as such, links the damage formulation to fracture mechanics. Additionally, the damage evolution is performed separately from the computation of the displacement field, which gives the TLS the natural ability to avoid convergence issues in challenging crack growth scenarios.

Since its first inspection by Moës et al. [13] and further studies by Stolz and Moës [14], the TLS method has been a research topic of several works. A brief tour of the works based on the method is given as follows. Bernard et al. [15] introduced improvements to the first TLS implementation for elastic materials in two-dimensional settings under quasi-static loading condition, especially the way how to compute the non-local configurational force, and how to achieve a strain discontinuity across a fully degraded region through an enrichment scheme. These two enhancements and the model itself by Bernard et al. [15] were used as basis for the three-dimensional implementation of the TLS by Salzman et al. [18], where a new way to construct the cracks faces are proposed, called double cut algorithm by the authors. Moreau et al. [19] proposed the TLS approach in the dynamical context of materials subjected to impact loading cases, in which different time discretization schemes were applied for the equilibrium equations and level-set-related update. Latifi et al. [20] proposed a model, called the Interface Thick Level Set (ITLS) model, in which the TLS is combined with interface elements for modeling quasi-static delamination growth in composites. Latifi et al. [21] and Voormeeren et al. [22] extended the ITLS approach to simulate fatigue crack growth. Contrary to the ITLS, where the crack path has to be known *a priori*, Niessen [23] proposed a TLS-based model to simulate a standard compact tension test in a fatigue context without defining interface elements. In other studies, the TLS method has been compared with alternative approaches, such as phase-field [24] and cohesive zone models [25], where the

numerical investigations carried out in both studies indicated similar results in terms of load-displacement curves and damage distribution among the approaches.

One application where the robustness of the TLS is particularly relevant is the simulation of cusp formation, which involves many merging cracks. Cusp formation is a process that accompanies mode II delamination crack growth, and it is understood to be one of the causes of the difference in fracture energy between mode I and mode II crack growth [26–28]. This process starts with an array of inclined cracks which are perpendicular to the direction of maximum principal stress (see Fig. 1.1). As these cracks evolve, S-shaped cracks are formed, which eventually merge, leading to a single crack on a higher level of observation. Appropriate theoretical and computational tools that can predict this process on the microscale may lead to a better understanding of the mechanisms behind the observed variability in fracture energy, which may in turn allow for devising physics-based models for macroscopic crack growth [26, 27].

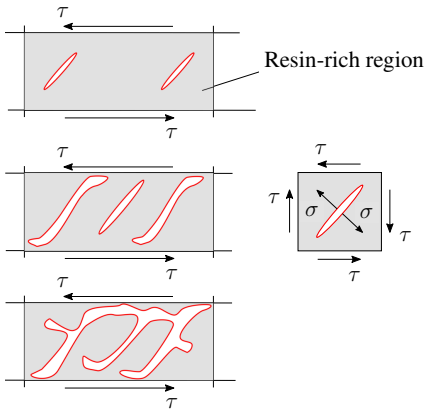


Figure 1.1: Cusp formation stages in a resin-rich region subject to shear (τ). Cracks are perpendicular to the direction of maximum principal stress (σ).

One noteworthy fact in this direction is that the TLS method, in the version by Van der Meer and Sluys [29], has already been used to model the cusp crack pattern in elastic sandwich-like specimens. Based on their investigations, the authors improved the representation of free-sliding deformations in mode II failure condition by introducing a special constitutive law that only takes into account stiffness recovery on the strain component normal to the plane that defines the interface, and made the resistance parameter against damage growth a function of the size of damaged zones in order to capture different stress levels for damage onset and propagation.

More recently, a new (second) version of the TLS, designated as the TLSV2, has been proposed as a new concept for coupling continuum damage modeling and cohesive crack modeling, combining concepts from continuous and discontinuous approaches, for failure analysis in solids [1]. The main objective of this new framework is to profit from both modeling approaches, the continuum part allows for handling crack initiation, branching and merging, whereas the cohesive part brings the capability to handle discrete cracks with large crack opening or sliding without heavily distorted elements,

and with the possibility to model stiffness recovery and friction upon contact, i.e., the main issues found in simulations presenting cusp crack patterns [29]. The method depends on identifying the location of a so-called skeleton curve of the level set field, i.e., the line along which the slope of the level set field is discontinuous, which defines the middle of a damaged zone and on which the discontinuity in the displacement field should be inserted. However, the problem of locating the skeleton curve can be a complicated task, especially when it has to be sufficiently smooth for defining the discontinuity along which sliding can be modeled. The complexity increases further when topological events, such as crack branching takes place. Despite its promising advantages and benefits, the TLSV2 has only been applied to test cases with straight skeleton curves at *a priori* known locations [1].

Coupling the continuous and discontinuous descriptions within a single framework is not a novelty. As examples, we refer to [30–33], in which the cohesive cracks are introduced as soon as the damage variable exceeds a damage threshold. Unlike these approaches, the TLSV2 permits a concurrent evolution of quantities (e.g., damage variables) related to the bulk and cohesive parts by means of a unique level set field in the whole domain [1].

1.2. SCOPE AND OUTLINE

The objective of this thesis is to develop numerical tools based on the TLS approach keeping the cusp formation process in resin-rich regions of polymer matrix of composite materials subject to mode II delamination as benchmark application. The thesis is restricted to 2D analysis under quasi-static loading, and we will consider crack growth through elastic and elasto-plastic materials. Plasticity is combined with the TLS solution scheme since polymers may behave plastically prior to failure when loaded in shear. The TLSV2 capabilities are extended in order to deal with free-form skeleton curves. Parallel computing is employed in order to accelerate numerical simulations since the TLS may be a time demanding approach.

The core numerical tools developed in this work are detailed in the following four chapters of the thesis, which are composed of three published in peer reviewed journal papers with minor changes in their original introduction and conclusion sections in Chapters 2 to 4, respectively, and an additional investigation that integrates the work from these preceding chapters in Chapter 5. What follows next is a brief description on topics treated in all the remaining chapters of the thesis.

In Chapter 2, the TLS method is combined with plasticity. To accommodate plasticity, several changes to the TLS framework are introduced. A new loading scheme is devised that does not rely on secant unloading. Numerical experiments demonstrate the accuracy and effectiveness of the proposed model to handle simulation of crack growth in a medium with hardening plasticity.

In Chapter 3, a domain decomposition strategy to obtain a parallel implementation of the TLS method is introduced. It describes how to handle the numerical features specific to the TLS analysis steps involving level set update, equilibrium solution and damage front advance. For each of these tasks an appropriate parallel computing strategy is proposed. Numerical experiments demonstrate the accuracy and efficiency of the proposed framework to handle parallel computing with the TLS method.

In Chapter 4, the TLSV2 approach is developed beyond the proof of concept from the paper in which it was introduced. Challenges are addressed on the development of algorithms to track the skeleton curve, and on how to model the displacement discontinuity over a crack based on the skeleton location. Compared to its predecessor, a slightly different algorithm scheme is devised for the TLSV2. Numerical experiments demonstrate the accuracy and ability of the proposed model to handle simulation of failure analysis presenting non-trivial topological crack patterns with actual displacement discontinuity at the position of the crack.

In Chapter 5, the innovations from Chapters 2 to 4 are combined into a single framework. Extra features are added to the original parallel implementation in order to accommodate the specific operations related to the TLSV2. Despite adding new tasks into the parallel implementation in Chapter 3, numerical examples show that the additional tasks associated with the TLSV2 do not substantially change the total runtime compared to the parallel version designed for its predecessor model.

Finally, in Chapter 6, the main scientific conclusions are presented, followed by recommendations for future research.

2

COMBINING THE THICK LEVEL SET APPROACH WITH PLASTICITY

2.1. INTRODUCTION

The Thick Level Set (TLS) method to model damage in solids was originally proposed by Moës et al. [13]. In the TLS, the location of the front of a damaged zone is implicitly represented as the zero level set of an auxiliary field and its evolution is handled with the level set method [16, 17]. Unlike conventional continuum damage models, in which the damage variable is a direct function of the local strain field, the TLS considers a band of damage with a predefined characteristic length where the damage variable depends on the level set value whose evolution is dictated by the non-local strain field. Macro-cracks, i.e., regions with damage equal to one, appear as a consequence of the front evolution. As the damage evolution is separated from computation of displacement [15, 29], the TLS is a robust method which can handle multiple branching and merging cracks without convergence problems.

Since the first paper on the TLS [13], improvements on its implementation have been proposed [15, 35], and more recently an extension was presented, where the main idea is to couple cohesive zone models with the TLS to capture crack opening [1]. In other publications, the TLS has been compared with alternative approaches, such as phase-field [24] and cohesive zone [36] models.

As already mentioned in Chapter 1, modeling numerically the process of cusp formation in polymer matrices is not a straightforward task, mainly because of complex crack patterns may arise during the analysis. One more key aspect that has to be taken into account in this mechanism is plasticity. When loaded in shear, polymers behave plastically prior to failure. Therefore, plasticity cannot be ignored in simulation of cusp formation. However, adding plasticity to the TLS can be a complicated issue, because the current solution procedure [15, 29] depends on the assumption of secant unloading behavior.

Apart from minor changes to its introduction and conclusion sections, this chapter was integrally extract from L. A. T. Mororó and F. P. van der Meer. *European Journal of Mechanics - A/Solids* 79 (2020) [34].

In this chapter, we seek to simulate the cusp formation process more realistically. For this purpose, the TLS method in the version by Van der Meer and Sluys [29] is extended with a plasticity formulation. To represent the plastic behavior of polymers, the pressure-dependent plasticity model by Melro et al. [37] is implemented in the TLS framework. A plasticity-related criterion for damage initiation is introduced, and a new loading scheme is devised that does not rely on secant unloading.

The chapter is structured as follows. Section 2.2 is devoted to the formulation of the proposed model, presenting the plasticity model, recalling some fundamentals of the TLS damage model and detailing the main features added to the framework from Van der Meer and Sluys [29]. Several numerical examples including plasticity are presented in Section 2.3 and used to assess the accuracy of the proposed model to deal with cusp formation. Finally, conclusions are presented in Section 2.4.

2.2. MODEL FORMULATION

2.2.1. SEPARATION BETWEEN DAMAGE AND PLASTICITY

In this section, the main features of the proposed model are outlined. The quasi-static problems that are assessed in this chapter are based on the framework of small displacements and additive decomposition of the total strain $\boldsymbol{\varepsilon}$ into an elastic (or reversible) part $\boldsymbol{\varepsilon}^{\mathrm{e}}$ and a plastic (or permanent) part $\boldsymbol{\varepsilon}^{\mathrm{p}}$:

$$\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^{\mathrm{e}} + \boldsymbol{\varepsilon}^{\mathrm{p}}. \tag{2.1}$$

The equilibrium equation and the relation between the total strain $\boldsymbol{\varepsilon}$ and the displacement field $\mathbf{u}$ in a body Ω without body force read, respectively:

$$\nabla \cdot \boldsymbol{\sigma} = 0 \quad \text{and} \quad \boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \mathbf{u} + \nabla \mathbf{u}^{\mathrm{T}}\right), \tag{2.2}$$

in which $\boldsymbol{\sigma}$ is the stress tensor.

The starting point of the proposed model is a separation between damage and plasticity. The free energy is defined under the hypothesis of decoupling between elasticity-damage and plasticity, and it is assumed that plasticity only evolves in the intact material and not in regions where the damage is activated. Thus, the specific free energy ψ is assumed to be split up into elastic-damage ψ^{ed} and plastic ψ^{p} contributions according to:

$$\psi(\boldsymbol{\varepsilon}^{\mathrm{e}}, d, \varepsilon^{\mathrm{p}}_{\mathrm{eq}}) = \psi^{\mathrm{ed}}(\boldsymbol{\varepsilon}^{\mathrm{e}}, d) + \psi^{\mathrm{p}}(\varepsilon^{\mathrm{p}}_{\mathrm{eq}}), \tag{2.3}$$

where $\varepsilon^{\mathrm{p}}_{\mathrm{eq}}$ and d are the internal variables, signifying the equivalent plastic strain and the damage, respectively.

For instance, assuming an elastic-damage potential that accounts for isotropic stiffness degradation given by:

$$\psi^{\mathrm{ed}}(\boldsymbol{\varepsilon}^{\mathrm{e}}, d) = \frac{1}{2}(1-d)\,\boldsymbol{\varepsilon}^{\mathrm{e}} : \mathbf{D}^{\mathrm{e}} : \boldsymbol{\varepsilon}^{\mathrm{e}}, \tag{2.4}$$

where $\mathbf{D}^{\mathrm{e}}$ is the elasticity tensor from Hooke's law. The stress-strain relation is obtained by differentiating the potential as:

$$\boldsymbol{\sigma} = \frac{\partial \psi^{\mathrm{ed}}}{\partial \boldsymbol{\varepsilon}^{\mathrm{e}}} = (1-d)\,\mathbf{D}^{\mathrm{e}} : \boldsymbol{\varepsilon}^{\mathrm{e}}. \tag{2.5}$$

The local energy release rate Y is defined as:

$$Y = -\frac{\partial \psi^{\mathrm{ed}}}{\partial d} = \frac{1}{2}\,\boldsymbol{\varepsilon}^{\mathrm{e}} : \mathbf{D}^{\mathrm{e}} : \boldsymbol{\varepsilon}^{\mathrm{e}}. \tag{2.6}$$

Along with the loading/unloading conditions in Kuhn-Tucker form for time-independent models, Y is used to described the the damage evolution:

$$(Y - Y_c) \leq 0, \quad \dot{d} \geq 0, \quad \dot{d}(Y - Y_c) = 0 \tag{2.7}$$

in which Y_c is the material resistance to damage growth.

Using the effective stress concept, Eq. (2.5) can alternatively be expressed as:

$$\boldsymbol{\sigma} = (1-d)\,\hat{\boldsymbol{\sigma}}, \tag{2.8}$$

where $\hat{\boldsymbol{\sigma}}$ is the effective stress defined as:

$$\hat{\boldsymbol{\sigma}} = \mathbf{D}^{\mathrm{e}} : \boldsymbol{\varepsilon}^{\mathrm{e}} = \mathbf{D}^{\mathrm{e}} : \left(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^{\mathrm{p}}\right). \tag{2.9}$$

In order to guarantee the admissibility of stresses, the yield criterion $f(\hat{\boldsymbol{\sigma}}, \varepsilon^{\mathrm{p}}_{\mathrm{eq}}) \leq 0$ must be satisfied. The plastic strain rate is written as the product of the plastic multiplier $\dot{\gamma}$ and the direction of plastic flow $\mathbf{n}$ [38]:

$$\dot{\boldsymbol{\varepsilon}}^{\mathrm{p}} = \dot{\gamma}\,\mathbf{n}. \tag{2.10}$$

The evolution of plastic multiplier $\dot{\gamma}$ is such that the Kuhn-Tucker conditions are satisfied:

$$f \leq 0, \quad \dot{\gamma} \geq 0, \quad \dot{\gamma} f = 0. \tag{2.11}$$

The plasticity model is completed with an evolution law for the equivalent plastic strain, which is defined as a function of the equivalent plastic strain rate $\dot{\varepsilon}^{\mathrm{p}}_{\mathrm{eq}}$.

By design of the TLS, uncoupling damage from plasticity in terms of evolution of the internal variables can be achieved in a straightforward manner, since the level set field ϕ separates the domain Ω into an undamaged zone and a damaged one. In this case, the evolution of plasticity can only occur in zones where the level set function $\phi \leq 0$, whereas the TLS handles the damage evolution law in the region with $\phi > 0$ by taking the permanent strain contribution into account via Eq. (2.9). Therefore, the damage front decides where and when the plasticity is evaluated.

2.2.2. PLASTICITY MODEL

In this section, the equations presented in Section 2.2.1 for a general plasticity model are particularized to the pressure-dependent plasticity model for polymers by Melro et al. [37] as adapted by Van der Meer [39]. The backward Euler scheme is considered to discretize all rate quantities for plasticity. Because plasticity is only updated where $d = 0$ and $\boldsymbol{\sigma} = \hat{\boldsymbol{\sigma}}$, the hat symbol on the effective stress tensor is dropped.

A paraboloidal yield surface is considered:

$$f(\boldsymbol{\sigma}, \varepsilon^{\mathrm{p}}_{\mathrm{eq}}) = 6J_2 + 2I_1(\sigma_c - \sigma_t) - 2\sigma_c\sigma_t \tag{2.12}$$

with

$$\sigma_c = \sigma_c(\varepsilon^{\mathrm{p}}_{\mathrm{eq}}) \quad \text{and} \quad \sigma_t = \sigma_t(\varepsilon^{\mathrm{p}}_{\mathrm{eq}}), \tag{2.13}$$

where J_2 is the second invariant of the deviatoric stress tensor, I_1 is the first invariant of the stress tensor, and σ_c and σ_t are the uniaxial compressive and tensile yield stresses, respectively. The pressure dependency comes from the term $2I_1(\sigma_c - \sigma_t)$. For the case of $\sigma_c = \sigma_t$, the yield surface is equivalent to the classic Von Mises criterion. Both σ_c and σ_t are defined as a function of $\varepsilon^{\mathrm{p}}_{\mathrm{eq}}$ to match measured hardening curves.

If the material is loaded at the yield stress, plastic flow takes place. Using a non-associative flow rule, the plastic strain increment (cf. Eq. (2.10)) is given by:

$$\Delta\boldsymbol{\varepsilon}^{\mathrm{p}} = \Delta\gamma\left(3\mathbf{S} + \frac{2}{9}\alpha I_1\mathbf{I}\right), \tag{2.14}$$

where $\mathbf{S}$ is the deviatoric stress tensor, $\mathbf{I}$ is the identity matrix, and α is the parameter that controls the plastic volumetric flow and depends on the plastic Poisson's ratio ν_p:

$$\alpha = \frac{9}{2}\frac{1-2\nu_p}{1+\nu_p}. \tag{2.15}$$

The increment of the equivalent plastic strain $\varepsilon^{\mathrm{p}}_{\mathrm{eq}}$ is defined as:

$$\Delta\varepsilon^{\mathrm{p}}_{\mathrm{eq}} = \sqrt{k\Delta\boldsymbol{\varepsilon}^{\mathrm{p}} : \Delta\boldsymbol{\varepsilon}^{\mathrm{p}}}, \tag{2.16}$$

in which $k = 1/(1+2\nu_p^2)$.

In order to check for admissibility of stress state and determine the increment of the plastic multiplier $\Delta\gamma$ such that the constraints in Eq. (2.11) are satisfied, an iterative elastic predictor/return mapping algorithm is used. Details on the return mapping algorithm and the consistent tangent matrix can be found in [39].

2.2.3. THICK LEVEL SET METHOD

In the TLS, the front of one or more damaged zones is implicitly represented as the iso-contour (or level set) of an auxiliary field, and its evolution is accomplished by the level set method [16, 17]. The advantage of the level set method is that one can deal with geometric features (e.g., merging and branching) involving surfaces or curves on a discretized domain, without having to explicitly mesh boundaries of these objects. The location of the front Γ_0 is tracked as the zero level set (or the iso-zero) of a single auxiliary field $\phi(\mathbf{x})$ (see Fig. 2.1).

If ϕ is a smooth well-behaved function, the definition of ϕ on a discretized domain at a given point $\mathbf{x}$ is determined by interpolating the values of ϕ from nodes to $\mathbf{x}$. A convenient choice for definition of ϕ is the signed distance function [16, 17], which is mathematically equivalent to:

$$|\nabla\phi| = 1 \quad \text{on} \quad \Omega. \tag{2.17}$$

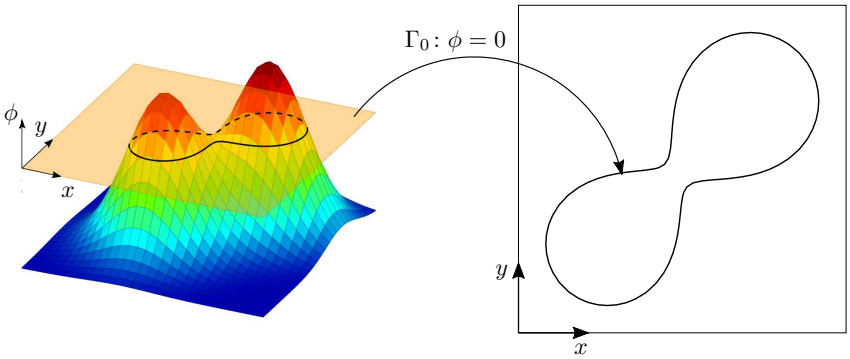


Figure 2.1: The level set method: the front Γ_0 is implicitly located as the zero level set of ϕ.

By definition, this choice for ϕ guarantees that the absolute value of ϕ at a given point is the shortest distance to the front from that point.

DAMAGE DEFINITION

As it has been mentioned, the level set $\phi = 0$ separates the domain Ω into an undamaged zone and a damaged one, and ϕ is known at every point. In the TLS, the damage variable d is chosen to depend only on ϕ. As depicted in Fig. 2.2, d is assumed to change from zero to one as ϕ goes from zero to the critical length l_c. Mathematically, the damage variable is expressed by:

$$d(\phi) = \begin{cases} 0, & \phi \leq 0 \\ q(\phi), & 0 < \phi \leq l_c \\ 1, & \phi > l_c \end{cases}, \tag{2.18}$$

where q is a function that has the properties of $q(0) = 0$, $q(l_c) = 1$ and $q' \geq 0$.

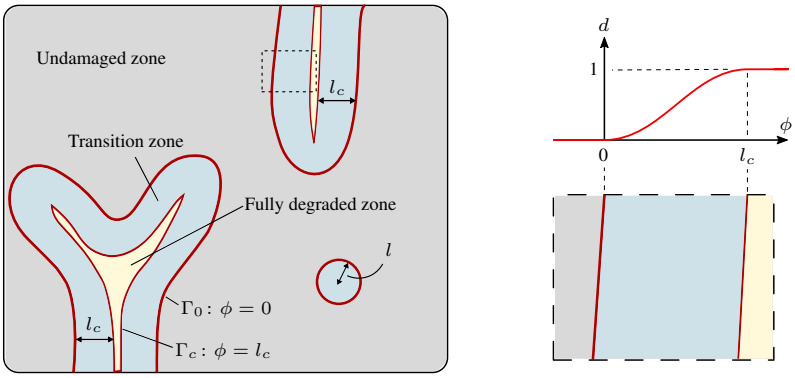


Figure 2.2: The TLS makes use of a single level set function to describe multiple zones. As illustrated on the right, the damage variable d is a function of level set ϕ.

An arc-tangent formula is used for $q(\phi)$:

$$q(\phi) = c_2 \arctan\left(c_1\left(\frac{\phi}{l_c} - c_3\right)\right) + c_4 \tag{2.19}$$

Following Bernard et al. [15] and Van der Meer and Sluys [29], all computations are performed with $c_1 = 10$ and $c_3 = 0.5$, which lead to a point symmetric profile, as schematically represented in Fig. 2.2. The other constants c_2 and c_4 are determined to satisfy the conditions $q(l_c) = 1$ and $q(0) = 0$, respectively:

$$c_2 = (\arctan(c_1(1 - c_3)) - \arctan(-c_1 c_3))^{-1} \tag{2.20}$$

and

$$c_4 = -c_2 \arctan(-c_1 c_3). \tag{2.21}$$

The position of the macro-cracks is located in the zone at a distance larger than l_c behind the front where $d = 1$. This region is easily identified due to the fact that ϕ is a distance function. To accommodate localized deformations, the elements crossed by the iso-l_c of the level set field need some particular enrichment in order to introduce a discontinuity in strain and provide strain-free localization at crack lips, as explained in [15].

FREE ENERGY: ASYMMETRIC BEHAVIOR IN TENSION/COMPRESSION

The free energy expression in Eq. (2.4) leads to material laws that present the same behavior in tension and compression, which can be applicable to failure analysis in tension dominated cases. However, if the damaged zone experiences compression, the energy release rate Y in Eq. (2.6) would still be nonzero, which may result in unphysical compressive cracks. For simulation of cases subjected to shear load conditions, this formulation leads to unrealistic 'X-shaped' cracks as reported by Van der Meer and Sluys [29].

Therefore, the free energy density expression from [15, 29] that accounts for stiffness recovery under compression is used in this thesis:

$$\psi^{\mathrm{ed}}(\boldsymbol{\varepsilon}^{\mathrm{e}}, d) = \mu(1 - \alpha_i d)(\varepsilon_i^{\mathrm{e}})^2 + \frac{\lambda}{2}(1 - \alpha_v d)\mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}})^2, \tag{2.22}$$

where λ and μ are Lamé's elastic constants, $\varepsilon_i^{\mathrm{e}}$ the eigenvalues of the elastic strain tensor, and the α_i and α_v the parameters relate the activation of damage to principal strain $\varepsilon_i^{\mathrm{e}}$ and the volumetric strain $\varepsilon_v = \mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}})$, respectively. Each of these constants assumes the value of 1 or 0 depending on the sign of the associated strain quantity in such way that if one of the principal strains or the volumetric strain becomes negative, the stiffness degradation is canceled for the corresponding term:

$$\alpha_i = \begin{cases} 1, & \varepsilon_i^{\mathrm{e}} > 0 \\ 0, & \varepsilon_i^{\mathrm{e}} < 0 \end{cases} \quad \text{and} \quad \alpha_v = \begin{cases} 1, & \mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}}) > 0 \\ 0, & \mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}}) < 0 \end{cases}. \tag{2.23}$$

By the definitions in Eqs. (2.5) and (2.6), the stress-strain relation and the driving force for damage growth can be rewritten as follows:

$$\sigma_i = \frac{\partial \psi^{\mathrm{ed}}}{\partial \boldsymbol{\varepsilon}^{\mathrm{e}}} = 2\mu(1-\alpha_i d)\varepsilon_i^{\mathrm{e}} + \lambda(1-\alpha_v d)\mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}}) \tag{2.24}$$

and

$$Y = -\frac{\partial \psi^{\mathrm{ed}}}{\partial d} = -\mu\alpha_i(\varepsilon_i^{\mathrm{e}})^2 - \frac{\lambda}{2}\alpha_v \mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}})^2. \tag{2.25}$$

One can observe that the damage has no effect on the material in compression: the undamaged stiffness is recovered, and Y becomes zero, which means damage does not grow.

NON-LOCAL EVOLUTION LAW

In the TLS approach, the non-locality is evident when the front moves [15]. Because the updating of the signed distance function, all points sharing the same curvilinear coordinate s[1] are affected as the front at $(0, s)$ moves (see Fig. 2.3). Therefore, the amount of energy per unit length that will be dissipated as the front moves a unit distance reads:

$$g(s) = \int_0^l d'(\phi)\, Y(\phi, s)\left(1 - \frac{\phi}{\rho(s)}\right) \mathrm{d}\phi, \tag{2.26}$$

where $d'(\phi) = q'(\phi)$ is the spatial derivative of damage with respect to ϕ, l is the size of the damaged zone $l \in (0, l_c]$ (see Fig. 2.2) and ρ is the curvature of iso-zero.

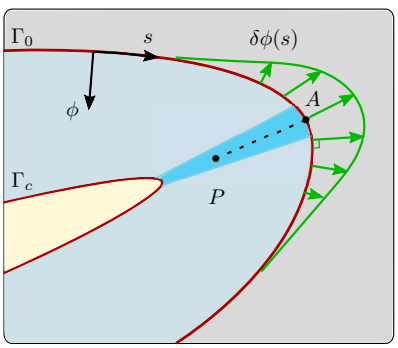


Figure 2.3: Curvilinear coordinate system (ϕ, s). Point P is affected as point A on the front experiences a front advance.

Now, the loading/unloading condition in Eq. (2.7) can be rewritten in terms of non-local energy release rate $g(s)$ and $g_c(s)$. However, for nucleation, that is, in the limit case that l tends to 0, $g(s)$ vanishes. Hence, in order to be able to capture nucleation and smaller damaged zones, the averaged value of Y across the damaged band is introduced as proposed by Bernard et al. [15]. For any position along the front, the averaged value $\bar{Y}$ is defined as the value that satisfies:

[1] A curvilinear system of coordinates (ϕ, s) using the change of variable $\mathrm{d}\Omega = \left(1 - \frac{\phi}{\rho(s)}\right)\mathrm{d}\phi\mathrm{d}s$ is introduced for derivation, following Moës et al. [13]. In the implementation, the curvilinear coordinate system does not need to be defined.

$$\int_0^l d'(\phi)\, Y(\phi, s)\left(1-\frac{\phi}{\rho(s)}\right)\mathrm{d}\phi = \int_0^l d'(\phi)\,\bar{Y}(s)\left(1-\frac{\phi}{\rho(s)}\right)\mathrm{d}\phi. \tag{2.27}$$

The averaged configuration force $\bar{Y}$, which tends to Y as l tends to zero, is thus a weighted average of Y. Finally, the front velocity v_n is defined as a function of $\bar{Y}$ and $\bar{Y}_c$, the weighted average of Y_c (see Section 2.2.3).

In order to compute the averaged configurational force along the front and avoid direct computation of the term depending on the curvature ρ, Eq. (2.27) is discretized as a field on the damaged domain Ω^d with $\bar{Y}$ as unknown. Following Bernard et al. [15], the constraint that $\bar{Y}$ must be constant along the level set gradient, i.e., $\nabla\bar{Y}\cdot\nabla\phi = 0$, is enforced with Lagrange multipliers. A discretized approximation of $\bar{Y}$ is introduced in combination with Galerkin's method leading to the following system of equations:

$$\begin{bmatrix}\mathbf{K} & \mathbf{L}\\ \mathbf{L} & \mathbf{0}\end{bmatrix}\begin{Bmatrix}\bar{\mathbf{Y}}\\ \mathbf{l}\end{Bmatrix} = \begin{Bmatrix}\mathbf{f}^{\mathrm{Y}}\\ \mathbf{0}\end{Bmatrix} \tag{2.28}$$

in which $\bar{\mathbf{Y}}$ and $\mathbf{l}$ are vectors with $\bar{Y}$ and Lagrange multiplier degrees of freedom, respectively. The matrices and the right-hand side vector are defined as:

$$K_{ij} = \int_{\Omega^d} d' N_i N_j + \frac{\kappa h^2}{l_c}\frac{\partial N_i}{\partial x_k}\frac{\partial N_j}{\partial x_k}\mathrm{d}\Omega, \tag{2.29}$$

$$L_{ij} = \int_{\Omega^d} l_c\left(\frac{\partial N_i}{\partial x_k}\frac{\partial \phi}{\partial x_k}\right)\left(\frac{\partial N_j}{\partial x_k}\frac{\partial \phi}{\partial x_k}\right)\mathrm{d}\Omega, \tag{2.30}$$

and

$$f_i^{\mathrm{Y}} = \int_{\Omega^d} N_i d' Y\,\mathrm{d}\Omega, \tag{2.31}$$

where N_i and N_j are the shape functions associated with nodes i and j, κ is a stabilization parameter, h is the characteristic size of the smallest element, and Y is the configurational force which depends on the current elastic strain field evaluated through Eq. (2.25). If the damage resistance Y_c is not constant over the domain, a similar system of equations, with a different right hand side where Y_c is used instead of Y, is solved to compute $\bar{Y}_c$.

FRONT MOVEMENT

To update the damage distribution, the advance of the level set field should be related to the configurational force $\bar{Y}$ and material resistance against damage growth $\bar{Y}_c$. The change in the level set field is related to the normal velocity of the front as:

$$\frac{\partial\phi}{\partial t} + v_n|\nabla\phi| = 0. \tag{2.32}$$

In absence of physical time in quasi-static simulations, the TLS does not directly work with velocities but with a front increment. In this case, $v_n\Delta t$ is regarded as the front increment. Using forward Euler time discretization and the property of $|\nabla\phi| = 1$, the update of the level set field is performed as:

$$\phi \leftarrow \phi + v_n \Delta t, \tag{2.33}$$

where Δt is the time increment size.

In the previous version of the TLS [13, 15, 29], the loading scheme was based on a unit load analysis in each time step, computing a critical load scale factor under the assumption of secant unloading. This loading scheme is not applicable when permanent strain due to plasticity is considered. Therefore, the framework of TLS must be adapted. Here, the following relation for the normal velocity is used, given by [20, 40]:

$$v_n = \frac{1}{\eta} \left\langle \frac{\bar{Y}}{\bar{Y}_c} - 1 \right\rangle_+, \tag{2.34}$$

where η is a parameter that can be interpreted as viscous resistance against crack growth. In the limit of $\eta \to 0$, the Kuhn-Tucker conditions for quasi-static crack growth with $\bar{Y} - \bar{Y}_c \leq 0$ are approached. Brackets are used to denote the positivity condition, which reflects the irreversibility of crack growth.

In order to ensure stability of the explicit level set update, the Courant-Friedrichs-Lewy condition is applied [17]:

$$\Delta t < \frac{h}{\max\{v_n\}}, \tag{2.35}$$

where h is the characteristic size of the smallest element, and $\max\{v_n\}$ is the largest value of v_n over the entire domain. Here, this conditions is rewritten in a more conservative form according to [17, 40]:

$$\Delta t = \min\left\{\Delta t^0, \alpha_n \frac{h}{\max\{v_n\}}\right\}, \tag{2.36}$$

in which Δt^0 is the default and maximum time increment, and α_n is a constant defined as $0 < \alpha_n < 1$.

The level set update with Eq. (2.33) requires the velocity to be known throughout the domain. However, Eq. (2.34) is only calculated along the front. The velocity computed at the nodes of elements that contain the front is propagated through the domain by solving:

$$\nabla\phi \cdot \nabla v_n = 0. \tag{2.37}$$

This is done with a fast marching method [16, 17, 40]. When the level set field ϕ^n in the previous time step n, which is a signed distance function, moves $v_n \Delta t$ units forward in the normal direction, the updated level set field ϕ^{n+1} obtained by Eq. (2.33) remains a signed distance function. This arises from the fact that the gradient of Eq. (2.33) leads to $\nabla\phi^{n+1} = \nabla\phi^n$, since $v_n \Delta t$ is spatially constant in ϕ direction, i.e., $\nabla_\phi(v_n \Delta t) = 0$. Thus, if ϕ^n is initially a signed distance function, it will stay a signed distance function [16, 17].

However, in the discretized model, the level set field may drift away from being an accurate representation of signed distance. Thus, another fast marching method needs to be applied periodically in order to keep ϕ as signed distance function [40]. Since it is a relatively cheap procedure, this reinitialization is performed every time step.

INITIATION

To deal with damage activation for the proposed model, the criterion $Y \geq Y_c$ must be satisfied at undamaged point. Note that this criterion is purely local, which is consistent with Eq. (2.27), since $\bar{Y}$ and $\bar{Y}_c$ tend to Y and Y_c as l tends to zero. When this criterion is met at any point, a circle with radius $\phi_0 < l_c$ is inserted around that point and the signed distance function is reinitialized accordingly. The size of this nucleus from then on increases according to the same framework as introduced in Sections 2.2.3 and 3.2.3.

Following the proposed model in [29], the resistance Y_c is made into a function of the size of the damaged zone in order to handle initiation and propagation with separate material parameters. In this approach, Y_c changes from an initial strength-based value for initiation Y_c^0 to a fracture energy-based value for crack growth Y_c^G, as the size of the damaged zone changes from zero to a circle with radius l_c. The intermediate values of Y_c are interpolated between the two bounds in the space of $\log(Y_c)$:

$$\log(Y_c) = \log(Y_c^0) + \frac{\bar{\phi}}{\bar{\phi}_{\max}} \left(\log(Y_c^G) - \log(Y_c^0) \right), \tag{2.38}$$

where $\bar{\phi}$ is a measure for the size of a damaged zone over a closed damaged subdomain defined as the average of ϕ over the subdomain and $\bar{\phi}_{\max}$ represents the size for which the damaged zone is considered a crack. In Fig. 2.4, three stages are schematically sketched as $\bar{\phi}$ varies from $\bar{\phi} = 0$, for a very small circle with radius ϕ_0, to $\bar{\phi} = \bar{\phi}_{\max}$, for a circle with radius l_c, to $\bar{\phi} = l_c/2$, for a long straight damaged zone with width $2l_c$.

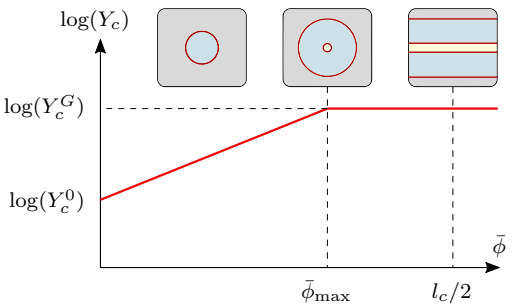


Figure 2.4: Interpolation of Y_c between Y_c^0 and Y_c^G (adapted from [29]).

The averaged value $\bar{\phi}$ is computed in each time step in a similar way as $\bar{Y}$ in Eq. (2.28) by substituting level set values ϕ and unknowns $\bar{\phi}$ for Y and $\bar{Y}$, respectively, and omitting the weight factor d' from left-hand side matrix and right-hand side vector. Variation of $\bar{\phi}$ in the normal direction is again eliminated with Lagrange multipliers, while variation in the curvilinear direction is eliminated by considering a high value for κ [29].

In contrast to what was proposed in [29], in which Y_c^0 was set to a constant value equal to the free energy in the case where the uniaxial stress equals the tensile strength, Y_c^0 is here bounded by the following surface based on Eq. (2.12):

$$f^0 = \frac{3\hat{J}_2}{f_c f_t} + \frac{\hat{I}_1 (f_c - f_t)}{f_c f_t} - 1 = 0, \tag{2.39}$$

with f_c and f_t being the compressive and the tensile strengths which are set equal to the ultimate yield stress values of the material. The two invariants $\hat{J}_2$ and $\hat{I}_1$ are determined using the effective stress. The motivation to use such surface for Y_c^0 comes from the fact that Y_c^0 as a function of a single parameter f_t may prohibit initiation for certain stress states when a plasticity model with perfectly plastic tail is used. For the particular case where a uniaxial tensile strength equals f_t, a constant Y_c^0 would be defined as (see Eq. (2.25)):

$$Y_c^0(f_t) = \frac{f_t^2}{2E}\left(\frac{1+2\nu-2\nu^2}{1+\nu}\right), \tag{2.40}$$

where E is the Young's modulus and ν is the Poisson's ratio. Note that the term depending on ν in the expression above is a correction factor which was missing in the previous TLS model (cf. [29], where $Y_c^0 = f_t^2/2E$). This factor is needed because under uniaxial tension not all α's in Eq. (2.22) evaluate to 1.

Figure 2.5 illustrates the problem of using the constant $Y_c^0(f_t)$ from Eq. (2.40) as initiation criterion in presence of plasticity with a given ultimate yield surface f^0 that bounds the admissible stress states. It can be observed that, if the ultimate uniaxial tensile yield stress is used for f_t in $Y_c^0(f_t)$, no initiation under pure shear is possible. The damage cannot be activated since the plasticity bounds the stress states.

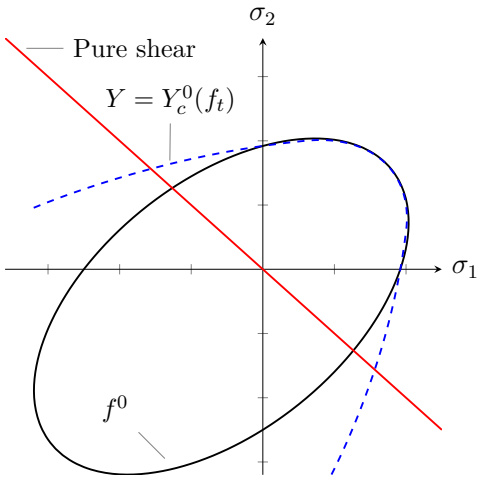


Figure 2.5: Envelopes in principal stress space illustrating the inability of the criterion $Y = Y_c^0(f_t)$, related to the single parameter f_t, to deal with damage initiation under pure shear.

This illustrates that, in presence of plasticity, more control is needed over the initiation envelope. Here, a relation is devised between Y_c^0 and the stress orientation so that the failure envelope coincides with the final yield surface given by f^0. The relation for Y_c^0 is for 2D plane stress state derived as a function of the angle θ, defined in principal stress space as $\theta = \arctan(\sigma_2/\sigma_1)$. The idea is that for a given stress vector $\boldsymbol{\sigma}(\sigma_1,\sigma_2)$, Y_c^0 is given as $Y_c^0(\boldsymbol{\sigma}_c^0)$, with the critical stress vector $\boldsymbol{\sigma}_c^0(\sigma_{c1}^0,\sigma_{c2}^0)$ on f^0 obtained by scaling $\boldsymbol{\sigma}$ through:

$$\boldsymbol{\sigma}_c^0 = \frac{r}{|\boldsymbol{\sigma}|}\boldsymbol{\sigma}, \tag{2.41}$$

where r corresponds to the norm $|\boldsymbol{\sigma}_c^0|$, as illustrated in Fig. 2.6. To evaluate r as a function of $\boldsymbol{\sigma}$, the length of the 'radius' r is parametrically written as an ellipse with a single parameter θ.

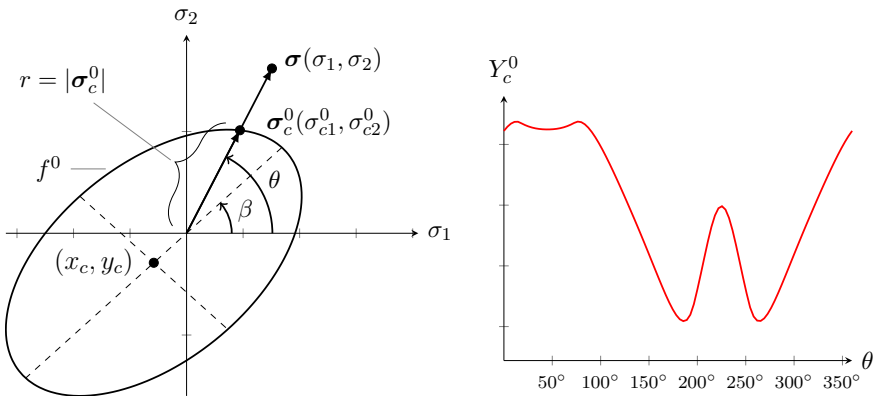


Figure 2.6: Definition of the length of r (left) and resulting Y_c^0 as a function of θ.

The parametric equation for r as a standard form of an ellipse can be deduced from the fact that Eq. (2.39) under plane stress assumption can be rewritten in principal stress space as:

$$f^0 = A\sigma_1^2 + B\sigma_1\sigma_2 + C\sigma_2^2 + D\sigma_1 + E\sigma_2 + F = 0, \tag{2.42}$$

with $A = C = \frac{1}{f_c f_t}$, $B = -\frac{1}{f_c f_t}$, $D = E = \frac{(f_c - f_t)}{f_c f_t}$ and $F = -1$. The discriminant of the equation above, which is defined as $\Delta = B^2 - 4AC$, is always negative, which implies that Eq. (2.42) represents an ellipse [41, 42]. In this case, Eq. (2.42) can be written as the standard expression for an ellipse centered at (x_c, y_c) and rotated through an angle β (see Fig. 2.6):

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1, \tag{2.43}$$

where

$$\begin{aligned} x &= (\sigma_1 - x_c)\cos\beta + (\sigma_2 - y_c)\sin\beta \\ y &= -(\sigma_1 - x_c)\sin\beta + (\sigma_2 - y_c)\cos\beta, \end{aligned} \tag{2.44}$$

in which x_c, y_c, β, and the axes of the ellipse a and b are expressed as a function of the constants A, B, C, D, E and F [41, 42]. For this particular case, one can show that $\beta = 45°$ and $x_c = y_c = D/B$. Substitution of the relations $\sigma_1 = r\cos\theta$ and $\sigma_2 = r\sin\theta$ into Eq. (2.44) and Eq. (2.43) leads to a quadratic equation, which is solved for r, where the positive root is chosen. Note that such approach can also directly be performed in principal strain space by expressing $\hat{J}_2$ and $\hat{I}_1$ as a function of principal strains and Lamé's elastic constants which could allow for generalization to 3D.

The remaining parameter Y_c^G is computed from the fracture energy G_c and l_c by considering the following relation [15]:

$$G_c = 2AY_c^G l_c, \tag{2.45}$$

with A the area under the curve d. For the point symmetric damage profile, $A = 0.5$.

2.2.4. MAPPING OPERATORS

In the TLS, integration points, from one load increment to another, are dynamically allocated for elements that are cut by the iso-0 and iso-l_c, in order to improve the accuracy of numerical integration. This is not an issue for elastic materials if one uses a total stress-strain formulation. On the other hand, in the case of crack propagation in elastic-plastic materials, history terms influence the local response. These history terms are stored at integration points. When the integration scheme changes, transfer of history terms from 'old' to 'new' integration points is needed. In this study, the inverse distance weighted interpolation [43] and the superconvergent patch recovery (SPR) [44–46] techniques for transferring plasticity terms between old and new integration schemes are compared. For the elasto-plastic model, the plastic strain tensor $\boldsymbol{\varepsilon}^{\mathrm{p}}$ and the equivalent plastic strain $\varepsilon_{\mathrm{eq}}^{\mathrm{p}}$ are the history variables that need to be transferred.

INVERSE DISTANCE WEIGHTED INTERPOLATION

Let s_i^{old} be an old history term at an old integration point i. With inverse distance-based interpolation, the new history term s_j^{new} at a new integration point j is given by:

$$s_j^{\mathrm{new}} = \frac{\sum_{i=1}^{n_{\mathrm{old}}} s_i^{\mathrm{old}} \left(1/l_{ij}\right)}{\sum_{i=1}^{n_{\mathrm{old}}} \left(1/l_{ij}\right)}, \tag{2.46}$$

where n_{old} is the number of old integration points in an element and l_{ij} is the distance between an old integration point i and a new integration point j inside the same element (see Fig. 2.7). If a new integration point coincides with an old one, the old history is kept.

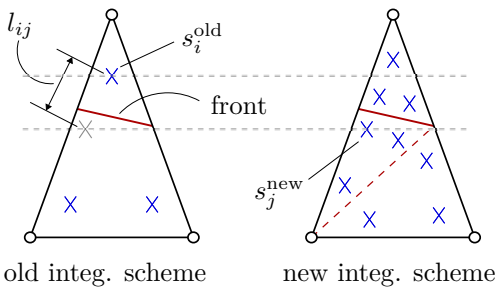


Figure 2.7: Inverse distance weighted interpolation: sub-triangulation.

SUPERCONVERGENT PATCH RECOVERY

The SPR technique is carried out in two steps. Firstly, a history term s_k^N at a node k within an element patch (see Fig. 2.8) is estimated by:

$$s_k^N = \mathbf{Pa}, \tag{2.47}$$

where $\mathbf{P}$ contains the appropriate terms of a complete polynomial expansion of order p and $\mathbf{a}$ is a set of unknown coefficients. For two dimensions and quadratic expansion, for instance:

$$\mathbf{P}(x,y) = \begin{bmatrix} 1 & x & y & x^2 & xy & y^2 \end{bmatrix} \tag{2.48}$$

and

$$\mathbf{a} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \end{bmatrix}^{\mathrm{T}}. \tag{2.49}$$

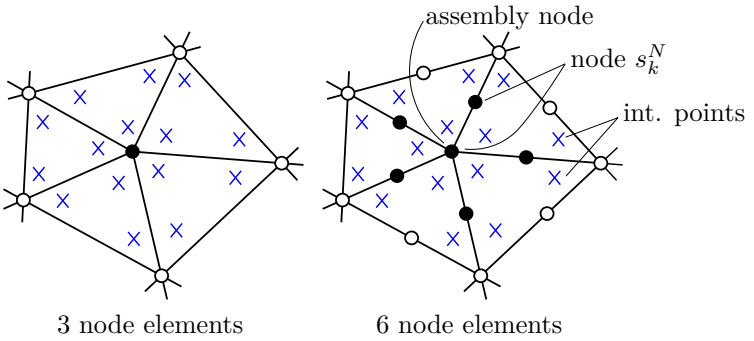


Figure 2.8: Triangular element patches for linear and quadratic finite elements: • nodal values determined by recovery procedure (Eq. (2.50)).

The coefficients in $\mathbf{a}$ are determined via the least square method fitting from the integration points within the element patch, which results in:

$$\mathbf{a} = \mathbf{A}^{-1}\mathbf{b}, \tag{2.50}$$

where

$$\mathbf{A} = \sum_{i=1}^{n_p} \mathbf{P}(x_i,y_i)^{\mathrm{T}}\mathbf{P}(x_i,y_i) \tag{2.51}$$

and

$$\mathbf{b} = \sum_{i=1}^{n_p} \mathbf{P}(x_i,y_i)^{\mathrm{T}} s_i^{\mathrm{old}}, \tag{2.52}$$

with n_p and (x_i,y_i) being the total number and coordinates of integration points in the element patch, respectively. Secondly, the new history terms at a new integration point j are obtained by shape function interpolation, which has the same order p as $\mathbf{P}$:

$$s_j^{\mathrm{new}} = \mathbf{N}\mathbf{s}^N. \tag{2.53}$$

It must be emphasized that Eq. (2.50) is evaluated for each history term using the same matrix solution. Besides, only a single evaluation of $\mathbf{A}$ is necessary per patch. Once $\mathbf{a}$

is determined, the recovery nodal values s^N are simply computed by inserting the appropriate coordinates into Eq. (2.50). In Fig. 2.8, for instance, the nodes for linear and quadratic triangular elements that are considered for recovery are shown. For internal element nodes ($p \geq 2$), which cannot be an assembly node, the history terms will be considered from several patches and they are therefore averaged as suggested in [44–46]. For nodes at boundaries and nodes that give rise to a patch with a single element, such nodes cannot be an assembly node and they are recovered as an internal element node [44, 45].

VERIFICATION: 1D PROBLEM

A 1D model is developed (see Fig. 2.9) to assess the performance of both operators for plasticity quantities and answer the question which the most suitable for the TLS with a moving front and frozen history behind the front. In this simulation, a uniform mesh of linear truss elements with two nodes, uniform cross-section area and two integration point are used. When an element is cut by the front, two more integration points are added to that element. An elementary constitutive model for linear isotropic hardening plasticity is used as presented in [47] and shown in Fig. 2.9. The expressions needed to implement the elastic predictor/return mapping algorithm and tangent modulus D for this model can also be found in a closed-form analytical manner in [47].

The bar is constantly loaded with rate $\dot{u}^0 = \Delta u^0 / \Delta t^0 = 0.01\,\mathrm{mm\,s^{-1}}$. Young's modulus, hardening modulus, length and cross-section area are, respectively, $E = 200\,\mathrm{GPa}$, $K = 5\,\mathrm{GPa}$, $L = 1.5\,\mathrm{m}$ and $A = 100\,\mathrm{mm}^2$. When the uniaxial stress reaches immediately around the yield stress $\sigma_y = 250\,\mathrm{MPa}$, the front l starts to move from left to right side. From this stress level, the front moves continuously with a constant velocity $v_n = 0.02\,\mathrm{mm\,s^{-1}}$ from $l = 0.55\,\mathrm{m}$ to $l = 1\,\mathrm{m}$. In line with the proposed TLS framework, the plasticity is not allowed to increase behind the front.

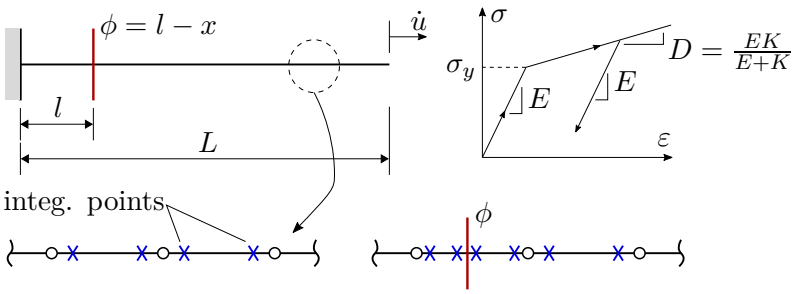


Figure 2.9: 1D problem for assessing the performance of mapping operators. The level set value ϕ at coordinate x is related to the position of the front l.

A reference response is also computed with a mesh of 500 elements. For this response, each finite element has 100 integration points and the integration scheme does not change. In Fig. 2.10, a comparison between results obtained with the two operators and the reference response is shown in terms of a load-displacement curve, obtained with a mesh with three elements for both operators, along with a convergence study. For the convergence study, the areas under load-displacement graphs are used and their relative differences are computed between the two operators and the reference response.

It is clear particularly for coarse meshes that the SPR technique performs better than inverse distance-based interpolation as reported in both graphs in Fig. 2.10. Therefore, for sake of robustness and accuracy, the SPR technique is used for all the numerical examples presented in this chapter.

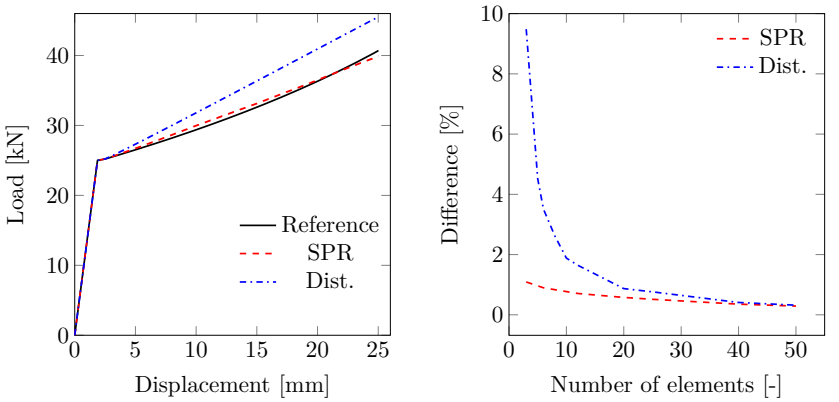


Figure 2.10: Typical load-displacement graph and convergence study for 1D problem.

2.2.5. ALGORITHM

In Box 2.1, the global algorithm solution, where the problem is solved in a staggered fashion [15, 29], is schematically summarized. Every time step consists of three main parts. Firstly, damage initiation is evaluated, ϕ is reinitialized in order to guarantee the properties of a signed distance function and $\bar{\phi}$ is computed. Secondly, with a given damage distribution, the displacements, and consequently strains and stresses are computed according to a standard finite element analysis for elasto-plastic constitutive models in conjunction with the concept of effective stress. The plastic state is only updated for integration points in undamaged zones. Finally, the displacements, permanent strains and $\bar{\phi}$ are used to compute the configurational force $\bar{Y}$ and the material resistance $\bar{Y}_c$. In this part, the velocities are first computed at nodes of elements that contain the front and subsequently extended by a fast marching method. Before going to the next time step, the time increment size is adjusted based on the stability condition if necessary.

The update of prescribed displacement for the following time step is performed according to the same adaptive time step size:

$$u \leftarrow u + \Delta u^0 \frac{\Delta t}{\Delta t^0}. \tag{2.54}$$

This allows to capture sharp load drops. If $\bar{Y}$ becomes very high, Δt will become very small, which results in a restraint on the increase in prescribed displacement during unstable damage growth.

For each time step:

1. Damage distribution:
   (a) Add new front ϕ_0 for nucleation if $Y \geq Y_c$ is met
   (b) Reinitialize ϕ on the whole domain with fast marching method (optionally only every n steps)
   (c) Assemble and solve linear system similar to Eq. (2.28) for $\bar{\phi}$
2. Finite element analysis:
   (a) Update of prescribed displacements (Eq. (2.54))
   (b) Compute displacements, strains and stresses. Plastic state is only evaluated for integration points in undamaged zone
3. Grow fronts:
   (a) Compute configurational force $\bar{Y}$ and material resistance $\bar{Y}_c$ (Eqs. (2.28) and (2.38))
   (b) For nodes on fronts, compute normal velocity v_n (Eq. (2.34))
   (c) Extend normal velocity over the entire domain: solving $\nabla\phi \cdot \nabla v_n = 0$ (Eq. (2.37)) with fast marching method
   (d) Adjust new time increment size Δt (Eq. (2.36))
   (e) Update level set field: $\phi \leftarrow \phi + v_n \Delta t$ (Eq. (2.33))

Box 2.1: Global algorithm for single time step.

2.3. RESULTS AND DISCUSSION

The numerical examples in this section are performed with $\kappa = 1$ for $\bar{Y}$ (Eq. (2.28)) and $\kappa = 1 \cdot 10^4$ for $\bar{\phi}$. For nucleation, the size of a new damage nucleus is set to $\phi_0 = 0.1 l_c$. Furthermore, the constant α_n in Eq. (2.36) is set to $\alpha_n = 0.5$, $\bar{\phi}_{\max} = l_c/3$ [29] and the default displacement rate is $\dot{u}^0 = \Delta u^0/\Delta t^0 = 0.005\,\mathrm{mm\,s^{-1}}$. The obtained results are quantitatively and qualitatively compared with those available in the literature. Unstructured meshes of triangles generated with Gmsh [48] are used.

2.3.1. V-NOTCHED BAR

In this section, the plane strain response of a V-notched specimen in tension is investigated. The aim of this example is to demonstrate the ability of the proposed model to deal with ductile fracture by means of comparisons against experimental data by Li et al. [49] and numerical results by Miehe et al. [50]. Boundary conditions and geometry of the V-notched specimen are demonstrated in Fig. 2.11. The model properties are listed in Table Table 2.1.

In this example, second order elements are used to avoid volumetric locking with the classic Von Mises model. However, linear elements are used for the discretization of the level set field ϕ and the normal velocity v_n because it simplifies the fast marching algorithms.

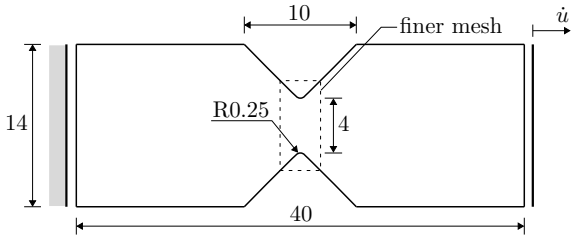


Figure 2.11: V-notched bar: boundary conditions and geometry and associated dimensions (in mm).

Names	**Values**
Young's modulus (E)	68.9 GPa
Poisson's ratio (ν)	0.33
Hardening law	$\sigma_y(\varepsilon^{\mathrm{p}}_{\mathrm{eq}}) = 700(0.03+\varepsilon^{\mathrm{p}}_{\mathrm{eq}})^{0.12}$ MPa
Ultimate stress (f_t, f_c)	600 MPa
Fracture energy (G_c)	18 N/mm
Plastic Poisson's ratio (ν_p)	0.5

Table 2.1: Model parameters for V-notched bar (Al-6061) [49–51].

The geometry of the problem leads to a non-uniform stress state near to the notches. A region around the notches with refined mesh is defined (see Fig. 2.11) where the effective element size $h = 0.05$ mm. The value of $h = 0.05$ mm was determined by carrying out a convergence study in terms of the peak load. In this simulation, the crack growth resistance parameter and critical length are, respectively, $\eta = 25\,\mathrm{s\,mm^{-1}}$ and $l_c = 0.4$ mm. The value of η is determined as fitting parameter. The influence of l_c and η on the response after choosing l_c based on the model geometry and the mesh size is addressed later.

The evolution of the damage front is illustrated in Fig. 2.12. In Fig. 2.13, the load-displacement graphs from Li et al. [49], Miehe et al. [50] and the TLS are drawn together. The results verify the accuracy of the proposed model.

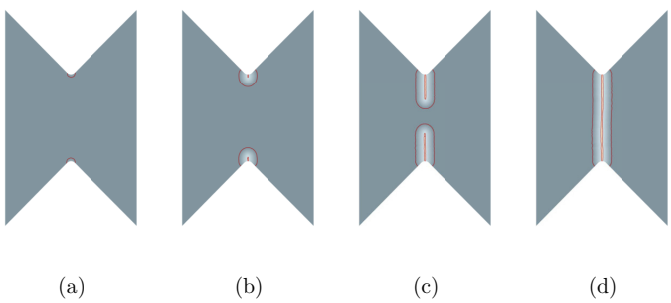


Figure 2.12: Evolution of the damage front on V-notched bar at different loading stages: (a) ≈ 3.6 kN, (b) ≈ 3.5 kN, (c) ≈ 2.0 kN and (d) final failure.

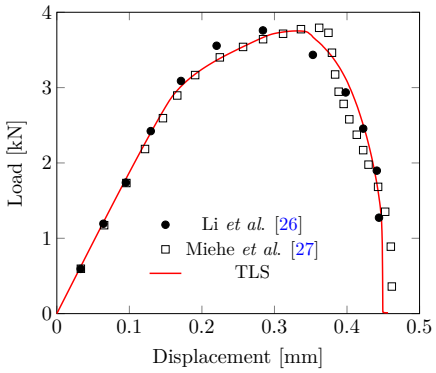


Figure 2.13: Load-displacement graphs for V-notched bar.

To investigate the influence of l_c on the global response, simulations have been performed with different values of l_c between 0.3 mm and 0.6 mm with a fixed value of $\eta = 25\,\mathrm{s\,mm^{-1}}$ and a fixed mesh. Load-displacement curves from these simulations are shown in Fig. 2.14. Unlike what was shown in [13, 15], where the global response in terms of load-displacement curves and energy dissipations did not change considerably when varying l_c for the TLS with linear elastic materials in a quasi-static context, a delayed-failure response for decreasing l_c is observed in the presence of plasticity. This is due to the fact that for small values of l_c, the stress state around the crack tip is higher than the stress state for larger values of l_c, which in turn leads to more plastic strain.

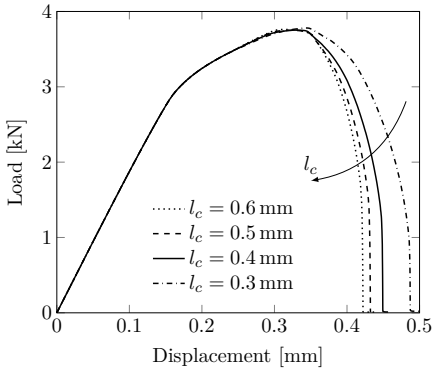


Figure 2.14: Influence of l_c on the global response. Increasing values of l_c are indicated by the arrow.

To illustrate the effect of the resistance parameter on the post-critical range after damage initialization, the same simulation with $l_c = 0.4$ mm is repeated with values of $\eta = 15\,\mathrm{s\,mm^{-1}}$, $\eta = 20\,\mathrm{s\,mm^{-1}}$, $\eta = 25\,\mathrm{s\,mm^{-1}}$ and $\eta = 30\,\mathrm{s\,mm^{-1}}$. The results are compared in terms of load-displacement curves in Fig. 2.15. It can be observed that η influences the shape of the post-critical response. As expected, delayed failure behavior is obtained by increasing the value of η. The value of $\eta = 25\,\mathrm{s\,mm^{-1}}$, which gave the good fit in Fig. 2.13, is clearly in the regime where there is significant influence of η on the re-

sponse. This means that the actual fracture energy in the simulation is rate-dependent and higher than the G_c from Table 2.1.

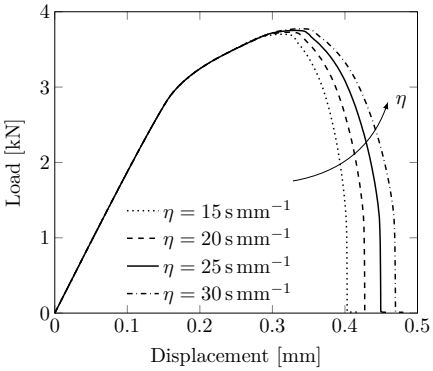


Figure 2.15: Influence of η on peak response. Increasing values of η are indicated by the arrow.

In order to assess the performance of both mapping operators in a more realistic simulation, once again, a convergence study is carried out. In Fig. 2.16, the peak load obtained with inverse distance-based interpolation and SPR technique are compared with the result obtained by a reference model in which SPR and $h = 0.015\,\mathrm{mm}$ are used. The same trend is observed that has been presented for the 1D model, in which both operators converge toward the same response with mesh refinement and that the results with SPR are generally more accurate. It is also observed that the mesh with $h = 0.05\,\mathrm{mm}$, which was used in Fig. 2.13, is suitable since the difference in terms of peak load is negligible.

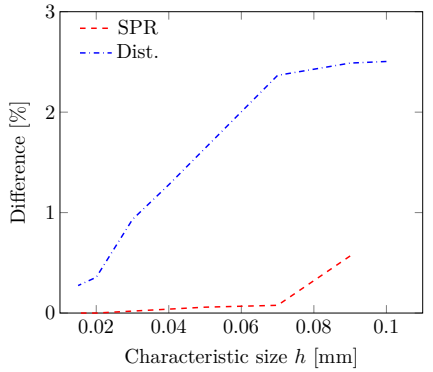


Figure 2.16: V-notched bar: convergence study for the inverse distance-based operator in terms of peak load.

2.3.2. RAIL SHEAR TEST

This numerical example is inspired by rail shear test for mode II failure following Van der Meer and Sluys [29]. The case consists of a sandwich with stiff faces and a weak core (see

Fig. 2.17). The faces are loaded in opposite direction so that the core is sheared. This setup mimics the delamination process in composites, where the stiff faces represent the plies or fibers and the core represents the resin-rich region around the interface.

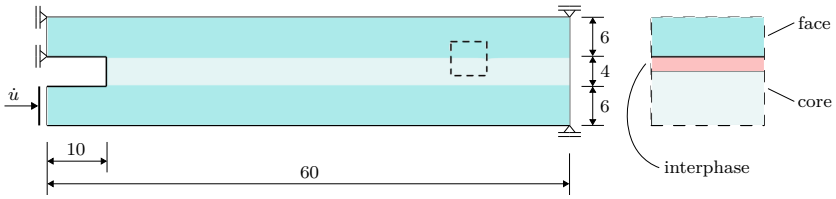


Figure 2.17: Rail shear test: boundary conditions and geometry and associated dimensions (in mm).

In addition, three different cross-sections are considered as depicted in Fig. 2.18. These variations in cross-section are identical to those of the specimens that have been tested by Rogers [26] to produce cusp-like features on Polyvinylchloride (PVC) foam material. The curvature and width of profiles mimic respectively the influence of fibers radius and the inter-fiber spacing in composites. In the 2D model, the geometry of the cross-section is modeled by varying the model thickness as a function of position.

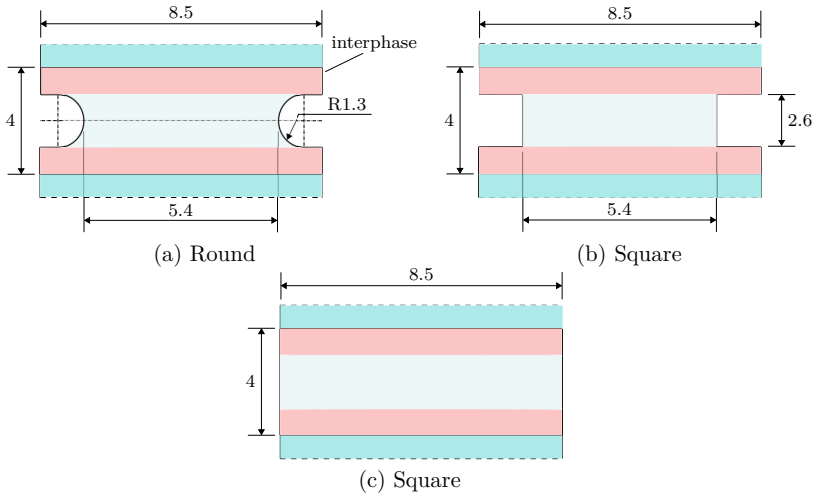


Figure 2.18: Rail shear test: cross-section profiles (dimensions in mm).

Young's modulus, Poisson's ratio, and fracture energy of the core material are, respectively, $E = 3760\,\text{MPa}$, $\nu = 0.3$, and $G_c = 0.9\,\text{N/mm}$. For plasticity, a plastic Poisson's ratio of 0.39 is used, and the fundamental hardening curves are given in Fig. 2.19 [37, 39]. For the face material, the properties are $E = 200\,\text{GPa}$, $\nu = 0.3$, and $G_c = 9\,\text{N/mm}$. The typical element size h is 0.14 mm throughout the core, and the critical length l_c is equal to 0.6 mm. The distance between a new damage circle and existing damage is set to be at least 3 mm [29]. The resistance parameter is $\eta = 1\,\text{s}\,\text{mm}^{-1}$. Linear elements are used for the discretization of all fields.

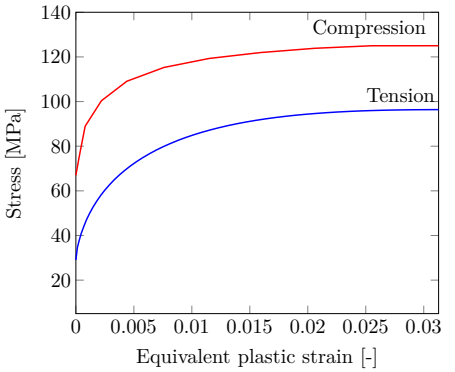


Figure 2.19: Input hardening curves for plasticity model.

When the crack reaches the interface between face and core, sliding deformations lead to one of the principal strains in this zone become negative, in this case the asymmetric constitutive law (Eq. (2.22)) leads to stiffness recovery and stress transfer across the crack, which is undesired here. In order to guarantee traction-free sliding deformation in this zone, the interphase constitutive law introduced in [29] is applied (see Figs. 2.17 and 2.18). This constitutive law makes use of a vector **n** normal to the interface and accounts only for stiffness recovery on the strain component along this vector.

First, the difference in response between the proposed model and the earlier TLS model without plasticity for this shear test setup is illustrated by means of load-displacement curves in Fig. 2.20. Only the round cross-section is considered in this comparison. The difference between the two frameworks in the pre-peak behavior is caused by the added plasticity, while the absence of oscillations in the post-peak behavior is related to the new definition of velocity from Eq. (2.34).

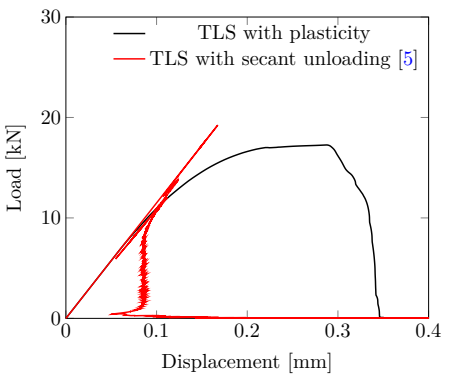


Figure 2.20: Load-displacement response for current and proposed TLS frameworks in the rail shear test.

The framework in [13, 15, 29] is based on a unit load analysis in each time step, which assumes a secant unloading behavior, and has the ability of capturing snap-backs. The oscillations that are observed in the post-peak response in Fig. 2.20 are related to nu-

merical noise in the maximum value for $\bar{Y}$ that is used to compute the load scale factor such that the scaled maximum value for the configurational force is exactly equal to the material resistance $\bar{Y}_c$. This algorithm based on a unit load analysis and a load scale factor is only applicable to mechanical problems with proportional loading and secant unloading.

Introduction of plasticity in the mechanical problem requires that the computation of displacements is performed at the actual load level. In this sense, the criterion that the configurational force should not exceed the material resistance at any point should be determined iteratively or relaxed by introducing a viscous parameter between front velocity and configurational force, as proposed in this chapter, following [40]. The absence of oscillations is a positive side-effect of this change in loading scheme.

The evolution of damage and the equivalent plastic strain distributions for the round cross-section are shown in Fig. 2.21. It can be observed that damage initiation takes place around the onset of perfect plasticity, first with a single damage spot near the left edge of the soft material and soon after in a series of spaced damage nuclei. As the load increases, all damage nuclei grow to a certain size, until enough energy is available to let a number of inclined cracks grow from these nuclei as the load drops. It is interesting to note that two of the damage spots do not evolve into an inclined crack, which indicates that the numerical spacing does not completely govern the final crack pattern. The same observation was also made in [29] for linear elastic materials. Eventually, the load drops to zero as the inclined cracks coalesce to form a single crack. It can be observed that much of the crack growth takes place in the steep final drop of the load-displacement graph. The adaptive time step according to Eq. (2.36) ensures that time increments and consequently displacement increments are very small in this phase.

Figures 2.22 and 2.23 show the load-displacement curves and final damage distributions obtained with three different cross-sections. It is observed that the change in profile shape has a considerable influence on the fracture morphologies and equilibrium curves. For the round configuration, the number of randomly spaced inclined cracks along the specimen length, which eventually coalesced to form cusps, is larger than for the square and flat configurations. Rogers [26] reported the same trend from experimental observations on PVC foam specimens under similar shear loading conditions. The initial stiffness of the flat specimen is higher than that of the round and square configurations, because of the flexibility that is introduced by side grooving.

2.4. CONCLUSION

In this chapter, the TLS method for non-local damage modeling has been extended to include plasticity: the elasto-plastic constitutive model for epoxy resin by Melro et al. [37] has been combined with the TLS damage formulation, and a new loading scheme to take into account permanent strain has been proposed.

In addition, due to plasticity that bounds stress states, a new criterion for damage nucleation has been developed in order to relate crack initiation under different stress states to a given failure surface. Since the change in the integration scheme for those elements that are cut by the iso-0 and iso-l_c curves, the SPR technique was found to be accurate for transfer of history.

The influence of l_c and η on the global response has been investigated and found to

be of similar nature. For a given value of l_c, the optimal value of η has been determined through fitting. In practice, the values from which l_c can be chosen is limited. An upper bound is given by the geometry of the problem at hand, in which the width of the damage band has to be relatively smaller than the geometrical dimensions of the problem. A lower bound for l_c relies on the computational cost, because elements have to be several times smaller than l_c.

The TLS was validated from a good agreement with experimental and numerical results for ductile fracture in a V-notched bar in tension.

The proposed model was successfully applied to the simulation of shear failure including cusp formation. By varying the profile geometry of the core, different load-displacement graphs and fracture morphologies were produced. Cusp development comparable to that in composites was more pronounced in curved-profile configuration, which is in agreement with experimental observations.

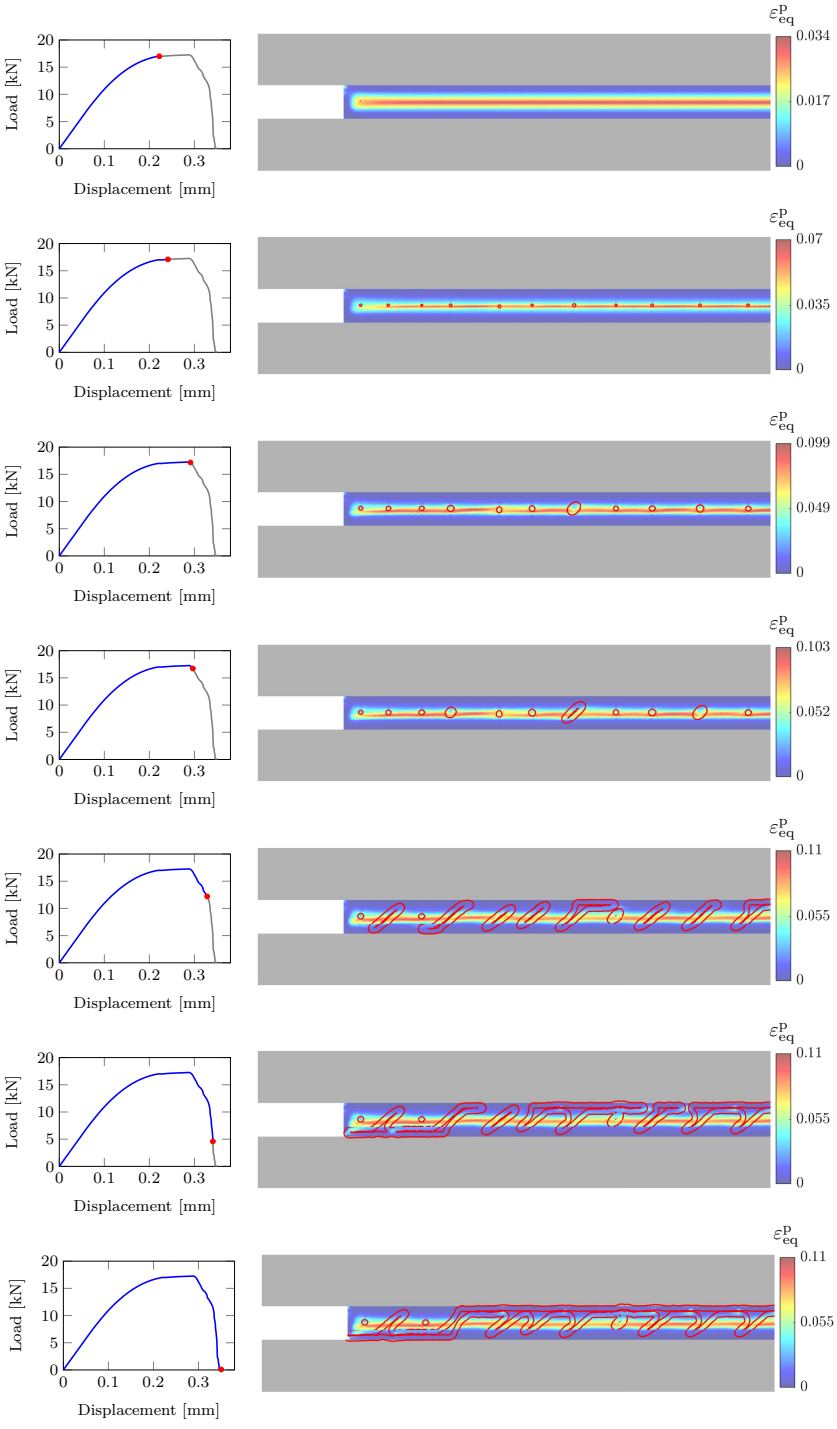


Figure 2.21: Rail shear test: damage and equivalent plastic strain distributions for the round configuration. The red bullets correspond to some time steps when the time increment size was reduced.

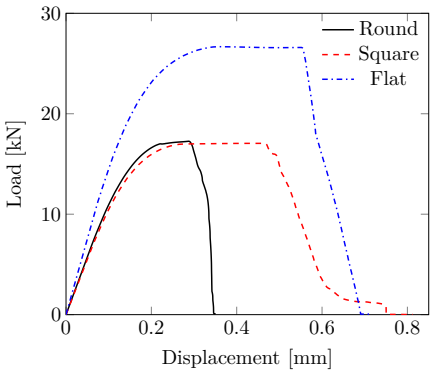
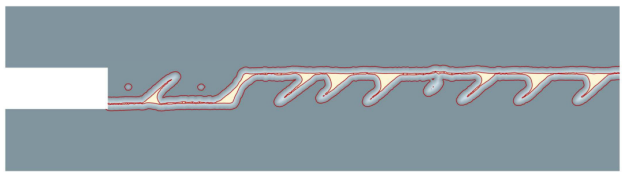
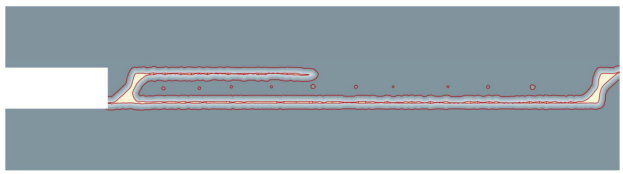


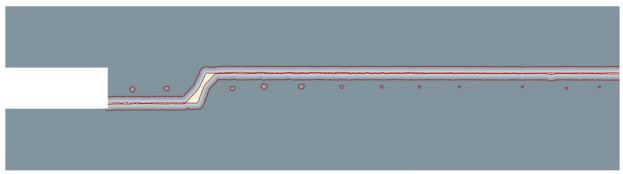
Figure 2.22: Load-displacements curves for round, square and flat configurations.



(a) Round



(b) Square



(c) Flat

Figure 2.23: Rail shear test: final crack distribution for (a) round, (b) square and (c) flat configurations.

3

PARALLEL COMPUTING WITH THE THICK LEVEL SET METHOD

3.1. INTRODUCTION

The Thick Level Set (TLS) method, first introduced by Moës et al. [13], is a damage model that contains a non-local damage definition to prevent spurious strain localization. In this method, the location of the damage front that separates the damaged material from the undamaged material is tracked as the zero level set of an auxiliary field in which its evolution is dictated by the non-local energy release rate of the material. The TLS damage variable depends on the distance to the damage front as evaluated with the signed-distance level set field and varies over a thick band of material with a predefined width according to a user-defined damage function. Since its first advent, the TLS has been expanded in order to enhance its numerical implementation for quasi-static loading condition [15, 19], to deal with three-dimensional quasi-static problems [18] and dynamics [19], to couple with cohesive zone models [1], treat fatigue crack growth [20], to improve the control of damage initiation and representation of free sliding in shear [29], and to couple damage with plasticity [34].

The TLS works with a staggered solution scheme in which displacements and damage are separately computed. From an existing level set field, i.e., a given damage distribution, an equilibrium problem is solved for the displacement field in a standard finite element analysis. After computing the displacements, the configurational force for front movement is evaluated with which the level set field is updated. As a result of the staggered approach, the TLS provides a robust framework for handling topological events like merging and branching. From an implementation point of view, the global solution scheme of the TLS contains three modules, where each one encapsulates specific tasks: update of the level set field, equilibrium solution, and damage front evolution [20, 29, 34].

Apart from minor changes to its introduction and conclusion sections, this chapter was integrally extract from L. A. T. Mororó and F. P. van der Meer. *SIAM Journal on Scientific Computing* 43 (6) (2021) [52].

Despite its robustness, the TLS can be a time-demanding approach, mainly because of the equilibrium solution phase [53]. The high computational demand comes from the fact that the TLS requires element sizes smaller than the width of the damaged band in order to achieve a desirable accuracy, especially for the computation of non-local quantities [15, 20], giving rise to a system of equations with many degrees of freedom (DOFs). This issue can be amplified if the size of the width of the damage band is constrained to be small relative to the geometry of the problem being investigated, for instance, interlaminar cusp formation in a polymer matrix of composite materials subjected to mixed mode loading condition [54], which takes place in a very narrow area and involves multiple cracks that eventually merge. Additionally, in order to guarantee numerical stability of the level set update, the damage front advance is constrained such that it does not move more than one element length per time step [15, 20, 29, 34]. Therefore, for simulations up to final failure of specimens with long cracks, many time steps are required [29, 34].

Parallel computing may be used to mitigate the computational effort associated with finite element simulations, where many operations (e.g., assembly of matrices and vectors) can be performed simultaneously for different parts of the domain on different cores. To take advantage of the parallel architecture of a machine for solving large systems of equations, numerical techniques for decomposing the original problem into collaborating sub-problems are needed. In a finite element context, Domain Decomposition (DD) methods are used to build parallel frameworks running on different cores [55, 56]. The main idea behind a DD approach is to divide the whole domain into subdomains that can be solved almost independently on different cores. Since the solution on one subdomain is not completely independent from other subdomains, some exchange of data limited to the interface (or to a small overlapping region) between neighboring subdomains is necessary. The first mathematical studies on DD gave rise to a family of Schwarz algorithms based on overlapping domains (e.g., Restricted Additive Schwarz method [57]). Later, non-overlapping methods whose interpretation is more mechanically oriented were proposed, such as Finite Element Tearing and Interconnecting (FETI) method [58] and Balanced Domain Decomposition (BDD) [59].

The main premise of this chapter is that such strategies can be advantageous to speed up the equilibrium solution stage that constitutes the main computational bottleneck related to the TLS method. However, when applying domain decomposition to one task of the TLS, care is required for the remaining analysis tasks (update of the level set field and damage front evolution).

In this chapter, we seek to describe how to apply the DD framework to obtain an efficient parallel version of the TLS method. It demonstrates how to handle the TLS-specific analysis phases that spawn multiple operations on different processors, where time-demanding parts of the solution scheme are performed in parallel, while other parts that require global solution strategies are kept sequential. Two slightly different TLS implementations are considered as starting point: firstly, the version by Mororó and Van der Meer [34], for ductile fracture and secondly, the version by Van der Meer and Sluys [29], which assumes a secant unloading behavior of material. Both of these build on original concepts of the first paper on the TLS method by Moës et al. [13] and the improvements on its implementation by Bernard et al. [15]. The parallel iterative solver combined with

Schwarz-type DD strategy by Lingen et al. [60] is used.

It is worth mentioning that apart from the two TLS implementations considered here, the proposed parallel framework can easily be extended to other TLS-based methods, such as the interfacial thick level set method by Latifi et al. [20].

The chapter is structured as follows. Section 3.2 is dedicated to presenting the main features that comprise the TLS method. The objective of Section 3.2 is to be complete and to provide the context for the parallel framework introduced in Section 3.3 without providing motivation for all details of the TLS as presented elsewhere in literature. Numerical examples considering linear elasticity and plasticity are presented in Section 3.4 and used to assess the performance of the proposed framework. Numerical results are presented with focus on the scalability of the framework and accuracy of the parallel solution. Finally, conclusions are presented in Section 3.5.

3.2. THE THICK LEVEL SET METHOD

The TLS method makes use of the level set method [16] to track the location of a damage front Γ_0 that separates the undamaged material from a degraded region (see Fig. 3.1). The damage front is defined as coinciding with the zero level set (or the 'iso-0') of an auxiliary field $\phi(\mathbf{x})$, the level set field. The level set field ϕ is chosen to be the signed distance function to Γ_0 in which its gradient satisfies the eikonal equation:

$$\| \nabla \phi \| = 1 \quad \text{on} \quad \Omega, \tag{3.1}$$

where Ω is the domain on which ϕ is defined. This equation guarantees that the absolute value of ϕ at a given point $\mathbf{x}$ is the shortest distance between that point and Γ_0. On a discretized finite element domain, the definition of ϕ at $\mathbf{x}$ is determined by interpolating the values of ϕ from nodes to $\mathbf{x}$ using finite element shape functions.

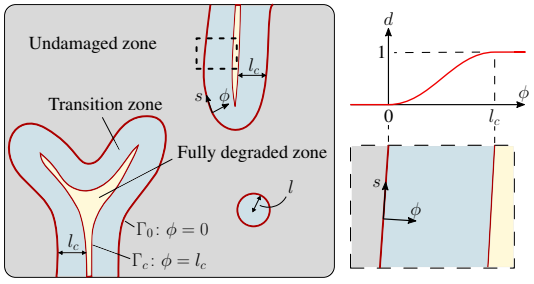


Figure 3.1: The TLS makes use of a single level set function to describe multiple damaged zones; the damage variable d is a function of the level set ϕ.

In the TLS, a damage variable is introduced which is defined as a function of ϕ. The damage variable is constrained to follow a given profile in a transition zone with fixed length between undamaged and fully degraded zones as:

$$d(\phi) = \begin{cases} 0, & \phi \leq 0 \\ q(\phi), & 0 < \phi \leq l_c \\ 1, & \phi > l_c \end{cases}, \tag{3.2}$$

where $q(\phi)$ is a function that has the properties of $q(0) = 0$, $q(l_c) = 1$ and $q'(\phi) \geq 0$ on $\phi \in [0, l_c]$. Therefore, the damage variable changes from zero to one as ϕ goes from zero to the critical length l_c over a band bounded by $\Gamma_0 : \phi = 0$ and $\Gamma_c : \phi = l_c$. In this chapter, the arctangent profile proposed by Bernard et al. [15] is used for all numerical simulations:

$$q(\phi) = c_2 \arctan\left(c_1 \left(\frac{\phi}{l_c} - c_3\right)\right) + c_4 \tag{3.3}$$

with $c_1 = 10$ and $c_3 = 0.5$ and the other coefficients given by $c_4 = -c_2 \arctan(-c_1 c_3)$ and $c_2 = (\arctan(c_1(1 - c_3)) - \arctan(-c_1 c_3))^{-1}$.

A summary of the staggered algorithm used in this chapter is given in Fig. 3.2. For every time step, there are three main modules (or *analysis phases*) that are named *level set update*, *equilibrium solution* and *front evolution*. The level set update phase consists of the update of the level set field and its reinitialization, evaluation of damage initiation given an elastic strain field $\boldsymbol{\varepsilon}^e$, and computation of the size of damaged zones $\bar{\phi}$. In the equilibrium solution phase, for a given damage distribution (which could consist of zero damage throughout Ω), displacements and consequently strains and stresses are computed in a standard finite element analysis. In the front evolution phase, $\bar{\phi}$ and the elastic strain from the equilibrium solution, $\boldsymbol{\varepsilon}^e$, are used to compute the non-local configurational force $\bar{Y}$ and the averaged material resistance to damage growth $\bar{Y}_c$, which in turn are used to compute the front velocity v_n along the front that is subsequently extended throughout Ω. With the velocity at hand, the resulting new level set field can be determined and used for the next time step.

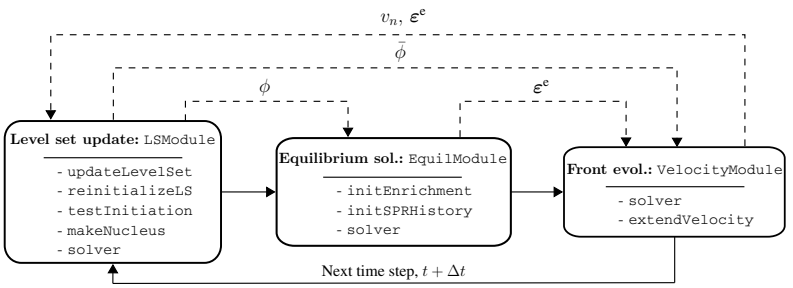


Figure 3.2: The global sequential staggered solution scheme for a single time step. Dashed arrows represent the data exchange among the three modules.

The remainder of this section is dedicated to outlining the main operations for each of the three modules. Modular pseudo-codes are provided to clarify the implementation. Parts of these pseudo-codes are highlighted in blue to indicate modifications related to the parallelism that will be addressed in Section 3.3. In the present section, where the original sequential algorithm is presented, these parts can be ignored. For sake of compactness, the framework for ductile fracture is firstly presented, and the other one for

linear elastic fracture assuming secant unloading behavior is then addressed, inheriting the structure from the former.

3.2.1. LEVEL SET UPDATE

The first task in every time step is to define the level set field for that step. In the first time step, the level set field is initialized. In all other time steps it is updated with the nodal normal velocities, as shown in Algorithm 3.1. The level set field at every node belonging to $\mathcal{N}$ (the complete set of nodes in the mesh) is then updated as:

$$\phi \leftarrow \phi + v_n \Delta t, \tag{3.4}$$

where Δt is the time increment size. In order to guarantee the numerical stability of the level set update, Δt is constrained as [34, 40]:

$$\Delta t = \min\left\{\Delta t^0, \alpha_n \frac{h}{\max\{v_n\}}\right\}, \tag{3.5}$$

in which Δt^0 is the default and maximum time increment size, α_n is a constant defined as $0 < \alpha_n < 1$, h is the characteristic size of the smallest element[1], and $\max\{v_n\}$ is the largest value of v_n over the entire domain.

Algorithm 3.1 The `updateLevelSet` algorithm.

Input: the nodal normal velocities v_n; the default and maximum time increment size Δt^0; and the constant α_n and parameter h
Output: the updated level set field ϕ at nodes and time increment size Δt

$v_{\max} \leftarrow 0$ /* Compute the maximum velocity v_{max} */
for all node $i \in \mathcal{N}$ **do**
  if $v_{ni} > v_{\max}$ **then**
    $v_{\max} \leftarrow v_{ni}$
  end if
end for
`Allreduce(`$v_{\max}$`)` /* Get the maximum v_{max} from all processes */
$\Delta t \leftarrow \alpha_n \frac{h}{v_{\max}}$
if $\Delta t^0 < \Delta t$ **then** /* Update the time increment size, if required */
  $\Delta t \leftarrow \Delta t^0$
end if
for all node $i \in \mathcal{N}$ **do** /* Update the level set field */
  $\phi_i \leftarrow \phi_i + v_{ni} \Delta t$
end for

In theory, when the level set field is updated with $v_n \Delta t$ and a properly extended velocity field, the updated level set field obtained by Eq. (3.4) remains a signed distance function. However, the discrete nature of the level set update deteriorates the property of Eq. (3.1). Thus, a reinitialization procedure is periodically performed with a fast marching algorithm [16, 40] in order to keep ϕ as an accurate representation of the signed distance function. Since it is a relatively cheap procedure [29, 34, 40], this reinitialization is performed every time step.

[1] h is defined as the length of the diagonal of the smallest possible rectangle around an individual element in the mesh.

Next, an initiation check is performed at every integration point in a user-defined set of elements $\mathscr{E}_{\text{nucl}}$ where nucleation is allowed, according to Algorithm 3.2. A damage nucleus, i.e., a small region with positive level set values, is inserted when the local driving force for damage growth Y that depends on the elastic strain tensor $\boldsymbol{\varepsilon}^{\text{e}}$ is greater than or equal to the material resistance to damage initiation Y_c^0. If $Y \geq Y_c^0$ is met, a circle with radius ϕ_0 is inserted around the point $\mathbf{x}_{\text{nucl}}$ with the highest ratio Y/Y_c^0. The initiation check is only performed in those elements belonging to $\mathscr{E}_{\text{nucl}}$ that are at least a distance ϕ_{spacing} away from an existing damage front.

Algorithm 3.2 The `testInitiation` algorithm.

Input: the parameter ϕ_{spacing}; the elastic strain field; and a function `getIPCoords` to retrieve the coordinates of a integration point j for a given element i
Output: the highest value of Y/Y_c^0 and its corresponding location $\mathbf{x}_{\text{nucl}}$

for all element $i \in \mathscr{E}_{\text{nucl}}$ **do**
  if any node-ϕ-values of element $i > -\phi_{\text{spacing}}$ **then**
    continue
  end if
  $max \leftarrow 0$ `/* initialize the maximum value of Y/Y_c^0 */`
  for all integration point j **do**
    $\mathbf{x}_j \leftarrow$ `getIPCoords`(i, j) `/* Get coordinates of integ. point j */`
    $Y_j \leftarrow$ Eq. (3.9)
    $Y_{cj}^0 \leftarrow$ Eq. (3.17)
    $ratio \leftarrow \frac{Y_j}{Y_{cj}^0}$
    if $ratio > max$ **then** `/* Update max and x_nucl, if required */`
      $max \leftarrow ratio$
      $\mathbf{x}_{\text{nucl}} \leftarrow \mathbf{x}_j$
    end if
  end for
end for
return max and $\mathbf{x}_{\text{nucl}}$

After the nucleation check, if a new damage front is added to the problem, the value of ϕ is checked at every node belonging to $\mathscr{N}$ and updated if the node is closer to the new front than to the existing front, as schematically shown in Algorithm 3.3.

Algorithm 3.3 The `makeNucleus` algorithm.

Input: the nucleus coordinates $\mathbf{x}_{\text{nucl}}$ and size of new front ϕ_0
Output: the updated level set field ϕ at nodes after nucleation

for all node $i \in \mathscr{N}$ **do**
  $d_i \leftarrow \| \mathbf{x}_i - \mathbf{x}_{\text{nucl}} \|$
  $\phi_n \leftarrow \phi_0 - d_i$
  if $\phi_n > \phi_i$ **then**
    $\phi_i \leftarrow \phi_n$
  end if
end for

The level set update phase is finished with the computation of a measure for the size of enclosed damaged zones. For this purpose, the averaged level set value $\bar{\phi}$ is computed in each time step in a similar way to $\bar{Y}$ in Eq. (3.13) by substituting level set values and unknowns $\bar{\phi}$ for Y and $\bar{Y}$, respectively, and by leaving out the weight factor d' in Eqs. (3.14) and (3.16) [29]. Unlike the computation of $\bar{Y}$, in which only the variation in the normal direction of $\bar{Y}$ is eliminated by means of Lagrange multipliers, the variation of $\bar{\phi}$ along the front is also eliminated by considering a high value for κ [29]. By eliminating the vari-

ation of $\bar{\phi}$ in both directions, a single representative value for each individual damaged subdomain is obtained. By adopting the structure of Eq. (3.13), the computation of $\bar{\phi}$ does not add much complexity to the framework since its implementation inherits the basic structure from what is already needed for the computation of $\bar{Y}$ and $\bar{Y}_c$.

The level set update phase is summarized in Algorithm 3.4.

Algorithm 3.4 The `LSModule` algorithm.

Input: the nodal normal velocity v_n; the time increment size Δt^0; the constant h; the parameter α_n; the size of new front ϕ_0; the parameter ϕ_{spacing} and elastic strain field
Output: the insertion of a new damage front and updated level set field ϕ

```
updateLevelSet(v_n, Δt^0, α_n, h)
Gather(φ)                                   /* Gather updated φ to the root process */
reinitializeLS()
Scatter(φ)                                  /* Scatter φ all over the processes */
Y/Y_c^0, x_nucl ← testInitiation(φ_spacing, ε^e)
Allreduce(Y/Y_c^0, x_nucl)                  /* Get the maximum Y/Y_c^0 and x_nucl from all processes */
if Y/Y_c^0 ≥ 1 then
    makeNucleus(x_nucl, φ_0)
end if
Gather(K, L, f^φ)                           /* Gather matrices and right-hand side vector for Eq. (3.13) */
φ̄ ← solver( Eq. (3.13) )
Scatter(φ̄)                                  /* Scatter φ̄ over the processes */
```

3.2.2. EQUILIBRIUM SOLUTION

Following [34], the equilibrium solution phase is executed in the framework of elasto-plastic finite element analysis for a given damage distribution assuming small displacements and additive decomposition of the total strain $\boldsymbol{\varepsilon}$ into an elastic part $\boldsymbol{\varepsilon}^{\text{e}}$ and a plastic part $\boldsymbol{\varepsilon}^{\text{p}}$, i.e., $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^{\text{e}} + \boldsymbol{\varepsilon}^{\text{p}}$. Moreover, plasticity is only allowed to evolve in the undamaged material and not in regions where $\phi > 0$.

The equilibrium equation without body force and the relation between the total strain $\boldsymbol{\varepsilon}$ and the displacement field $\mathbf{u}$ in Ω read, respectively, $\nabla \cdot \boldsymbol{\sigma} = \mathbf{0}$ and $\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \mathbf{u} + \nabla \mathbf{u}^{\text{T}}\right)$, in which $\boldsymbol{\sigma}$ is the stress tensor. The displacement field is subjected to Dirichlet boundary conditions $\mathbf{u} = \hat{\mathbf{u}}$ on Ω_u.

Under the hypothesis of decoupling between elasticity damage and plastic hardening, the specific free energy ψ is written as a function of the elastic strain $\boldsymbol{\varepsilon}^{\text{e}}$, damage variable d, and equivalent plastic strain $\varepsilon^{\text{p}}_{\text{eq}}$ and is split as $\psi(\boldsymbol{\varepsilon}^{\text{e}}, d, \varepsilon^{\text{p}}_{\text{eq}}) = \psi^{\text{ed}}(\boldsymbol{\varepsilon}^{\text{e}}, d) + \psi^{\text{p}}(\varepsilon^{\text{p}}_{\text{eq}})$ into a sum of an elastic-damage contribution ψ^{ed} and a contribution due to hardening ψ^{p}.

The following elastic-damage energy density that takes into account stiffness recovery under compression [15] is used in all numerical simulations:

$$\psi^{\text{ed}}(\boldsymbol{\varepsilon}^{\text{e}}, d) = \mu(1 - \alpha_i d)(\varepsilon^{\text{e}}_i)^2 + \frac{\lambda}{2}(1 - \alpha_v d)\text{tr}(\boldsymbol{\varepsilon}^{\text{e}})^2, \tag{3.6}$$

where λ and μ are Lamé's elastic constants, ε^{e}_i the principal strain values, $\text{tr}(\boldsymbol{\varepsilon}^{\text{e}})$ the trace of the elastic strain tensor,

$$\alpha_i = \begin{cases} 1, & \varepsilon_i^{\mathrm{e}} > 0 \\ 0, & \varepsilon_i^{\mathrm{e}} < 0 \end{cases} \quad \text{and} \quad \alpha_v = \begin{cases} 1, & \mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}}) > 0 \\ 0, & \mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}}) < 0 \end{cases}. \tag{3.7}$$

With Eq. (3.6), the constitutive relation in principal stress space and the local driving force for damage growth can, respectively, be expressed as:

$$\sigma_i = \frac{\partial \psi^{\mathrm{ed}}}{\partial \varepsilon_i^{\mathrm{e}}} = 2\mu(1-\alpha_i d)\varepsilon_i^{\mathrm{e}} + \lambda(1-\alpha_v d)\mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}}) \tag{3.8}$$

and

$$Y = \frac{\partial \psi^{\mathrm{ed}}}{\partial d} = -\mu\alpha_i(\varepsilon_i^{\mathrm{e}})^2 - \frac{\lambda}{2}\alpha_v \mathrm{tr}(\boldsymbol{\varepsilon}^{\mathrm{e}})^2. \tag{3.9}$$

In regions where $\phi \leq 0$, the basic form of the constitutive law for plasticity is

$$\boldsymbol{\sigma} = \mathbf{D}^{\mathrm{e}} : \boldsymbol{\varepsilon}^{\mathrm{e}} = \mathbf{D}^{\mathrm{e}} : \left(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^{\mathrm{p}}\right), \tag{3.10}$$

where $\mathbf{D}^{\mathrm{e}}$ is the elasticity tensor from Hooke's law. The plastic strain $\boldsymbol{\varepsilon}^{\mathrm{p}}$, whose evolution is defined by a plastic flow rule, is computed by using an elastic predictor/return mapping algorithm so that $\boldsymbol{\sigma}$ satisfies the yield criterion $f(\boldsymbol{\sigma}, \varepsilon_{\mathrm{eq}}^{\mathrm{p}}) \leq 0$ [38].

Inserting the constitutive relations in Eqs. (3.8) and (3.10) into the equilibrium equation results in a nonlinear system of equations that is iteratively solved with the Newton-Raphson method. In order to capture sharp load drops during unstable damage growth, the update of prescribed displacements $\hat{u}$ is performed with the adaptive time step size from Eq. (3.5), $\hat{u} \leftarrow \hat{u} + \Delta\hat{u}^0 \frac{\Delta t}{\Delta t^0}$, with $\Delta\hat{u}^0$ being the initial displacement increment [34, 40].

The region with $\phi > l_c$ where $d = 1$ represents stress-free macro-cracks in the TLS method. In simulations with regular finite elements, this region needs to be at least one row of elements wide to represent this stress-free state, causing mesh dependency [29]. In order to achieve a stress-free state in elements that are only partially in $\phi > l_c$, the enrichment strategy proposed by Bernard et al. [15] is employed, which allows for a strain discontinuity across the iso-l_c. In this strategy, extra DOFs are added to a node inside the fully degraded region if the elements in its node support are cut twice by the iso-l_c, as shown in Fig. 3.3. From an implementation point of view, nodes are dynamically enriched and unenriched as the damage front advances.

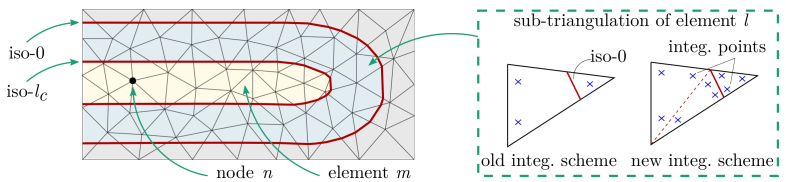


Figure 3.3: On the left, the nodes of element m and node n exemplify, respectively, typical cases of nodes without and with the necessity of being enriched. In the dashed box on the right, the change in integration scheme of the element l crossed by the iso-0 is illustrated (adapted from [15]).

The enrichment function is constructed as a continuous ramp function across the fully degraded area where $\phi > l_c$. In practice, this scheme divides the element support

for an enriched node into two lists of elements: positive and negative. This design will be important when building the parallel framework in Section 3.3.

The numerical integration scheme necessary for accuracy and robustness of the TLS changes in elements that are crossed by the iso-0 and iso-l_c as the front advances from one load increment to another, e.g., the element l illustrated in Fig. 3.3. In the case of crack propagation in elastic-plastic materials, extra care is required with this procedure since history variables that are stored at integration points have to be transferred from old to new integration schemes. In this chapter, the superconvergent patch recovery (SPR) technique [44, 45] is applied to transfer the plastic strain tensor $\boldsymbol{\varepsilon}^{\mathrm{p}}$ and equivalent plastic strain $\varepsilon^{\mathrm{p}}_{\mathrm{eq}}$, following [34].

In Algorithm 3.5, the equilibrium solution phase is summarized.

Algorithm 3.5 The `EquilModule` algorithm.

Input: the damage distribution $d(\phi)$ and level set field ϕ
Output: the nodal displacements; the new integration scheme; and transferred history variables

```
1: initEnrichment()
2: commEnrich()                          /* Enrichment update for receive nodes */
3: initSPRHistory()
4: û ← û + Δû^0 Δt/Δt^0
5: u, ε^p, ε^p_eq ← solver()
```

3.2.3. FRONT EVOLUTION

The last analysis phase concerns computation of the front velocity. It begins with evaluation of the averaged configurational force along the front. Due to the use of the signed distance function, all points sharing the same curvilinear coordinate s are affected when the front at $(0, s)$ (see Fig. 3.1) experiences a front advance[2]. Thus, movement of the front at $(0, s)$ leads to an increase in damage and, consequently, to energy dissipation in all associated points. Therefore, the energy dissipation per unit length as the front moves a unit distance reads:

$$g(s) = \int_0^l d'(\phi)\, Y(\phi, s) \left(1 - \frac{\phi}{\rho(s)}\right) \mathrm{d}\phi, \tag{3.11}$$

where $d'(\phi) = q'(\phi)$ is the spatial derivative of damage with respect to ϕ, l is the size of the damaged zone $l \in (0, l_c]$, and ρ is the curvature of the iso-0. To evaluate $g(s)$ in a discretized setting, an averaged configurational force $\bar{Y}(s)$ is introduced which is related to $Y(\phi, s)$ through:

$$\int_0^l d'(\phi)\, Y(\phi, s) \left(1 - \frac{\phi}{\rho(s)}\right) \mathrm{d}\phi = \int_0^l d'(\phi)\, \bar{Y}(s) \left(1 - \frac{\phi}{\rho(s)}\right) \mathrm{d}\phi. \tag{3.12}$$

Numerically, this averaged configurational force is discretized with DOFs on the nodes of those elements that are at least partially inside the damaged domain Ω^{d}. Lagrange multipliers are used to weakly enforce the constraint that $\bar{Y}$ is constant in ϕ direction [15]. The following system of equations is obtained [29]:

[2] A curvilinear coordinate system (ϕ, s) is introduced here for the derivation, but in the final equations as implemented, the s-coordinates are not used, or even constructed.

$$\begin{bmatrix} \mathbf{K} & \mathbf{L} \\ \mathbf{L} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \bar{\mathbf{Y}} \\ \mathbf{l} \end{Bmatrix} = \begin{Bmatrix} \mathbf{f}^{\mathrm{Y}} \\ \mathbf{0} \end{Bmatrix}, \tag{3.13}$$

in which $\bar{\mathbf{Y}}$ and $\mathbf{l}$ are vectors with $\bar{Y}$ and Lagrange multiplier DOFs, respectively. The matrices and the right-hand side vector are defined as:

$$K_{ij} = \int_{\Omega^{\mathrm{d}}} d' N_i N_j + \frac{\kappa h^2}{l_c} \frac{\partial N_i}{\partial x_k} \frac{\partial N_j}{\partial x_k} \mathrm{d}\Omega, \tag{3.14}$$

$$L_{ij} = \int_{\Omega^{\mathrm{d}}} l_c \left(\frac{\partial N_i}{\partial x_k} \frac{\partial \phi}{\partial x_k} \right) \left(\frac{\partial N_j}{\partial x_k} \frac{\partial \phi}{\partial x_k} \right) \mathrm{d}\Omega, \quad \text{and} \tag{3.15}$$

$$f_i^{\mathrm{Y}} = \int_{\Omega^{\mathrm{d}}} N_i d' Y \, \mathrm{d}\Omega, \tag{3.16}$$

where N_i and N_j are the shape functions associated with nodes i and j, κ is a stabilization parameter, h is the same parameter used in Eq. (3.5), and Y is the local configurational force which depends on the current elastic strain field through Eq. (3.9).

The material resistance to damage growth Y_c is made into a function of the size of the damaged zone in order to take into account independent input parameters for damage initiation and propagation [29]. In this approach, Y_c varies from an initial strength-based value Y_c^0 to an energy-based value Y_c^G as the size of the damaged zone $\bar{\phi}$ increases. For intermediate values of Y_c, the following interpolation is adopted [20]:

$$\log(Y_c) = \log(Y_c^0) + \frac{\bar{\phi} - \bar{\phi}_{\mathrm{init}}}{\bar{\phi}_{\max} - \bar{\phi}_{\mathrm{init}}} \left(\log(Y_c^G) - \log(Y_c^0) \right), \tag{3.17}$$

where $\bar{\phi}_{\mathrm{init}}$ and $l_c/3 \leq \bar{\phi}_{\max} \leq l_c/2$ are, respectively, the initial size of the damaged zone and the size for which the damaged zone is considered a crack. The quantity Y_c^0 is related to the strength of the material, where we follow earlier work in [34]. The parameter Y_c^G related to propagation is given by $Y_c^G = \frac{G_c}{2Al_c}$, with A being the area under the curve $q(\phi)$ given in Eq. (3.3) and G_c being the fracture energy [15].

As Y_c is not constant on Ω^{d}, due to its dependence on $\bar{\phi}$ (cf. Eq. (3.17)) and in order to account for the general case of multiple materials being used in the same mesh, the resistance is also averaged over the width of the damaged band. Therefore, $\bar{Y}_c$ is computed by solving once more the system of equations in Eq. (3.13) but then with $\bar{Y}_c$ in the right hand side vector instead of Y.

Next, the computation of nodal normal velocity v_n is carried out in two steps. Firstly, v_n is computed at the nodes belonging to $\mathcal{N}_0$ (the set of nodes of those elements that contain the front) based on $\bar{Y}$ and $\bar{Y}_c$ as:

$$v_n = \frac{1}{\eta} \left\langle \frac{\bar{Y}}{\bar{Y}_c} - 1 \right\rangle_+, \tag{3.18}$$

where η is a parameter that can be interpreted as viscous resistance against crack growth. Brackets are used to denote the positivity condition, which reflects the irreversibility of crack growth. Secondly, the nodal velocity is propagated throughout the domain by solving $\nabla\phi \cdot \nabla v_n = 0$ with a fast marching method [16, 40].

Algorithm 3.6 summarizes the operations of this phase.

Algorithm 3.6 The `VelocityModule` algorithm.

Input: the nodal values of $\bar{\phi}$; the elastic strain field; and parameters l_c, h, κ and η
Output: the nodal velocity v_n

1: `Gather(`$\mathbf{K}, \mathbf{L}, \mathbf{f}^{\bar{Y}}, \mathbf{f}^{\bar{Y}_c}$`)` `/* Gather matrices and right-hand side vector for Eq. (3.13) */`
2: $\bar{Y}, \bar{Y}_c \leftarrow$ `solver(` Eq. (3.13) `)`
3: `Scatter(`$\bar{Y}, \bar{Y}_c$`)` `/* Scatter` $\bar{Y}$ `and` $\bar{Y}_c$ `over the processes */`
4: **for all** node $i \in \mathcal{N}_0$ **do**
5: $\quad v_{ni} = \frac{1}{\eta}\left\langle \frac{\bar{Y}_i}{\bar{Y}_{ci}} - 1 \right\rangle_+$
6: **end for**
7: `Gather(`v_n`)` `/* Gather` v_n `at nodes of the front */`
8: `extendVelocity()`
9: `Scatter(`v_n`)` `/* Scatter` v_n `all over the processes */`

3.2.4. SECANT UNLOADING SCHEME

If one wants to let the crack grow under the condition that $\bar{Y}$ cannot exceed $\bar{Y}_c$ instead of with Eq. (3.18), that is possible with an alternative loading scheme under the assumption of secant unloading [13, 15, 29]. This alternative involves solving a unit load analysis in the equilibrium solution and determination of a load scale factor γ after the $\bar{Y}$ corresponding to unit loading is evaluated.

With this approach, there is no adaptive time increment procedure. The velocity is defined such that the maximum is always equal to v_{lim}. Hence, the first loop in Algorithm 3.1 is skipped and the second loop is passed with $\Delta t = 1$ s (cf. Eq. (3.4)). The nucleation check should be performed with the actual load level. The ratio Y/Y_c^0 is therefore scaled by γ^2 (see Eq. (3.19)) in the tenth and seventh lines of Algorithms 3.2 and 3.4, respectively. For simulation without plasticity, Y_c^0 is defined as $Y_c^0 = \frac{f_t^2}{2E}$, where f_t and E are, respectively, the tensile strength and Young's modulus.

Because unit load analysis only makes sense under secant unloading, this loading scheme cannot be used in combination with plasticity in the undamaged part of the material. Therefore, in Algorithm 3.5, the operations in lines three and four are skipped for this loading scheme.

The pseudo-code in Algorithm 3.7 illustrates the front evolution phase with this loading scheme. The strain field and, consequently, the averaged configurational forces $\bar{Y}$ are evaluated with unit load boundary conditions. The actual load level for a time step is then determined by scaling the unit-load solution with a load scale factor γ such that the maximum scaled value for $\bar{Y}$ along the front is equal to $\bar{Y}_c$:

$$\gamma^2 \max_{i \in \mathcal{N}_0} \left\{ \frac{\bar{Y}_i}{\bar{Y}_{ci}} \right\} = 1. \tag{3.19}$$

Finally, the front velocity v_n for every node in $\mathcal{N}_0$ is obtained through [29]:

$$v_n = k \left\langle \frac{c\gamma^2 \bar{Y}}{\bar{Y}_c} - 1 \right\rangle_+ \quad \text{with} \quad k = \frac{v_{\text{lim}}}{c-1}, \tag{3.20}$$

where v_{lim} is the maximum growth the front can experience for a time step. In order to guarantee the numerical stability of the staggered scheme, a value $v_{\text{lim}} = \alpha_n h$ (cf. Eq. (3.5)) is used in this chapter, following [29]. The parameter c influences the spread

of the front movement to nodes with lower values for the ratio $\bar{Y}/\bar{Y}_c$. For $c \to 1$, only the node with the highest value $\bar{Y}/\bar{Y}_c$ undergoes a front advance. On the other hand, for higher values of c, nonzero front movement is found in more nodes.

Algorithm 3.7 The `VelocityModule` algorithm for secant unloading scheme.

Input: the nodal values of $\bar{\phi}$; the elastic strain field and parameters l_c, h, c, k; and $v_{\max}$
Output: the nodal velocity v_n

```
Gather(K, L, f^Y, f^Yc)      /* Gather matrices and right-hand side vector for Eq. (3.13) */
Ȳ, Ȳc ← solver( Eq. (3.13) )
Scatter(Ȳ, Ȳc)                          /* Scatter Ȳ and Ȳc over the processes */
max ← 0                                  /* initialize the highest ratio Ȳ/Ȳc */
for all node i ∈ N0 do
    ratio ← Ȳi / Ȳci
    if ratio > max then
        max ← ratio
    end if
end for
Allreduce(max)                  /* Get the highest ratio Ȳ/Ȳc from all the processes */
γ = sqrt(1/max)                                  /* Compute the load scale factor */
for all node i ∈ N0 do
    v_ni = k⟨ cγ²Ȳi/Ȳci − 1 ⟩+
end for
Gather(v_n)                                  /* Gather vn at nodes of the front */
extendVelocity()
Scatter(v_n)                                /* Scatter vn all over the processes */
```

3.3. PARALLEL VERSION OF THE TLS METHOD

This section deals with the modifications to the sequential algorithm presented above that are needed to parallelize it. This parallelization consists of concurrent *tasks* or *processes*, being executed on distinct processors, in which each task encapsulates a copy of the sequential framework. Tasks are not completely independent; as a result, they need to interact by exchanging data (i.e., sending and receiving data) or *messages*. For the communication between processors, the Message Passing Interface (MPI) [61] communication protocol is used.

3.3.1. DOMAIN DECOMPOSITION

The DD strategy described by Lingen et al. [60] is used in this thesis. In this strategy, the problem is divided in sub-problems that are solved almost independently. Therefore, the original finite element mesh is first partitioned into non-overlapping groups of elements, each one corresponding to one subdomain as exemplified in Fig. 3.4. A task is defined for each subdomain. Both tasks and subdomains are identified by an integer number called *rank* that ranges from 1 to n_p. The n_p subdomains are then optionally extended with overlap regions by assigning extra nodes (and consequently elements) from neighboring subdomains.

The original nodes and elements on one subdomain are called *internal nodes* and *elements*, whereas the additional ones are called *overlapping nodes* and *elements* (see

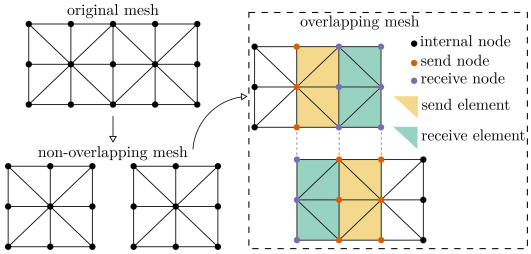


Figure 3.4: An original finite element mesh decomposed into two subdomains. The dashed box on the right indicates the mesh overlapping option.

Fig. 3.4). Observe that some nodes and elements on the overlapping region are also internal quantities; they are referred to as *send nodes* and *elements*. The counterparts of send nodes and elements are named *receive nodes* and *elements*. Although overlapping regions are not necessary in the DD strategy [62], it has been shown that this overlap can improve the convergence rate of parallel iterative algorithms [57].

Nodes and elements are indexed locally. However, the exchange of messages among subdomains is based on unique global indices (IDs), which are assigned during the partitioning of the original mesh [62, 63].

3.3.2. ALLREDUCE, GATHER AND SCATTER ROUTINES

Modifications to achieve a desirable parallelism for the level set update and front evolution analysis phases require communication among tasks. Unlike the equilibrium solution phase, where the majority of communications are point-to-point (i.e., send and receive communication routines) between neighboring subdomains, collective communication routines in the sense that they involve participation of all subdomains or a group of them are proposed for the level set update and front evolution analysis phases. The main idea behind setting up these communication routines is to be minimally intrusive to the operations in Algorithms 3.4, 3.6 and 3.7.

Three general collective routines are used: `Allreduce`, `Gather` and `Scatter`[3]. `Allreduce` is an all-to-all collective routine that is used to compute the maximum value of a quantity from all processes and distributes the result back, as schematically illustrated in Fig. 3.5. `Gather` and `Scatter` are, respectively, designed to collect and spread messages involving a single receiving or originating process, named the *root*. `Gather` is an all-to-one (or a 'some-to-one') collective function in which each process (root process included) sends the contents of their send buffer; the root process receives the messages and stores them in rank order. `Scatter` , on the other hand, is a one-to-all (or a 'one-to-some') collective routine used by the root to send a message, possibly with different sizes, to all processes or a group of them.

Note that `Gather` and `Scatter` are well suited for having operations executed by the root that are mostly sequential in nature and/or have the necessity of using global so-

[3] They have similar functionality as `MPI_Allreduce`, `MPI_Gather` and `MPI_Scatter` functions found in MPI. However, `Allreduce`, `Gather` and `Scatter` might operate on non-default MPI data type. Moreover, `Gather` and `Scatter` might involve only a small group of processes.

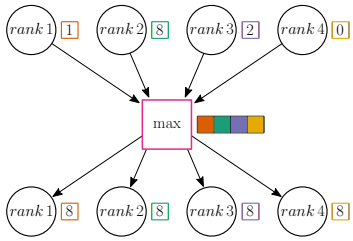


Figure 3.5: The collective routine `Allreduce` is used to compute the global maximum value of a quantity from all the local values $\{1,8,2,0\}$ allocated on different tasks and distribute the result back to all the tasks. Arrows and colored blocks indicate, respectively, the data flow and content.

lution strategies. Depending on the nature of the operation executed by the root, all processes or a subset of process participate in the `Gather` and `Scatter` routines, as illustrated in Fig. 3.6. In all algorithms, operations that are positioned in between a `Gather-Scatter` pair are only executed by the root, and the inputs and outputs necessary for this operation are exchanged between the root and other processes by the `Gather` and `Scatter` routines.

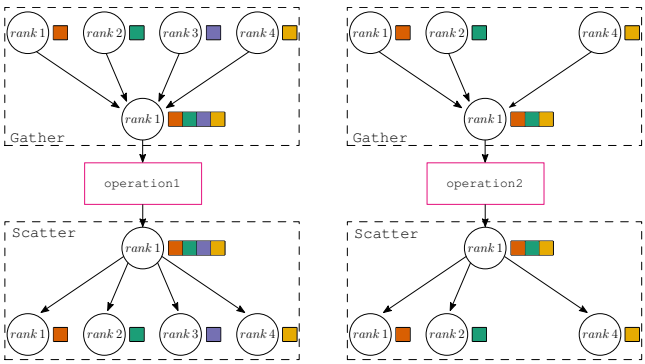


Figure 3.6: The root executes functions `operation1` and `operation2` along with a `Gather -Scatter` execution block. Rank 1 is considered as the root process. In this illustration, all processes contribute to both communications for `operation1`, whereas rank 3 is the only process that does not participate in the communication routines for `operation2`. Arrows and colored blocks indicate, respectively, the data flow and content.

The two fast marching algorithms used for the reinitialization of the level set field ϕ (i.e., the `reinitializeLS` function in Algorithm 3.4) and for the extension of front velocity v_n (i.e., the `extendVelocity` functions in Algorithms 3.6 and 3.7) are performed in combination with the `Gather` and `Scatter` routines, mainly because they use a global sorting of the nodes based on their level set values for determining the order in which the ϕ and v_n are updated [40].

Furthermore, `Gather` and `Scatter` routines are used for the solution of Eq. (3.13) for the averaged quantities $\bar{Y}$, $\bar{Y}_c$ and $\bar{\phi}$ (i.e., the `solver` functions in Algorithms 3.4, 3.6 and 3.7). When solving Eq. (3.13) concurrently, the imposed constraints (i.e., $\bar{Y}$, $\bar{Y}_c$ and

$\bar{\phi}$ are constant in ϕ direction, and $\bar{\phi}$ also in s direction) for these averaged quantities will not be satisfied in a global sense. Note that these quantities are only computed on the damaged domain Ω^{d}. In many cases, not all subdomains will contain damage, which is why some-to-one and one-to-some versions of the `Gather` and `Scatter` routines are relevant.

For the execution of these general `Gather` and `Scatter` communication routines, a data structure management, slightly different from what is already used for the parallel iterative solver, is adopted in which the root has information on the original mesh and each subdomain keeps an operator $\mathbf{R}$ to extract elements from a root vector as $\mathbf{a}_i = \mathbf{R}_i\mathbf{a}$, where $\mathbf{a}_i$ is a vector containing the elements from a root vector $\mathbf{a}$ associated with the i-th subdomain. The root also keeps a collection of operators $\mathbf{Q}_i$ associated with each subdomain for assembly of root vectors as $\mathbf{a} = \sum_{i=1}^{n_p} \mathbf{Q}_i^{\mathrm{T}}\mathbf{a}_i$. Note that n_p not necessarily includes all processes. The operators $\mathbf{R}$ and $\mathbf{Q}$ are non-square boolean matrices, and have similar structures as the right and left restriction operators defined in the parallel iterative solver proposed in [60].

3.3.3. LEVEL SET UPDATE

The additional modifications related to the level set update phase are addressed following the order of the operations shown in Algorithm 3.4. Firstly, `updateLevelSet` (see Algorithm 3.1) is executed. Evaluation of the time increment size with Eq. (3.5) needs the largest value of velocity v_n over the whole mesh. After executing the first loop in parallel, the function `Allreduce` is therefore called, as shown in Algorithm 3.1. `Allreduce` computes the maximum value of $\max\{v_n\}$ from all tasks and distributes the result back to all of them. For those subdomains without damage front, their contributions to this operation are null.

Next, the fast marching algorithm for reinitialization of ϕ is executed by the root only. The `reinitializeLS` function is therefore sandwiched between a `Gather-Scatter` pair where the root first receives the ϕ updated by `updateLevelSet` from each process and sends back the reinitialized values. All processes are involved, similar to `operation1` in Fig. 3.6.

After the reinitialization, `testInitiation` is called in order to concurrently perform the nucleation check, using `Allreduce` to compute the global maximum ratio Y/Y_c^0. For this particular operation, `Allreduce` is designed such that it also returns the corresponding coordinates $\mathbf{x}_{\mathrm{nucl}}$ related to the maximum value because `makeNucleus` needs $\mathbf{x}_{\mathrm{nucl}}$ in all processes to update ϕ concurrently.

Finally, the averaged quantity $\bar{\phi}$ is computed by the root. Again, the function `solver` is positioned in between the `Gather-Scatter` pair. The matrices and right-hand side vector necessary for solving Eq. (3.13) for $\bar{\phi}$ are concurrently partially assembled but only by processes belonging to $\mathscr{P}_{\mathrm{d}}$ (the set of processes that possess any damage front). The root process gathers these partial quantities from the processes in $\mathscr{P}_{\mathrm{d}}$ by means of a `Gather` routine in order to assemble the global system of equations. Once the solution for $\bar{\phi}$ is obtained, the root sends it back to the same set of processes through a `Scatter` routine.

Unlike the `Gather` routine used in the reinitialization operation in which a one-way communication is performed, i.e., the data flows from senders to the root with noth-

ing going back in return, the `Gather` routine considered in the assembly procedure of Eq. (3.13) encompasses three stages of data communication. First, the root queries processes that have a damage front (i.e., processes that form the set $\mathscr{P}_\mathrm{d}$) for global IDs of the nodes belonging to the set of elements completely or partially inside the damaged domain Ω^d. Once the root receives these nodal IDs, the root numbers DOFs at these nodes and sends them back along with the size of the final system of equations to the processes belonging to $\mathscr{P}_\mathrm{d}$ in rank order. Then, each process makes use of this information to assemble its own matrix and right-hand side vector and sends them back to the root. The root then assembles the final system of equations by summing these contributions, as depicted in Fig. 3.7.

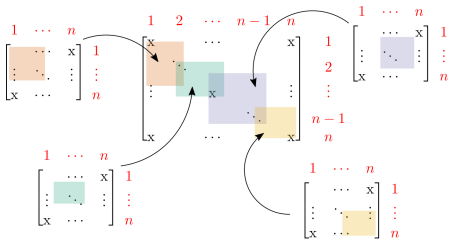


Figure 3.7: Assembly of the global matrix, belonging to $\mathscr{P}_\mathrm{d}$, used by the root to solve the final system of equations. All the small matrices, each belonging to a single process, have the same indexing as the global $n \times n$ matrix.

For sake of consistency and efficiency in terms of message communication, two points of extra attention exist in the aforementioned assembly procedure. First, to avoid duplicated contributions from the elements in the overlapping region (see Fig. 3.8), receive elements are excluded from the assembly procedure. Second, due to the sparse structure of the system of equations, there may be many zero entries which can unnecessarily overload these `Gather` and `Scatter` calls. To make matters worse, the expanded system of equations assembled on a single subdomain may have just a few non-zero elements. Therefore, a sparse data storage structure is necessary. For this purpose, a Compressed Row Storage (CRS) format is adopted [64, 65], which stores $2nnz+n+1$ elements instead of n^2 for an $n \times n$ matrix with nnz nonzero entries. Note that each quantity assembled concurrently by a single process belonging to $\mathscr{P}_\mathrm{d}$ uses the same global indexing from the root. This design allows for efficient summation of the global sparse matrix in the root process from a number of sparse matrices from the other processes.

3.3.4. EQUILIBRIUM SOLUTION

For solving the linear system of equations from the equilibrium problem, the parallel iterative Generalized Minimum Residual (GMRES) solver proposed by Lingen et al. [60] is used. This solver is equipped with a two-level preconditioner that consists of a restricted additive Schwarz preconditioner that acts on the level of subdomains and an algebraic coarse grid preconditioner that operates on the global level. The restricted additive Schwarz preconditioner is based on an incomplete Cholesky decomposition, while the coarse grid preconditioner is constructed in terms of the rigid body modes of the subdomains.

The interaction communication associated to this solver is mostly one-to-one, i.e., involving only the shared region between adjacent subdomains, except when global reduction operations such as a global sum necessary for computing a scalar product. The messages involved in this interaction consist of a vector with a given length and a data type (e.g., integer or double). The length of this vector which is usually proportional to the number of DOFs attached to the nodes (i.e., send and receive nodes) on the shared regions.

PARALLEL ENRICHMENT SCHEME

It is important to recall that as the iso-l_c evolves, DOFs are added to and/or removed from the problem, which changes the dimensions of stiffness matrix and force vectors. If one overlapping node is enriched or unenriched on a subdomain, its counterparts, i.e., the nodes with the same global ID on different subdomains, also have to be enriched or unenriched accordingly. This is necessary since the messages exchanged by the routines of the solver along the boundaries of neighboring subdomains need to be equal in size.

Hence, after performing its own enrichment procedure, each process has to exchange data with its neighbors about the enrichment status of its own overlapping nodes. Note that this communication is only necessary when a crack is propagating across subdomain boundaries. In this communication, the message is a vector with entries of a special data type, which encodes three types of information (or a *triplet*): the global ID of an overlapping node, its enrichment flag (whether or not it is enriched), and element support (positive and negative lists of global element IDs).

The send nodes control this process because they are interior nodes and, hence, always have sufficient information to decide whether the node should be enriched and because every node in the overlapping region is always send node in no more and no less than one subdomain. Thus, when DOFs attached to one send node are updated, receive nodes on the neighboring subdomains will follow, as shown in Fig. 3.8. Note that this is only possible as long as the continuity of ϕ is guaranteed across the boundaries of subdomains, which ensures that all subdomains involved in this communication process share the same geometric location of the iso-l_c on their overlapping regions. Consequently, these subdomains are able to determine the exact length of the message for both send and receive packets of data.

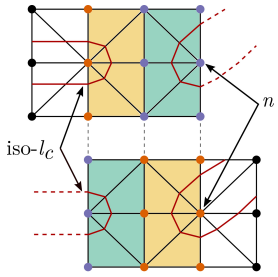


Figure 3.8: The node n is shared by two subdomains and can automatically be identified when the iso-l_c crosses both subdomains. Dashed lines represent the part of the iso-l_c that belongs to the neighboring subdomain.

The parallel enrichment scheme involves two stages. Firstly, each process p executes the function `initEnrichment` (see Algorithms 3.5 and 3.8) for all its nodes with level set value $\phi > l_c$, except for those that are receive nodes. During the execution of `initEnrichment`, the send and receive nodes are collected and their triplets (i.e., global ID, enrichment flag, and element support) are stored in two special data structures sendBuffer and recvBuffer, respectively. Both sendBuffer and recvBuffer store the exact amount of data for each neighbor of process p. At this point, the triplet of each node that is stored in recvBuffer only contains the global node IDs, whereas the triplet of each node that is stored in sendBuffer may be complete. If a send node is unenriched, the element support associated to this node is not stored and, hence, the enrichment flag in the triplet related to this node becomes false.

Secondly, exchange of information between processes is handled. The procedure for sending and receiving sendBuffer and recvBuffer on one process p is executed only if send and/or receive nodes have been collected. First, once data is stored in recvBuffer, process p loops over all its neighbors i calling the function `InitReceive`[4], which initiates a nonblocking receive communication [61]. This function returns a handle (or request recvreq) that can be used at a later time to check whether the message has been received. Note that if p does not have messages to be received from one specific neighbor i, the size of recvBuffer (i.e., recvBuffer[i] .size( ) ≤ 0) is checked and the call of `InitReceive` is then skipped. Because `InitReceive` does not block the calling process, messages can be called in any order without risking 'deadlock' issues. Next, as long as sendBuffer contains any data, the send procedure is executed in a similar way to the receive one. After these two receive and send loops, the functions `WaitAll` are called in order to complete the multiple receive and send requests. Finally, once p has received messages from all its neighbors, it updates the enrichment status for receive nodes by means of the function `setLEnrich`.

Note that `InitReceive` is called as early as possible to increase the chance that a matching call `InitSend` by another process can be completed immediately. This strategy helps to lower communication overhead because each message that cannot be moved directly to a receiver buffer must be temporarily stored in a pending queue [61]. This communication pattern is similar to what was proposed in [62] to deal with enriched nodes in an extended finite element context for hydraulic fracturing in elastic materials.

MAPPING OPERATORS

Regarding the SPR technique for transferring history, it is chosen to let each subdomain deal with its own execution of routine `initSPRHistory`, even though this means that incomplete patches are used for nodes at boundaries of subdomains. In order to be more consistent, an extra communication strategy would be necessary for those patches to be completely assembled. However, this will not have significant effect on the global response.

[4] `InitReceive` and `InitSend` make use of nonblocking functions from MPI, such as `MPI_Irecv` and `MPI_Isend`, as well as `MPI_Waitall` (which is represented here as `WaitAll`) for the completion of communication. A special MPI data type is also designed in order to deal with recvBuffer and sendBuffer data structures.

Algorithm 3.8 The `commEnrich` algorithm for communication of enrichment.

Input: the enrichment status of send nodes
Output: the enrichment update of receive nodes

```
1: if processor p has collected receive nodes then                    /* Receive data */
2:     for all neighbor i of process p do
3:         if recvBuffer[i] .size( ) ≤ 0 then
4:             continue                          /* Skip neighbors without message to be sent */
5:         end if
6:         InitReceive (recvBuffer[i], recvreq[i], i)
7:     end for
8: end if
9: if processor p has collected send nodes then                       /* Send data */
10:    for all neighbor i of process p do
11:        if sendBuffer[i] .size( ) ≤ 0 then
12:            continue                  /* Skip neighbors without message to be received */
13:        end if
14:        InitSend (sendBuffer[i], sendreq[i], i)
15:    end for
16: end if
17: if process p has collected receive nodes then                     /* End receive procedure */
18:    WaitAll (recvreq)
19:    setLEnrich(recvBuffer)                  /* Update enrichment for receive nodes */
20: end if
21: if process p has collected send nodes then                        /* End send procedure */
22:    WaitAll (sendreq)
23: end if
```

Another option would be the inverse distance weighted interpolation scheme, in which all the transferring of history takes place locally on the element level without iteration with its vicinity. However, it was shown in [34] that the SPR technique is more suitable, especially for coarse meshes.

3.3.5. FRONT EVOLUTION

The modifications of the front evolution analysis phase follow those introduced for the parallel version of level set update phase. Two `Gather-Scatter` execution blocks are introduced, as shown in Algorithm 3.6. For solution of Eq. (3.13) for the averaged forces $\bar{Y}$ and $\bar{Y}_c$, the same strategy is adopted as already discussed for $\bar{\phi}$. For the fast marching algorithm, `extendVelocity`, the strategy is the same as for the reinitialization of the level set field.

3.3.6. SECANT UNLOADING SCHEME

The main difference between the two loading schemes considered in this chapter is found in the front evolution phase, as outlined in Section 3.2.4. To parallelize the scheme with secant unloading (see Algorithm 3.7), one additional `Allreduce` call is introduced in order to compute the maximum value of the load scale factor, which is needed for the nucleation check by all subdomains, as shown in Algorithm 3.4.

3.4. RESULTS AND DISCUSSION

The performance of the parallel versions of the TLS model is investigated with numerical examples in this section. The models have been developed within the parallel open source Jem/Jive toolkit [65], which provides graph-based partitioning algorithms for de-

composition of the original mesh [62] and the parallel GMRES solver [63]. The simulations have been run on the Numerical Mechanics Cluster HPC27 which is a regular High Performance Computing master-slave system at Delft University of Technology. Each node is equipped with two Intel Xeon E5-2630 version 4 processors, having 10 cores each, and 128 GB memory.

For all numerical examples, unstructured meshes of linear triangles generated with Gmsh [48] are considered. For nucleation, the size of a new damage nucleus ϕ_0 is about the effective element size h. Regarding the stabilization parameter, two values of κ are used: $\kappa = 1$ for $\bar{Y}$ (Eq. (3.13)) and $\kappa = 1 \cdot 10^4$ for $\bar{\phi}$. All presented results are obtained under plane strain assumptions.

3.4.1. DOUBLY-NOTCHED SQUARE PLATE (DNSP)

As a first example, the response of a doubly-notched square plate (DNSP) [13] is simulated. The material is modeled as elastic, allowing the use of the secant unloading scheme. To investigate the scalability of the parallel approach, the same analysis is performed with different numbers of subdomains, each time using as many cores as there are subdomains. For each number of subdomains, the problem is run three times and the average runtime is computed.

Boundary conditions and geometry of the specimen are shown in Fig. 3.9. Poisson's ratio, Young's modulus, fracture energy, and tensile strength are, respectively, $\nu = 0.3$, $E = 7000\,\text{MPa}$, $G_c = 40\,\text{N/mm}$, and $f_t = 79\,\text{MPa}$. The critical length l_c is equal to 0.8 mm. This example is performed with $c = 2$, $\alpha_n = 0.5$, $\bar{\phi}_{\text{init}} = 0$, and $\bar{\phi}_{\text{max}} = l_c/3$. A region around the notches with refined mesh is defined where the effective element size $h = 0.1\,\text{mm}$. Away from this region, the element size is 0.5 mm, leading to a mesh of 151370 elements and 76136 nodes, i.e., 152272 DOFs. The nucleation check is only performed on elements in the fine mesh region.

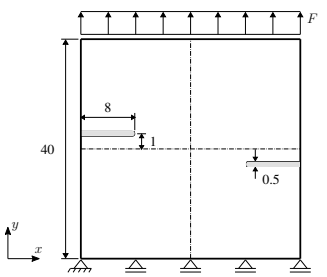


Figure 3.9: DNSP: boundary condition and geometry (dimensions in mm).

Before the number of subdomains is varied, the influence of using an overlapping region on the runtime is investigate. The analysis with 20 subdomains is performed with overlap region ranging from zero to six elements wide. Table 3.1 shows the total runtime for different layers of elements in the overlapping region. It is found that an overlapping region with one layer of elements is optimal for this example.

The load-displacement curve for a reference solution, obtained without parallelism, is compared with the result of using 20 subdomains in Fig. 3.10. Even in the oscillatory post-peak part, the results with 20 cores are exactly the same as those from a single core.

Table 3.1: Total runtime for different sizes of the overlap region.

Overlap layers	Time [s]
0	595.21
1	526.68
2	555.31
4	584.70
6	628.79

By using 20 cores, the parallel framework accelerates the sequential approach by a factor of 13.4 without loss of accuracy. Fig. 3.10 shows for the case of six subdomains that the continuity of the level set field is ensured even when the crack crosses the subdomain boundaries.

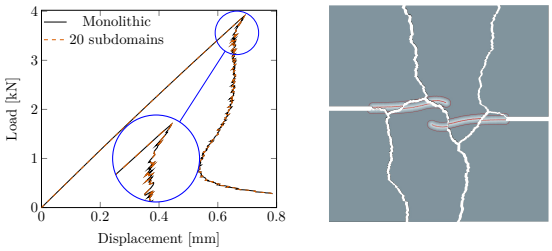


Figure 3.10: DNSP: load-displacement curve (right), and final crack distribution in the mesh partitioned into six subdomains (left).

Fig. 3.11 shows the total runtime as well as the total time spent for each analysis phase (level set update, equilibrium solution, and front evolution) as a function of number of subdomains. As expected, the equilibrium solution phase is the most time consuming. The total time spent on this phase scales very well as the number of subdomains increases with close to the optimum of linear scaling.

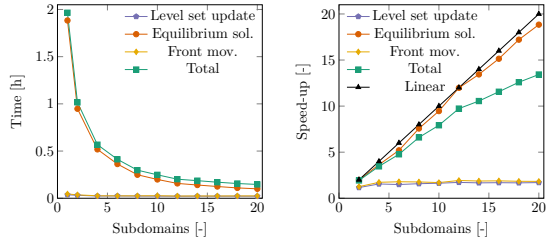


Figure 3.11: DNSP: wall clock time (left) and speed-up (right) graphs.

Unlike the equilibrium solution phase, the level set update and front evolution phases are barely accelerated since their main operations rely on collective communication patterns, yielding a low level of parallelism. Going from one to four subdomains, a speed-up of about a factor of two is obtained, but further increasing the number of subdomains does not lead to significant changes in total time needed for these phases. However, the

total time needed for these phases without parallelism is much less than for the equilibrium solution phase. Therefore, the overall scaling is still very favorable. Nevertheless, the fact that part of the framework is not scaling optimally means that the scaling in total runtime deviates increasingly from optimal linear scaling.

One possible option to achieve better scaling for the level-set-related operations in the level set update and front evolution phases would be to use the weak form, and its discretization, to solve the reinitialization and velocity extension problems following Adams et al. [66] and Dekker et al. [67]. In this approach, we have systems of equations for both problems which can also be solved by the parallel iterative solver. Matrices and right-hand side vectors have assembly procedure similar to standard finite elements. Thus, we could have a better parallelism level, i.e., a more one-to-one communication strategy, instead of a global-like solution strategy, which involves global reduction communication patterns as presented.

3.4.2. SINGLE-NOTCHED SHEAR TEST (SNST)

In the second example, the response of a single-edge notched plate considering plasticity is simulated. Once again, the scalability and accuracy of the parallel framework are assessed by means of load-displacement and speed-up curves considering different numbers of subdomains and cores for the same analysis.

Fig. 3.12 shows the boundary conditions and geometry of the example. A horizontal displacement is applied to the top half of the left edge. The material is modeled with the pressure-dependent plasticity model for polymers by Melro et al. [37] as revised by Van der Meer [39]. This plasticity model makes use of a paraboloidal yield surface that takes into account different compressive and tensile yield stresses. Young's modulus, Poisson's ratio, and fracture energy are, respectively, $E = 3760\,\text{MPa}$, $\nu = 0.3$, and $G_c = 0.9\,\text{N/mm}$ [37, 39]. The other properties of the material, such as hardening curves, plastic Poisson ratio, and the ultimate yield stress values for the nucleation check, are the same as in [39]. The critical length l_c is equal to 2 mm. This example is performed with $\alpha_n = 0.5$, $\bar{\phi}_{\text{init}} = 0$, and $\bar{\phi}_{\text{max}} = l_c/3$. The whole geometry is meshed with $h = 0.4\,\text{mm}$, leading to 157600 elements and 79419 nodes or 158838 DOFs. The nucleation check is only performed on a region near the notch tip. The viscous parameter against grack growth and displacement rate are, respectively, $\eta = 5\,\text{s}\,\text{mm}^{-1}$ and $\dot{u} = \Delta u^0/\Delta t^0 = 0.05\,\text{mm}\,\text{s}^{-1}$.

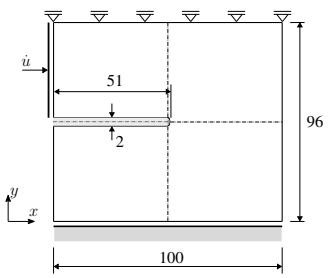


Figure 3.12: SNST: boundary condition and geometry (dimensions in mm).

Fig. 3.13 depicts the comparison between the reference solution obtained with the

sequential framework and the result of using 20 subdomains. Again, there is no loss of accuracy in the sense that both responses are in excellent agreement. The level set continuity across the subdomain boundaries is also preserved, as illustrated in Fig. 3.13 for the case of 20 subdomains. With 20 subdomains, the parallel framework accelerates the sequential approach by a factor of 13.2, as shown in Fig. 3.14.

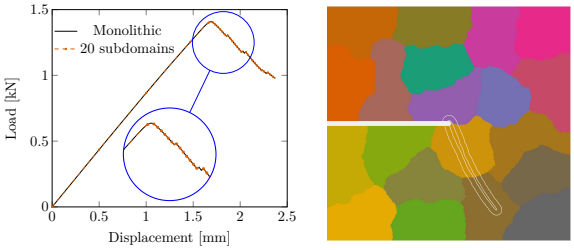


Figure 3.13: SNST: load-displacement curve (right) and final crack distribution in the mesh partitioned into its 20 subdomains (left). Shading indicates the subdivision into its 20 subdomains.

The total runtime, as well as the total time spent for each analysis phase, and their corresponding speed-ups are given in Fig. 3.14. The same trends presented for the previous example with secant unloading scheme are also observed for this example with plasticity. The equilibrium solution phase is the most time consuming and presents the best scalability among the three analysis phases, although the scalability of the equilibrium solution phase is not as good as in the previous example. The level set update and front evolution phases, again, do not scale as well as the equilibrium solution phase, but for this example they have an even smaller contribution to the total runtime.

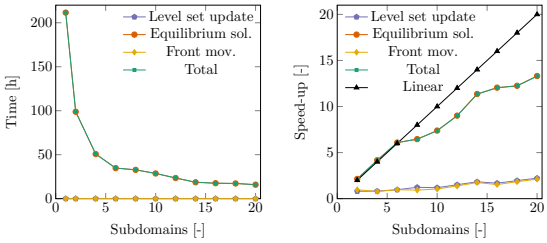


Figure 3.14: SNST: wall clock time (left) and speed-up (right) graphs.

Fig. 3.15 shows the percentage of the equilibrium solution phase to the total runtime for two cases discussed so far. Despite having slightly different mesh sizes, it is noteworthy that the equilibrium solution phase is even more dominant for the plasticity case, reinforcing the importance of a good parallel strategy for that particular phase. However, when comparing speed-up of the equilibrium solution phase to that from simulation without plasticity (Fig. 3.14 versus Fig. 3.11), it can be observed that the additional nonlinearity reduces the scalability of the parallel iterative solver. As a result, overall speed-ups are similar for the two cases.

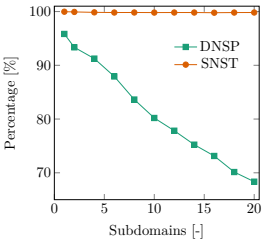


Figure 3.15: Percentage of the equilibrium solution phase to the total runtime for the first two examples.

3.4.3. THREE-POINT BEND END-NOTCHED FLEXURE (3ENF) TEST CASE

The final example is chosen as a case where computation time without parallel approach would become prohibitively long. The case is inspired by experimental observations of cusp crack patterns taking place in resin-rich regions of composite materials in mode II loading conditions [54, 68]. This process begins with an array of inclined cracks perpendicular to the maximum principal stress, which eventually merge to form a single crack on a higher level of observation. Therefore, this example requires the proposed parallel framework to deal with several damage nuclei arising on various subdomains that grow and join up in merging and branching events. Moreover, in realistic simulation of this process, nucleation and growth of the crack should take place in a medium with hardening plasticity. The three-point bend end-notched flexure (3ENF) setup is adopted (see Fig. 3.16). Because the cusp formation takes place in a narrow area close to the notch tip, a very fine discretization is needed near in this area, which gives rise to a large system of equations that would be prohibitive to be solved with a direct solver or with a single computer core.

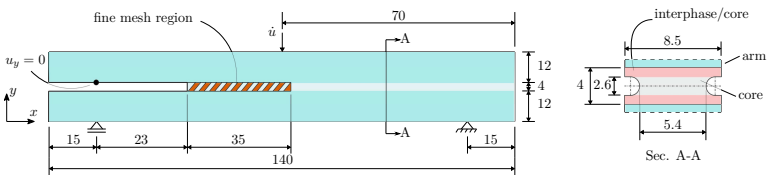


Figure 3.16: 3ENF: boundary condition and geometry (dimensions in mm).

The geometry consists of two stiff arms and one weak core as schematically illustrated in Fig. 3.16. The round cross-section is inspired by rail shear test in [26]. Note that the top arm is also supported in y direction in order to avoid interpenetration without having to model contact between the arms. The two faces are considered as linear elastic materials whose properties are $E = 200\,\text{GPa}$, $\nu = 0.33$, $G_c = 9\,\text{N/mm}$, and $f_t = 960\,\text{MPa}$. The core is modeled with the same plasticity model and material properties mentioned in the second example.

When the crack reaches the interface between core and faces, the crack cannot grow in pure mode I and the constitutive law in Eq. (3.6) leads to stress transfer across the crack. As a result, an artificial hardening is found, as reported by Van der Meer and Sluys [29]. In order to circumvent this undesirable behavior, the interphase constitutive law

introduced in [29] is adopted in a band next to the material interface as indicated in Fig. 3.16. This constitutive law only takes into account stiffness recovery on the strain component normal to the plane that defines the interface, which, in this case, is the strain component in y direction.

The region where the nucleation check is performed is indicated in Fig. 3.16. The smallest element size h is 0.1 mm for the nucleation check region. For the other region of the core and faces, the mesh has element size of 0.15 mm and 0.45 mm, respectively. The mesh therefore has 229919 elements, 115527 nodes, and, initially, 231054 DOFs. The critical length l_c is equal to 0.6 mm. The distance between a new damage nucleus and existing damage front is set to be $\phi_{\text{spacing}} = 4\,\text{mm}$. The viscous parameter against crack growth and displacement rate are, respectively, $\eta = 5\,\text{s}\,\text{mm}^{-1}$ and $\dot{u} = \Delta u^0/\Delta t^0 = 0.01\,\text{mm}\,\text{s}^{-1}$. This simulation is performed with $\alpha_n = 0.2$, $\bar{\phi}_{\text{init}} = 0.1$, and $\bar{\phi}_{\text{max}} = 0.36$.

Fig. 3.17 shows the load-displacement curves for different number of subdomains. The original mesh is divided into 20, 30, and 40 subdomains. The whole analysis takes 74.35 h by using 40 subdomains, whereas the models subdivided into 30 and 20 take 86.54 h and 118.94 h, respectively. Observe that all solutions are in excellent agreement.

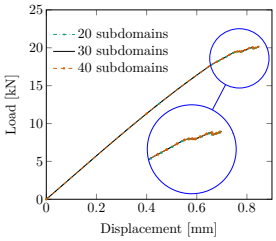


Figure 3.17: 3ENF: load-displacement curve.

Fig. 3.18 presents the damage distribution for the model subdivided into 30 subdomains. Observe that damage initiation takes place in different subdomains and some initial fronts evolve crossing multiple subdomains where the compatibility of ϕ is guaranteed on shared regions.

3.5. CONCLUSIONS

In this chapter, a parallel framework is proposed for the Thick Level Set method. Two TLS models have been adopted: one considering secant unloading loading scheme [29] and the other one for ductile fracture [34]. The parallel iterative solver by Lingen et al. [60] equipped with a DD scheme is used for the equilibrium solution stage. Profiting from the adopted DD scheme, collective communication strategies have been introduced in order to deal with the level set information and the computation of averaged quantities. Moreover, a special data type and communication pattern have been developed to handle enriched nodes belonging to shared regions in the mesh.

In three examples, a successful parallel implementation has been shown. The quality of the results is not influenced by the number of subdomains. The results show that it is possible to apply the parallel framework to different variations of the TLS to benefit from

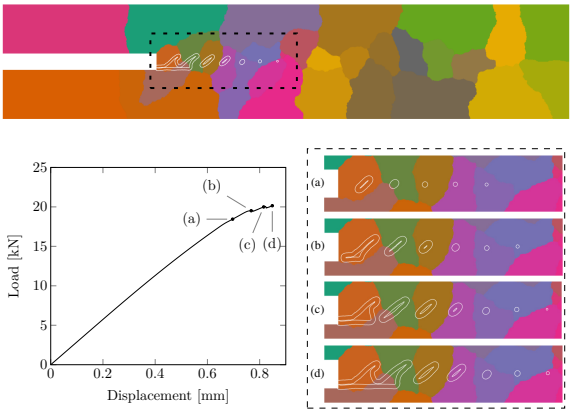


Figure 3.18: 3ENF: final crack distribution, load-displacement graph, and crack evolution (close-up) located ahead of the notch tip considering the mesh partitioned into 30 subdomains. The colored set of elements represents the 30 subdomains.

parallel computing power.

Near-ideal speed-ups are obtained for the equilibrium solution phase, which is the most time demanding part of the TLS in terms of computational cost. For the level set update and front evolution phases, speed-ups remained limited. However, because these phases are less demanding, they did not become real bottleneck for the investigate number of cores. Adding plasticity makes the equilibrium solution phase more dominant, but the added nonlinearity also reduces the speed-up of that particular phase such that overall speed-ups with and without plasticity are comparable. Altogether, substantial speed-ups have been achieved using a moderate amount of cores which decreased the total computational time significantly. This is a necessary improvement in order to use the TLS models for fracture analysis in large-scale problems, as exemplified with the simulation of shear cusps at the notch tip in mode II loading conditions.

4

SKELETON CURVE AND PHANTOM NODE METHOD FOR THE THICK LEVEL SET APPROACH TO FRACTURE

4.1. INTRODUCTION

The Thick Level Set (TLS) method, first introduced by Moës et al. [13], is a non-local damage model that couples damage and fracture mechanics within a single regularized framework. In this method, the damage variable that describes continuum stiffness degradation is made into a function of a level set field. The level set method with a signed distance function is used to construct the level set field in which its the zero level set is used to keep track of the damage front. The level set method gives the TLS method the natural ability of representing complex crack events, such as branching and merging [16, 17]. In the TLS, the damage variable gradually varies over a thick band of the material located behind the damage front until fully degraded regions arise. The presence of a characteristic length represented by the width of this damaging band gives the TLS a non-local nature that prevents spurious localization in the strain field. The update of damage is related to the averaged configurational force, which is obtained by integrating the local values of energy release rate over this characteristic length.

In the original method (the TLSV1), stress-free macro-cracks are determined by zones where the level set value is greater than the critical length, and damage is equal to one. In a domain discretized with regular finite elements, these zones need to be at least one row of elements wide to represent this stress-free state; as a result, mesh dependency might be present [29]. In order to circumvent this issue, Bernard et al. [15] proposed an

Apart from minor changes to its introduction and conclusion sections, this chapter was integrally extract from L. A. T. Mororó, A. Poot, and F. P. van der Meer. *Engineering Fracture Mechanics* 268 (2022) [69].

enrichment strategy for those elements that are cut by the iso-critical curve of the level set field, which allows for a discontinuity in the strain field across such iso-critical curve.

One application where the robustness of the TLSV1 has been a significant advantage is the simulation of cusp formation in resin-rich regions of composite materials loaded in mode II loading conditions [29, 34, 52]. For this particular problem, Van der Meer and Sluys [29] showed the necessity of using an asymmetric constitutive law with different behavior under tension and compression in order to avoid unrealistic 'X-shaped' crack configuration in the material loaded in shear. On the other hand, they also found that this constitutive model leads to unrealistic stiffness recovery and stress transfer across the crack along material interfaces. In order to allow for traction-free sliding deformation, Van der Meer and Sluys [29] proposed a special interphase constitutive law that prevents stiffness recovery on the direction of the material interface.

The TLS has been compared with alternative approaches, such as the phase-field method [24]. It is important to emphasize that the same issues found in the shear test case by Van der Meer and Sluys [29] with symmetric and asymmetric constitutive models may be present in simulations with phase-field models equipped with the same constitutive models.

More recently, a new version of the TLS, referred to as TLSV2 hereafter, has been proposed by Lé et al. [1]. The main objective of this method is to couple both continuum damage modeling and cohesive zone modeling within a single framework and, consequently, profit from the advantages of both: the directionality for crack propagation as well as crack branching and merging ability of the former, and the capacity to model discrete cracks of the latter. Building on the basic premise of the TLS, both damage variables are described by the same level set field. An advantage that has been highlighted by Lé et al. [1] with respect to the TLSV1 is the possibility to introduce complex interfacial behavior at the crack faces, such as frictional contact. To this we would like to add the possibility to describe free sliding deformation without need to include information of the orientation of a nearby material interface in the constitutive relation.

The TLSV2 method evaluates the cohesive forces and displacement jump on the so-called skeleton curve, in which its representation is dictated by the level set field. It is important to emphasize that Lé et al. [1] only investigated test cases with trivial skeleton curves at *a priori* known locations. A general implementation of the TLSV2 requires the extraction of the skeleton curve from an arbitrary level set field with corresponding damage front.

The objective of this chapter is to introduce a more general framework for the TLSV2 method. The main requirements that are addressed are the extraction of free-form skeleton curves and the construction of a discontinuity in the displacement field at the position of that skeleton as the analysis progresses. The concepts of the original paper on TLSV2 by Lé et al. [1] are taken as starting point. To determine the location of skeleton curve, an algorithm based on a combination of ball-shrinking [70, 71] and graph-based algorithms is designed. The resulting skeleton curve is mapped onto the finite element mesh in order to define the cohesive segments, by determining intersection points between the skeleton curve and finite element edges. Subsequently, the cohesive segments are used to define overlapping elements that are used in a phantom node approach [4, 72] in order to evaluate the cohesive contribution of the TLSV2. An *ad hoc* approach to

compute the averaged values of local quantities related to crack growth is introduced. In addition, a generalization of two constitutive models for the bulk is introduced to profit from the advantages of both.

This chapter is organized as follows. In Section 4.2, the main concepts of the TLSV2 method as proposed by Lé et al. [1] are outlined, highlighting the differences with the TLSV1. Section 4.3 is devoted to the algorithm for obtaining the skeleton curve for a given level set field. In Section 4.4, the main features of the phantom node method are outlined. Several numerical examples are presented in Section 4.5 to assess the accuracy of the proposed model and to demonstrate its ability to deal with various crack growth scenarios. Finally, conclusions are presented in Section 4.6.

4.2. THE THICK LEVEL SET V2 METHOD

This section is dedicated to outlining the main features of the TLSV2 method. As in the TLSV1 method, the location of the damage front Γ_0 is tracked as the zero level set (or the 'iso-0') of an auxiliary field $\phi(\mathbf{x})$, the level set field, which is constructed on the entire domain Ω as the signed distance function to Γ_0, such that the level set field satisfies the eikonal equation [16]:

$$\| \nabla\phi \| = 1 \quad \text{on} \quad \Omega. \tag{4.1}$$

On a discretized finite element domain, the definition of ϕ at a given point $\mathbf{x}$ is determined by interpolating the values of ϕ from nodes to $\mathbf{x}$ using finite element shape functions.

The TLSV2 implementation in this chapter inherits the main framework that has been adopted in earlier TLSV1 models [1, 15, 29, 34, 52]: a staggered solution scheme in which displacements and damage are computed separately, as schematically depicted in Fig. 4.1. More precisely, it inherits most of the sequential version detailed in [52] where every time step consists of three main modules (or *analysis phases*, as coined in [52]): `LSModule`, `EquilModule`, and `VelocityModule`. In Fig. 4.1, the highlighted functions indicate the new operations related to the TLSV2 method. For more background on the remaining functions, the reader is referred to [52] or to Chapter 3.

The global solution scheme is described as follows. Firstly, the level set field is updated. As a new task in the `LSModule`, the skeleton curve is determined from the position of the damage front through the `skeletonizer` function. Subsequently, overlapping nodes and elements are introduced in order to accommodate at a later time the phantom node method, for which `updateMesh` is responsible. Then, ϕ is reinitialized, evaluation of damage initiation given the elastic strain field $\boldsymbol{\varepsilon}$ is performed, possibly leading to insertion of a new damage nucleus, and the size of damaged zones, $\bar{\phi}$, is computed, for which the `computePhiBar` function is responsible.

Next, with a given damage distribution associated with ϕ (which could initially consist of negative values throughout Ω to represent an undamaged specimen), the displacements and, consequently, strains and stresses are computed in a standard finite element analysis performed in the `EquilModule`. In this analysis phase, the `IntSchemeUpdate` function (see Fig. 4.1) is responsible for defining displacement degrees of freedom to the new nodes that has been created in the `updateMesh` function and for

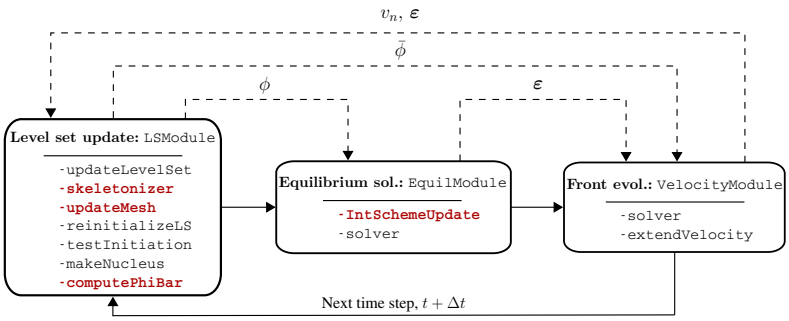


Figure 4.1: The global sequential staggered solution scheme for a single time step. Dashed arrows represent the data exchange among the three modules (cf. [52]).

updating the integration scheme by means of a subtriangulation scheme for those elements that contain the iso-0 or skeleton curves.

Finally, the displacement field and $\bar{\phi}$ are used to compute the configurational force. In this part, the actual load for a time step is determined by scaling the unit-load solution with a load factor, γ, such that the maximum scaled value for the non-local averaged configurational force, $\bar{Y}$, equals the averaged material resistance to damage growth, $\bar{Y}_c$, at one point along the front without exceeding it anywhere. With $\bar{Y}$ and $\bar{Y}_c$ along the front, the front velocity, v_n, is computed, which is subsequently extended throughout Ω by means of a fast marching algorithm. Knowing v_n everywhere, the resulting new ϕ can be obtained and used for the next time step.

Apart from the phantom node method in the equilibrium solution phase, the two functions `reinitializeLS` and `extendVelocity` in `LSModule` and `VelocityModule` that are, respectively, responsible for the reinitialization of ϕ and front velocity extension, as well as the two functions `computePhiBar` and `solver` in the same modules that are, respectively, responsible for the computation of $\bar{\phi}$, and the non-local fields ($\bar{Y}$ and $\bar{Y}_c$), are performed on the mesh that has been updated by the `updateMesh` function.

4.2.1. DAMAGE DEFINITION

In the TLSV2, there are two damage variables instead of one, introducing an interfacial damage variable d next to the bulk damage variable D from the TLSV1. In line with the TLS concept, both damage variables are defined as functions of a unique level set field ϕ, as illustrated in Fig. 4.2. Both damage variables are constrained to follow user-defined profiles within material layers with distinct widths and bounds. As such, the TLSV2 couples both bulk and cohesive damage models within a single framework.

Unlike the TLSV1 method, where D varies from zero to one as ϕ goes from zero to l_c, D in the TLSV2 no longer reaches one at $\phi = l_c$. Mathematically, the continuum damage variable is expressed by:

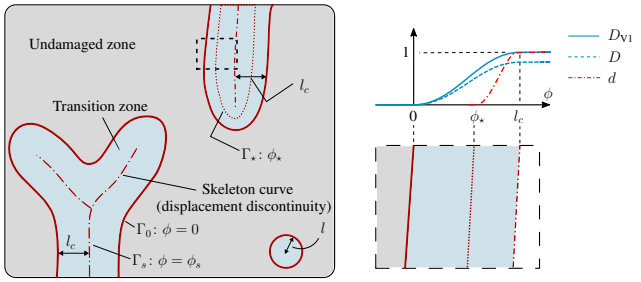


Figure 4.2: The TLSV2, as the TLSV1, makes use of a single level set function to describe multiple damaged zones. As illustrated on the right, the damage variables D and d related to the bulk and interfacial models are functions of ϕ and ϕ_s, respectively. For comparison, the damage variable associated to the TLSV1, D_{V1}, is shown.

$$D(\phi) = \begin{cases} 0, & \phi \leq 0 \\ Q(\phi), & 0 < \phi \leq l_c \\ Q(l_c), & \phi > l_c \end{cases}, \tag{4.2}$$

where Q is a function that has the properties of $Q(0) = 0$, $Q(l_c) < 1$ and $Q' \geq 0$. As D is always lower than one, stress-free deformation in the bulk is not possible. In the middle of the damage band, i.e., on the skeleton curve of the level set field, a displacement discontinuity is included, with cohesive tractions acting across it related to a second damage variable, d. The cohesive damage variable depends also on ϕ, more precisely on ϕ evaluated at the skeleton curve, Γ_s, $\phi_s = \phi(\mathbf{x}_s)$, with $\mathbf{x}_s$ being a point on Γ_s. The following set of equations is used for the relation between d and ϕ_s:

$$d(\phi_s) = \begin{cases} 0, & \phi_s \leq \phi_\star \\ q(\phi_s), & \phi_\star < \phi_s \leq l_c \\ 1, & \phi_s > l_c \end{cases}, \tag{4.3}$$

where q is a function that has the properties of $q(\phi_\star) = 0$, $q(l_c) = 1$ and $q' \geq 0$, and $\phi_\star$ is a user-defined constant. Observe that d does reach one at $\phi_s = l_c$ to allow for a fully degraded material.

Note that the cohesive zone model in the TLSV2 method changes in three stages as ϕ (and consequently ϕ_s) evolves. When $\phi_s < \phi_\star$ and $\phi > 0$, the TLSV2 behaves equivalently to the TLSV1, hence, $D \geq 0$, $d = 0$, and there is no cohesive forces acting in the system. The displacement discontinuity with cohesive tractions arises at $\mathbf{x}_s$, and d kicks in when $\phi_s > \phi_\star$. Finally, the crack is traction-free when $\phi_s = l_c$, i.e., when the damage band has a width of $2l_c$.

In order to fulfil the conditions in Eqs. (4.2) and (4.3), the following equations for Q and q are, respectively:

$$Q(\phi) = \eta f(\phi) \tag{4.4}$$

and

$$q(\phi_s - \phi_\star) = f(\phi_s - \phi_\star), \tag{4.5}$$

where f is an arc-tangent formula given by [15, 29, 34]:

$$f(\phi) = c_2 \arctan\left(c_1\left(\frac{\phi}{l_c} - c_3\right)\right) + c_4, \tag{4.6}$$

with $c_1 = 10$ and $c_3 = 0.5$ and the other coefficients given by $c_4 = -c_2 \arctan(-c_1 c_3)$ and $c_2 = (\arctan(c_1(1-c_3)) - \arctan(-c_1 c_3))^{-1}$ [15]. The user-defined parameter $0 < \eta < 1$ in Eq. (4.4) defines the value for $Q(l_c)$ in Eq. (4.2). Observe that q in Eq. (4.5) is shifted to the right on the horizontal axis by making the argument of f equal to $(\phi_s - \phi_\star)$ (see Fig. 4.2). Note that Lé et al. [1] used a different strategy for obtaining Q and q based on a 1D model.

4.2.2. EQUILIBRIUM PROBLEM

For the present study, the following potential energy definition is adopted:

$$E(\mathbf{u}, \phi) = \int_\Omega \Psi(\boldsymbol{\varepsilon}(\mathbf{u}), D(\phi))\, \mathrm{d}\Omega + \int_{\Gamma_s} \psi(\llbracket \mathbf{u} \rrbracket, d(\phi_s))\, \mathrm{d}\Gamma_s - \int_{\Gamma_N} \mathbf{t}_N \cdot \mathbf{u}\, \mathrm{d}\Gamma_N, \tag{4.7}$$

where $\boldsymbol{\varepsilon}$ is the elastic tensor defined as the symmetric part of the gradient of the displacement field $\mathbf{u}$, Γ_N is the surface on which Neumann boundary conditions are considered, and $\llbracket \mathbf{u} \rrbracket$ is the displacement jump over the crack surface Γ_s.

Following the previous TLS models [13, 15, 29], the free energy for the bulk part is adopted:

$$\Psi(\boldsymbol{\varepsilon}, D) = \mu(1 - \alpha_i D)(\varepsilon_i)^2 + \frac{\lambda}{2}(1 - \alpha_v D)\mathrm{tr}(\boldsymbol{\varepsilon})^2, \tag{4.8}$$

where λ and μ are Lamé's elastic constants, ε_i the principal strain values and $\mathrm{tr}(\boldsymbol{\varepsilon})$ the trace of $\boldsymbol{\varepsilon}$, and

$$\alpha_i = \begin{cases} 1, & \varepsilon_i > 0 \\ \beta, & \varepsilon_i < 0 \end{cases}, \quad \text{and} \quad \alpha_v = \begin{cases} 1, & \mathrm{tr}(\boldsymbol{\varepsilon}) > 0 \\ \beta, & \mathrm{tr}(\boldsymbol{\varepsilon}) < 0 \end{cases}, \tag{4.9}$$

with $0 \leq \beta \leq 1$ being a user-defined parameter that allows for equal stiffness loss under tension and compression $(\beta = 1)$, full stiffness recovery under compression $(\beta = 0)$ or anything in between. The influence of β on the global response of a shear test will be assessed in Section 4.5.3.

With Eq. (4.8), the stress-strain relation in principal stress space and the driving force for damage growth can respectively be expressed as:

$$\sigma_i = \frac{\partial \Psi}{\partial \varepsilon_i} = 2\mu(1 - \alpha_i D)\varepsilon_i + \lambda(1 - \alpha_v D)\mathrm{tr}(\boldsymbol{\varepsilon}) \tag{4.10}$$

and

$$Y = -\frac{\partial \Psi}{\partial D} = \mu \alpha_i (\varepsilon_i)^2 + \frac{\lambda}{2}\alpha_v \mathrm{tr}(\boldsymbol{\varepsilon})^2. \tag{4.11}$$

Regarding the cohesive part, the free energy based on the displacement jump expressed in the local frame (n,s) defined on the crack face with normal and shear components (see Fig. 4.12), $[\![\bar{\mathbf{u}}]\!] = \{[\![\bar{u}]\!]_n; [\![\bar{u}]\!]_s\}^{\mathrm{T}}$, is considered [73]:

$$\psi([\![\bar{\mathbf{u}}]\!], d) = (1 - \bar{\alpha}_{ni} d) \frac{1}{2} K [\![\bar{u}]\!]_i^2, \tag{4.12}$$

where the positive constant K is a penalty stiffness, and $\bar{\alpha}_n$ is defined as:

$$\bar{\alpha}_{ni} = \begin{cases} 1, & [\![\bar{u}]\!]_n > 0 \\ (1 - \delta_{ni}), & [\![\bar{u}]\!]_n < 0 \end{cases} \tag{4.13}$$

so that the possibility of interpenetration between crack faces is prevented by contact along the n-axis, where δ_{ij} is the Kronecker delta. Hence, the state equations associated with the cohesive model become:

$$\bar{t}_i = \frac{\partial \psi}{\partial [\![\bar{u}]\!]_i} = (1 - \bar{\alpha}_{ni} d) K [\![\bar{u}]\!]_i, \tag{4.14}$$

and

$$y = -\frac{\partial \psi}{\partial d} = \frac{1}{2} \bar{\alpha}_{ni} K [\![\bar{u}]\!]_i^2. \tag{4.15}$$

4.2.3. CONFIGURATIONAL FORCE

The non-locality of the TLS method appears when a portion δs of the damage front, Γ_0, moves outwards of a distance $\delta\phi$ [15] (see Fig. 4.3). Because of the signed distance function, all the points having the same curvilinear coordinate s[1] are affected when Γ_0 experiences this front advance. The equation that expresses the amount of dissipated energy as the front moves of $\delta\phi$ on δs can be obtained by differentiating Eq. (4.7) with respect to $\delta\phi$ (cf. [13, 15]):

$$\delta E = -\int_\Omega Y D'(\phi) \delta\phi \, \mathrm{d}\Omega - \int_{\Gamma_s} y d'(\phi_s) \delta\phi_s \, \mathrm{d}\Gamma_s, \tag{4.16}$$

where D' is the derivative of D with respect to ϕ, and d' is the derivative of d with respect to ϕ_s.

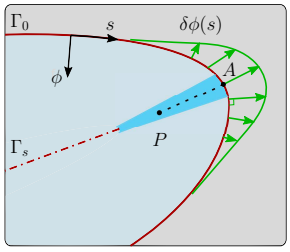


Figure 4.3: The curvilinear coordinate system (ϕ, s). Point P at (ϕ_P, s) is affected as point A at $(0, s)$ on the front experiences a front advance.

[1] The change in variable is obtained by means of $\mathrm{d}\Omega = \left(1 - \frac{\phi}{\rho(s)}\right) \mathrm{d}\phi \, \mathrm{d}s$, following Moës *et al.* [13]. However, in the final equations as implemented, the curvilinear system are not used, or even constructed.

Following Bernard et al. [15], we introduce an averaged quantity $\bar{Y}$ to construct a discretized measure for δE as a function of ϕ. Here, the cohesive part of the TLSV2 is added to the formulation, such that $\bar{Y}$ is the solution of the problem:

$$\int_{\Omega^{\mathrm{d}}} \bar{Y} D'(\phi) \hat{\bar{Y}} \, \mathrm{d}\Omega = \int_{\Omega^{\mathrm{d}}} Y D'(\phi) \hat{\bar{Y}} \, \mathrm{d}\Omega + \int_{\Gamma_s} y d'(\phi_s) \hat{\bar{Y}} \, \mathrm{d}\Gamma \quad \forall \hat{\bar{Y}} \in \bar{\mathscr{Y}}, \tag{4.17}$$

with $\bar{\mathscr{Y}}$ being the space of constant fields along the gradient of ϕ and along the coordinate s in the damaged domain Ω^{d}, i.e., in the region where $\phi > 0$.

This problem is discretized as a field on the nodes of those elements that are at least partially inside Ω^{d} with $\bar{Y}$ as unknown. The constraint that $\bar{Y}$ is constant along the level set gradient, i.e., $\nabla \bar{Y} \cdot \nabla \phi = 0$, is weakly enforced with Lagrange multipliers. Galerkin's method is employed to find nodal values of $\bar{Y}$ from the discretized version of Eq. (4.17) giving rise to the following system of equation:

$$\begin{bmatrix} \mathbf{K} & \mathbf{L} \\ \mathbf{L} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \bar{\mathbf{Y}} \\ \mathbf{l} \end{Bmatrix} = \begin{Bmatrix} \mathbf{f}^{\mathrm{Y}} \\ \mathbf{0} \end{Bmatrix}, \tag{4.18}$$

in which $\bar{\mathbf{Y}}$ and $\mathbf{l}$ are vectors with $\bar{Y}$ and Lagrange multiplier degrees of freedom, respectively. The matrices and the right-hand side vector are defined as (cf. [29]):

$$K_{ij} = \int_{\Omega^{\mathrm{d}}} D' N_i N_j + \frac{\kappa h^2}{l_c} \frac{\partial N_i}{\partial x_k} \frac{\partial N_j}{\partial x_k} \, \mathrm{d}\Omega, \tag{4.19}$$

$$L_{ij} = \int_{\Omega^{\mathrm{d}}} l_c \left(\frac{\partial N_i}{\partial x_k} \frac{\partial \phi}{\partial x_k} \right) \left(\frac{\partial N_j}{\partial x_k} \frac{\partial \phi}{\partial x_k} \right) \mathrm{d}\Omega, \quad \text{and} \tag{4.20}$$

$$f_i^{\mathrm{Y}} = \int_{\Omega^{\mathrm{d}}} N_i D' Y \, \mathrm{d}\Omega + \int_{\Gamma_s} N_i d' y \, \mathrm{d}\Gamma, \tag{4.21}$$

where N_i and N_j are the shape functions associated with nodes i and j, κ is a stabilization parameter, h is the length of the diagonal of the smallest possible rectangle around an individual element in the whole mesh, Y is the local configurational force which depends on the current elastic strain field through Eq. (4.11), and y is the configurational force related to the interfacial model evaluated through Eq. (4.15). It is interesting to note that this system of equation keeps the same solution procedure already found in several TLS-based models [15, 20, 29, 34] simply adding the interfacial contribution in Eq. (4.21). This does come at the cost of losing the direct relation between $\bar{Y}$ and the energy released per unit crack growth as existed in the TLSV1. Observe that Lé et al. [1] made use of a different strategy to compute non-local fields; instead, they followed the approach based on approximation functions (modes), as explained in [35].

Finally, it is remarked that Eq. (4.16) as measure for total energy dissipation is only valid when it is assumed that no energy is dissipated where the displacement discontinuity is introduced, i.e., that the displacement jump is zero at $\phi_\star$. This is only accurate if K is chosen sufficiently high to mimic initially rigid behavior. However, very high K may affect how y is distributed along the cohesive crack [20, 22]. Perhaps, an initially rigid formulation [72, 74] can be adopted, but this is considered out of scope for the present investigation.

4.2.4. FRONT MOVEMENT

The front propagation criterion at a given point at the front is obtained by comparing the averaged configurational force $\bar{Y}$ with $\bar{Y}_c$, the averaged value of a resistance parameter against the damage growth, Y_c. For the general case where Y_c can be a function of spatial coordinates, the averaged resistance $\bar{Y}_c$ is computed by solving a system of equations similar to Eq. (4.18), but replacing Y by Y_c, and $\bar{Y}$ by $\bar{Y}_c$ [29] while the surface contribution (over Γ_s) is left out. If a simulation has a single material and a single value of Y_c, the averaging procedure is not performed and $\bar{Y}_c = Y_c$. The change in the level set field is related to the normal velocity (or level set field increment), v_n, as:

$$\phi_i \leftarrow \phi_i + v_{ni} \tag{4.22}$$

with $i \in \mathcal{N}$, the complete set of nodes in the mesh.

Before computing the front increment along Γ_0, the configurational force $\bar{Y}$ has been computed with the displacement field from the unit load boundary condition. The actual load level for a time step is then determined by scaling the unit-load solution with a load scale factor γ such that the maximum scaled value for $\bar{Y}$ along the front is equal to $\bar{Y}_c$:

$$\gamma^2 \max_{i \in \mathcal{N}_0} \left\{ \frac{\bar{Y}_i}{\bar{Y}_{ci}} \right\} = 1, \tag{4.23}$$

where $\mathcal{N}_0$ is the complete set of nodes of those elements that contains the front. Finally, the front velocity v_n for every node in $\mathcal{N}_0$ is obtained through [29]:

$$v_{ni} = k \left\langle \frac{c\gamma^2 \bar{Y}_i}{\bar{Y}_{ci}} - 1 \right\rangle_+ \quad \text{with} \quad k = \frac{v_{\max}}{c-1}, \tag{4.24}$$

where $v_{\max}$ is the maximum growth the front can experience for a time step. Brackets $\langle \cdot \rangle_+$ are used to denote the positivity condition, which reflects the irreversibility of damage growth. In order to guarantee the numerical stability of the staggered scheme, a value $v_{\max} = \xi h$ is used in this chapter, following [29]. The parameter c influences the spread of the front movement to nodes with lower values for the ratio $\bar{Y}/\bar{Y}_c$. For $c \to 1$, only the node with the highest value $\bar{Y}/\bar{Y}_c$ undergoes a front advance. On the other hand, for higher values of c, nonzero front movement is found in more nodes.

As Eq. (4.24) for v_n is only calculated along the front Γ_0, and due to the fact that the level set update with Eq. (4.22) requires the velocity to be known throughout the domain Ω, v_n has to be determined on Ω. Therefore, the velocity computed at the nodes of elements that contain the front is propagated through Ω by solving:

$$\nabla\phi \cdot \nabla v_n = 0, \tag{4.25}$$

which is carried out with a fast marching algorithm. In theory, the updated level set field obtained with Eq. (4.22) remains a signed distance function [16, 17]. However, the discrete nature of the model may cause it to deviate from being an accurate representation of a signed distance function. Thus, a reinitialization procedure is periodically performed with another fast marching algorithm [29, 40].

4.2.5. INITIATION

In order to take into account independent input parameters for damage initiation and crack propagation, Y_c is made into a function of $\bar{\phi}$, a quantity that measures the size of a damaged zone. With values for Y_c related to an initial strength-based value, Y_c^0, and to an energy-based value, Y_c^G, the following interpolation is adopted [20] (see Fig. 4.4):

$$\log(Y_c) = \log(Y_c^0) + \frac{\bar{\phi} - \bar{\phi}_{\text{init}}}{\bar{\phi}_{\text{max}} - \bar{\phi}_{\text{init}}} \left(\log(Y_c^G) - \log(Y_c^0)\right), \tag{4.26}$$

where $\bar{\phi}_{\text{init}}$ and $\bar{\phi}_{\text{max}}$ are, respectively, the initial size of the damaged zone and the size for which the damaged zone is considered a crack. The quantity Y_c^0 is related to the strength of the material, in which we follow the work in [29]:

$$Y_c^0 = \frac{f_t^2}{2E}, \tag{4.27}$$

where f_t and E are, respectively, the tensile strength and Young's modulus. Unlike earlier studies in [15, 29] where Y_c^G is a direct function of the fracture energy, G_c, and l_c, Y_c^G in this study is obtained through a fitting procedure based on mode I specimens so that the area under the load displacement curve produces an accurate value of G_c (see Section 4.5.1).

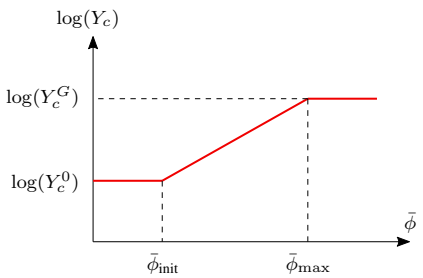


Figure 4.4: Interpolation of Y_c between Y_c^0 and Y_c^G (adapted from [20]).

In contrast to what was proposed in [29], in which $\bar{\phi}$ is computed in a similar way as $\bar{Y}$ in Eq. (4.18), $\bar{\phi}$ is related to the front length of each closed damaged subdomain making use of additional information on the damage front that is anyway needed for the skeleton curve construction (see the function `computePhiBar` in Fig. 4.1). The computation of $\bar{\phi}$ encompasses two stages. First, for a given damage front, its length is computed, and its value is stored at the set of nodes of those elements which contains this damage front. Then, this nodal value is propagated in the same way the front velocity is extended in Eq. (4.25), i.e., by a fast marching algorithm (see Fig. 4.5); however, $\bar{\phi}$ is only propagated through the region where $\phi > 0$. As a result, we have a single value of $\bar{\phi}$ for each damaged subdomain.

In order to deal with damage initiation, the criterion $Y \geq Y_c^0$ is used. If this inequality is met at any undamaged point, a circle with radius $\phi_0 < l_c$ is inserted around the point $\mathbf{x}_{\text{nucl}}$ with the highest ratio Y/Y_c^0. Note that Y used in this criterion is purely local, but still consistent with the non-local growth criterion since $\bar{Y} \to Y$ and $\bar{Y}_c \to Y_c$ as $l \to 0$.

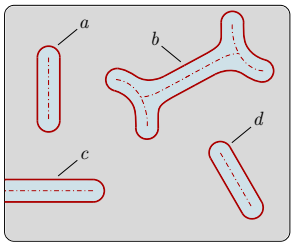


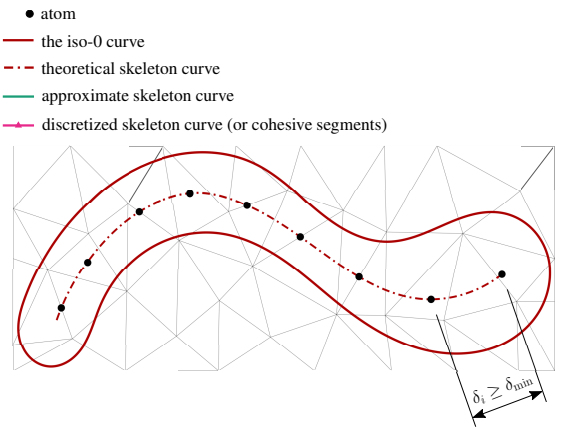
Figure 4.5: Computation of $\bar{\phi}$ for four distinct damaged regions. For each of the four damaged regions, $\bar{\phi}$ is computed as the length of associated damage front.

4.3. SKELETON CURVE

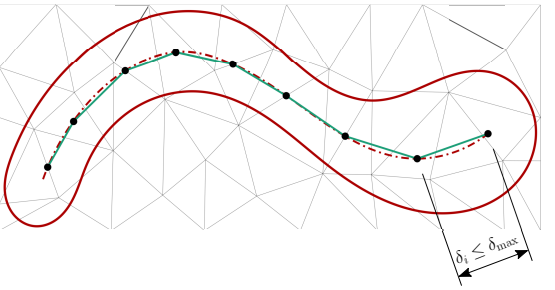
In a setting where the damage front can evolve in arbitrary directions, the TLSV2 requires identification of the skeleton curve of the level set field. Several approaches for determining the location of the skeleton curve have been proposed in the literature, based on the direct information on non-uniqueness of the gradient of the level set field $\nabla\phi$ [75], and on Voronoi cell-based algorithms [76, 77]. In this work, we make use of the ball-shrinking method proposed by Ma et al. [70] due to its implementation simplicity and robustness.

This method relies on the concept of maximally inscribed ball (or *interior medial ball*) of a given region delimited by a contour (or surface), such as damaged regions bounded by the damage fronts found in the TLS-based methods. By definition of the signed distance function, the center of such maximal inscribed ball lies at the skeleton curve [70, 71, 76]. As such, the skeleton curve consists of the set of centers of all interior medial balls of a given region. In a discrete setting, one can obtain a point-based representation of a skeleton curve with a finite set of maximal inscribed balls. This method only requires the set of points that form Γ_0, i.e., the iso-0 curve, and the gradient $\nabla\phi$ at these points as input.

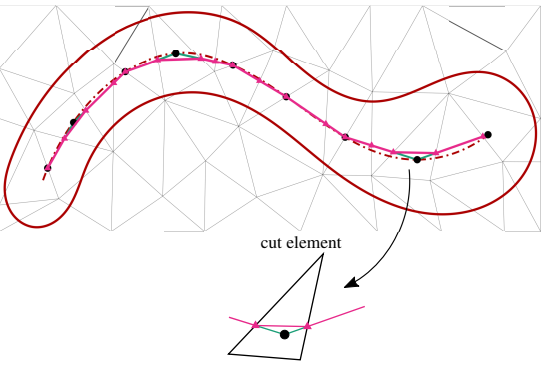
Fig. 4.6 schematically illustrates the *Skeletonizer* algorithm that has been designed in order to determine the skeleton curve for the TLSV2 method. The Skeletonizer is comprised of three parts. Firstly, for a given accurate damaged subdomain enclosed by its iso-0 curve, a ball-shrinking algorithm is executed in order to obtain discrete points that lie at the 'theoretical' skeleton curve, which are called *atoms* or *atom points*, as depicted in Fig. 4.6a. Secondly, these atoms are connected in a manner that corresponds to the connectivity of the skeleton curve, which is a first approximation of the skeleton curve, as illustrated in Fig. 4.6b. This is done by means of a graph-based algorithm. Thirdly, once this approximate skeleton curve is fully known, it can be mapped onto the mesh in order to determine the intersection between the skeleton curve and the finite element edges, as shown in Fig. 4.6c. These intersection points are used to construct the segments that are later used for the phantom node method.



(a) Atom points.



(b) Approximate skeleton curve.



(c) Discretized skeleton curve (cohesive segments).

Figure 4.6: The *Skeletonizer* algorithm for a given damaged region.

The remainder of this section outlines these three parts of the Skeletonizer algorithm. For the sake of simplicity, Ω^d refers to a single damaged region enclosed by Γ_0, unlike elsewhere in this chapter, where it refers to the union of all damaged regions. More-

over, let I_0 be the set of all the points that defines Γ_0 in a finite element domain, i.e., the points that are determined by the intersection between the iso-0 curve and finite element edges. Although the Skeletonizer algorithm is outlined through a single generic damaged domain, it can be applied to problems that consist of multiple damage fronts. In this case, one can apply the proposed skeletonizer algorithm for each damaged subdomain (for instance, see Figs. 4.5 and 4.13).

4.3.1. BALL-SHRINKING ALGORITHM

The atom points are computed with the ball-shrinking algorithm proposed by Ma et al. [70] as improved by Peters [71]. The skeleton curve of Ω^d consists of the set of centers of all interior medial balls in Ω^d. According to Peters [71], a ball is an interior medial ball of Ω^d if it is maximal, namely, if a ball is a subset of Ω^d and any ball that contain such ball is not contained in Ω^d, as exemplified in Fig. 4.7. Furthermore, a medial ball does not contain any part of Γ_0; however, a medial ball does touch Γ_0 at least two points.

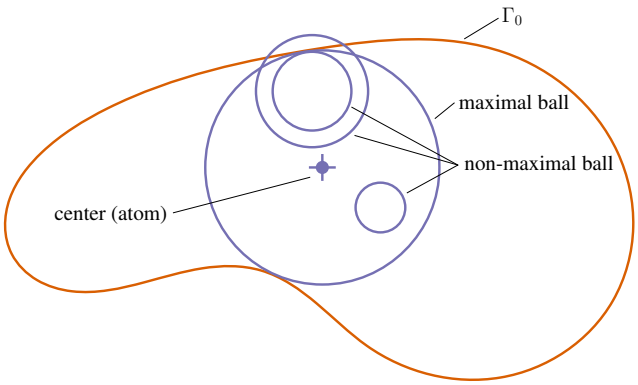
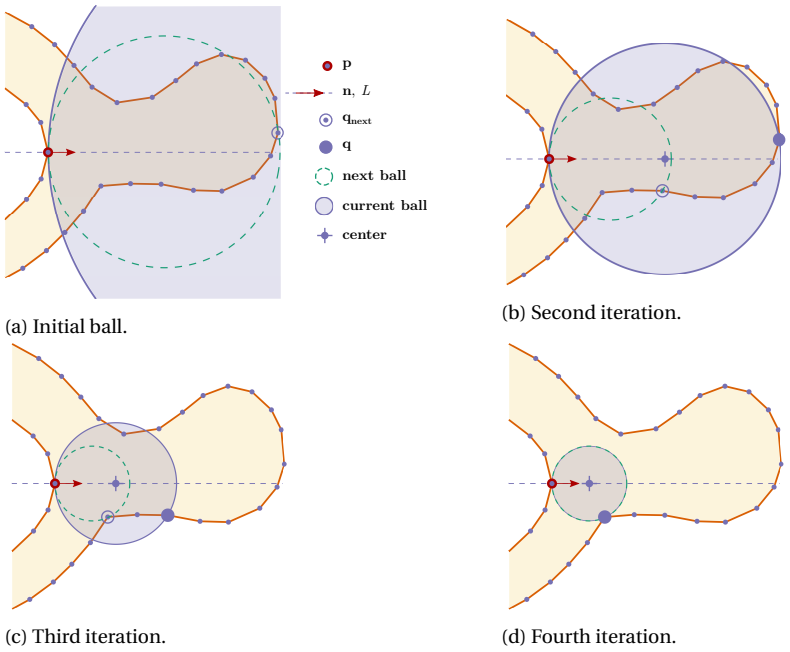


Figure 4.7: Examples of maximal and non-maximal inscribed balls (disc in 2D) (adapted from Blum [78], as cited in Peters [71]).

Fig. 4.8 and Algorithm 4.1 schematically show the ball-shrinking algorithm for a point $\mathbf{p}$ from I_0 with normal vector $\mathbf{n} = \nabla\phi$. The main premise of the algorithm is that the maximal inscribed ball associated with point $\mathbf{p}$ must have its center point on line L, which is the line through $\mathbf{p}$ parallel to $\mathbf{n}$. To find this center point, an initial huge ball is iteratively shrunk along the line L until an interior medial ball is obtained. The algorithm constructs a new candidate ball that is smaller than the previous one and closer to the final interior medial ball at each iteration [71]. Initially, a huge ball with radius r_{init} is generated touching $\mathbf{p}$. For the center of the current ball, $\mathbf{c}$, its closest point to Γ_0, denoted $\mathbf{q}_{\text{next}}$, is computed. With $\mathbf{p}$, $\mathbf{n}$ and $\mathbf{q}_{\text{next}}$, the ball touching the points $\mathbf{p}$ and $\mathbf{q}_{\text{next}}$ is defined for the next iteration. The algorithm terminates when a maximal inscribed ball is found, i.e., when it is not possible to shrink the ball any further (see line eight in Algorithm 4.1).



(a) Initial ball.

(b) Second iteration.

(c) Third iteration.

(d) Fourth iteration.

Figure 4.8: Ball-shrinking algorithm: an interior medial ball is obtained at the fourth iteration (adapted from [71]).

Algorithm 4.1 The `shrinkBall` algorithm (adapted from [71]).

Input: a set of points I_0 and its corresponding KD-tree data structure T_0; point $\mathbf{p} \in I_0$ and its normal vector $\mathbf{n}$; and denoising angle θ_1
Output: the medial ball center $\mathbf{c}$; and radius r associated to $\mathbf{p}$

$i \leftarrow 0$
$r \leftarrow r_{\text{init}}$
$\mathbf{c} \leftarrow$ `computeCenter` $(\mathbf{p}, \mathbf{n}, r)$
while *true* **do**
  $\mathbf{q}_{\text{next}} \leftarrow$ `nearestNeighbour` $(T_0, \mathbf{c})$
  $r_{\text{next}} \leftarrow$ `computeRadius` $(\mathbf{p}, \mathbf{n}, \mathbf{q}_{\text{next}})$
  $\mathbf{c}_{\text{next}} \leftarrow$ `computeCenter` $(\mathbf{p}, \mathbf{n}, r_{\text{next}})$
  if $r_{\text{next}} \geq r - \epsilon_{\text{tol}}$ **then**
    break
  end if
  if $i > 0$ *and* $\theta_1 > \angle \mathbf{p}\mathbf{c}_{\text{next}}\mathbf{q}_{\text{next}}$ **then**
    break
  end if
  $\mathbf{c} \leftarrow \mathbf{c}_{\text{next}}$
  $r \leftarrow r_{\text{next}}$
end while

Algorithm 4.1 makes use of three special functions: `nearestNeighbour`, `computeRadius` and `computeCenter`. The `nearestNeighbour` function simply returns the closest point from the set I_0 to a given query point, which is efficiently implemented using a KD-tree data structure [79, 80], represented by T_0 in Algorithm 4.1. Note that this tree-like data object has to be built prior to the iterative loop of the ball-shrinking algorithm [71]. The remaining two functions are schematically illustrated in Fig. 4.9.

The function `computeRadius` determines the radius of a new ball for a given triplet: **p**, **n**, and $\mathbf{q}_{\text{next}}$. The function `computeCenter` gives the center of a new ball for a given triplet: **p**, **n**, and r_{next}.

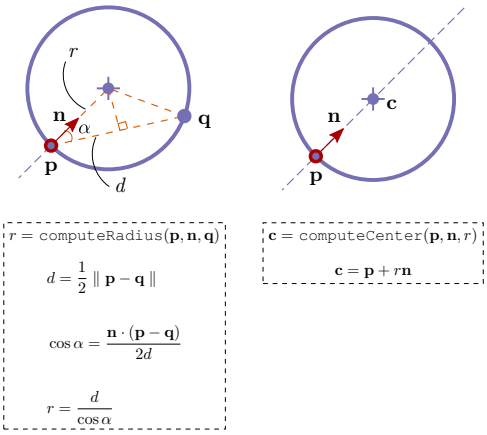


Figure 4.9: The auxiliary functions, `computeRadius` and `computeCenter` , used in Algorithm 4.1 (adapted from [71]).

Building on the original algorithm by Ma et al. [70], Peters [71] proposed a denoise approach in order to avoid *spurious* interior medial balls due to the presence of noisy points at Γ_0. The main premise of this procedure is that even in the presence of a noisy point, a proper medial interior ball is usually found before shrinking it further to become a spurious one. In order to determine at which iteration the ball-shrinking algorithm has to be interrupted, Peters [71] suggested to use a separation angle, θ, observing that a point **q** that varies at each iteration may be shifted from one side to the other side of Γ_0; as a result, θ suddenly becomes smaller, as illustrated in Fig. 4.10.

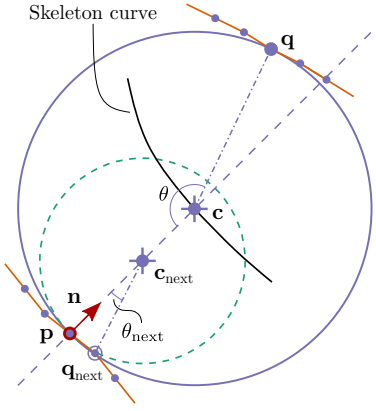


Figure 4.10: Example of the concept of separation angle when a disk flips side, from one side of the Γ_0 to the other. The spurious ball can be detected by the small separation angle θ_{next} (adapted from [71]).

The threshold angle θ_1 is used to detect a spurious ball to obtain a balance between robustness and sensitivity to small noise in Γ_0 [71]. Because exterior medial balls are not considered in this work, unlike in [71], only one such condition is included in the algorithm.

Note that Algorithm 4.1 computes the interior medial ball for a single point $\mathbf{q}$ in I_0. Hence, to obtain all the atoms associated to the region Ω^d, it is necessary to loop over all the points in I_0, as shown in Algorithm 4.2. In order to avoid overlapping atoms and atoms too close to one another (also avoiding elements with multiple atoms), a distance threshold, $\delta_{\min}$ (see Fig. 4.6a), is imposed. Therefore, before storing a center of a medial ball, this distance restriction is checked via `distNearestAtom` on line four in Algorithm 4.2, where the distance between a given center $\mathbf{c}$ and its closest existing neighbor atom is computed.

Algorithm 4.2 The `makeAtoms` algorithm.

Input: a set of points I_0
Output: interior medial balls store in M_{atom} for a region whose boundary are formed from all the points in I_0

```
1: T_0 ← makeKDTree (I_0)
2: for all p ∈ I_0 do
3:     c ← shrinkBall (p, n, T_0)
4:     δ ← distNearestAtom (M_atom, c)
5:     if δ ≥ δ_min then
6:         M_atom ← push (c)
7:     end if
8: end for
```

4.3.2. APPROXIMATE SKELETON CURVE

In order to connect the atoms to form a skeleton curve (see Fig. 4.6b), a spanning-minimum-tree problem is solved. In this problem, a weighted and undirected graph is given, for which the smallest possible tree (an acyclic graph) is found that still connects all vertices of the original graph [81].

Therefore, before solving this problem, a graph data type needs to be constructed, $G = (V, E)$, consisting of:

- a complete set of vertices, V, which corresponds to the atoms;
- a complete set of edges connecting the atoms, E. For each edge, an edge weight, the distance between two linked neighbor atoms, is set.

Algorithm 4.3 shows how the minimum spanning tree associated with the atom points is built. The function `initAtomsGraph` on line one is responsible for connecting atoms whose outcome is a graph data representation of atoms. Once this graph is constructed, Prim's algorithm is used to obtain its corresponding minimum spanning tree, as represented by the function `primsAlgorithm` on line two. This greedy algorithm finds a minimum tree from a graph representation by constructing this tree one atom at a time, from an arbitrary starting atom, at each step adding the closest possible connected atom from the graph to another atom [79, 81]. After determining the minimum spanning tree associated to atom points, a loop over such tree is executed in order to check if the edges of tree are too long through a maximum distance value for threshold edge selection, $\delta_{\max}$ (Fig. 4.6b), or are crossing Γ_0 at any point. The helper function

`distEdge` computes the length of a given tree edge. If one of these conditions is met, the corresponding edge is removed from the tree via the function `removeEdge`.

Algorithm 4.3 The `makeAtomsGraph` algorithm.

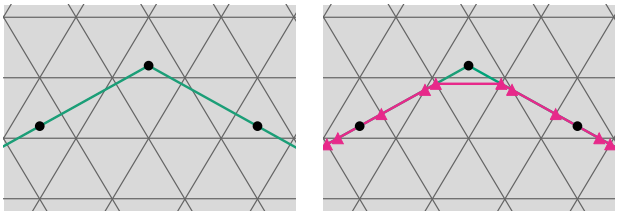
Input: a set of medial balls M_{atom}
Output: a graph representation of atoms, G
1: $G \leftarrow$ `initAtomsGraph` (M_{atom})
2: $G \leftarrow$ `primsAlgorithm` (G)
3: **for all** $edge \in G$ **do**
4: $\quad \delta \leftarrow$ `distEdge` ($edge$)
5: $\quad$ **if** $\delta \geq \delta_{\max}$ **or** $edge$ *crosses* Γ_0 **then**
6: $\quad\quad$ `removeEdge` ($G, edge$)
7: $\quad$ **end if**
8: **end for**

4.3.3. DISCRETIZED SKELETON CURVE

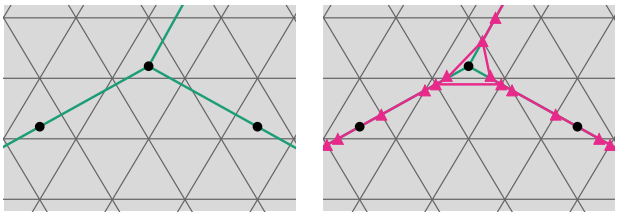
The skeleton curve obtained with the previous two steps connects atoms that are generally not on element edges. However, for the phantom node method, we are looking to construct segments that cross elements in a single straight line. The location of the cohesive segments can be determined by projecting the approximate skeleton curve onto the finite element mesh (see Fig. 4.6c). This operation is performed in two stages. Firstly, crack branches (end-to-end, end-to-junction, or junction-to-junction branches) are generated from the connectivity of the approximate skeleton curve. Next, looping over these branches, the intersection points between the approximate skeleton curve and the elements are determined. As a result, the approximate skeleton curve is mapped on the mesh. This last stage can be efficiently carried out by a R-tree data structure used for spatial searching. An R-tree can store any set of objects (polygons, line segments, points, for instance), and it can give intersection queries with any other object. For an extensive overview of R-trees and their implementation, see [79, 82].

The resulting intersection points and finite elements crossed by the cohesive segments can again be stored as a graph, in which the elements are treated as vertices and the intersection points are treated as edges. This storage scheme facilitates determining for any given finite element which intersection points it contains, and to which finite element it is connected.

Two points of extra attention exist in the element-intersection graph representation. First, one atom might give rise to a 'corner', as exemplified in in Fig. 4.11. Second, unlike the graph representation of the approximate skeleton curve that has acyclical characteristic (see Fig. 4.13), the element-intersection graph representation might present local cycles at locations where one atom contains a junction, as exemplified in Fig. 4.11. Corners are considered to be elements with more than one intersection points lying on the same element edge (see Fig. 4.11a); on the other hand, junctions are considered to be elements containing more than two intersection points (see Fig. 4.11b). Furthermore, the approximate skeleton segment might cross the same element edge twice, which in turn produce two edges that both connect the same of pair of vertices since these intersection are stores as edges in the element-intersection graph. In order to deal with such scenarios, shortcuts are made in the discretized skeleton curve, as illustrated by the magenta curves in Fig. 4.11.



(a) Corner element.



(b) Junction element.

Figure 4.11: Example of corner and junction elements, which contain local cycles or loops. Triangle markers indicate intersection points between the approximate skeleton curve and finite element edges used to define the discretized skeleton curve

4.4. PHANTOM NODE METHOD

Once the discretized skeleton curve has been located, i.e., when Γ_s has been determined, the phantom node method is used to introduce a discontinuity in the solution basis at that location. Fig. 4.12 schematically illustrates the phantom node method. When an element is crossed by a crack at Γ_s, such element is divided into two complementary subdomains, A and B, as illustrated in Fig. 4.12. Phantom nodes are added over the original ones, i.e., $\tilde{n}_1$, $\tilde{n}_2$, and $\tilde{n}_3$. Thus, the original element is substituted by two new elements, A and B, which have the exact same geometry but different connectivity. The shaded area indicates which part of a new element is active, Ω_A and Ω_B. The internal force vector and stiffness matrix contribution of each new element to the global system of equations are obtained by integrating only over the active part of the elements.

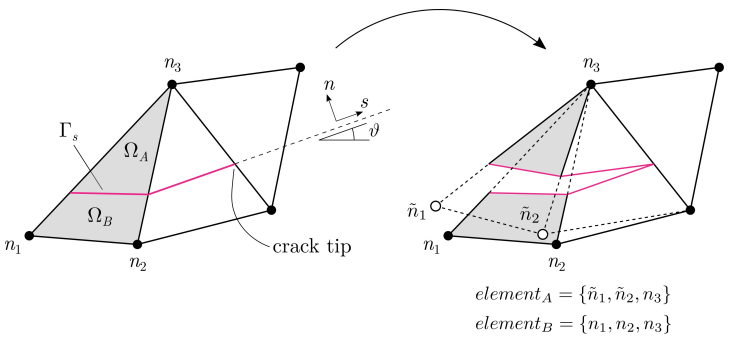


Figure 4.12: Connectivity and active parts of two overlapping triangular finite elements in phantom node method. Local frame (n, s) and its orientation angle ϑ.

The discontinuous displacement field in the pair of overlapping elements is defined as:

$$\mathbf{u}(\mathbf{x}) = \begin{cases} \mathbf{N}(\mathbf{x})\mathbf{u}_A, & x \in \Omega_A \\ \mathbf{N}(\mathbf{x})\mathbf{u}_B, & x \in \Omega_B \end{cases}, \tag{4.28}$$

where $\mathbf{N}$ contains the standard finite element shape functions, and $\mathbf{u}_A$ and $\mathbf{u}_B$ are the nodal displacements from element A and B. The displacement jump over the crack is computed as the difference between the displacement fields of the two overlapping elements:

$$[\![\mathbf{u}]\!](\mathbf{x}) = \mathbf{N}(\mathbf{x})(\mathbf{u}_A - \mathbf{u}_B), \quad \mathbf{x} \in \Gamma_s. \tag{4.29}$$

Observe that Eq. (4.29) gives the displacement jump in the global frame, while the constitutive-related expressions in Eqs. (4.13) to (4.15) are evaluated in the local frame (n, s). The transformation from global to local coordinate frame is performed as:

$$[\![\bar{\mathbf{u}}]\!] = \mathbf{Q}[\![\mathbf{u}]\!], \tag{4.30}$$

where the transformation matrix $\mathbf{Q}$ is given as:

$$\mathbf{Q} = \begin{bmatrix} -\sin\vartheta & \cos\vartheta \\ \cos\vartheta & \sin\vartheta \end{bmatrix}. \tag{4.31}$$

It is also possible to relate:

$$\bar{\mathbf{t}} = \mathbf{Q}\mathbf{t}. \tag{4.32}$$

More details on how the internal force vector and linearized stiffness matrix are assembled and their contribution to the global system of equations can be found in [4, 5, 72].

For a known discretized skeleton curve, possibly containing junctions and multiple disconnected parts, the phantom nodes and overlapping elements are generated in the `updateMesh` function (see Fig. 4.1). This is done by looping over the vertices of the element-intersection graph turning 'right' wherever it meets a junction. Figure 4.13 gives a general view of how these loops are constructed. Looping along each of the dashed lines, the part of the elements on the right side of the skeleton curve is taken as the active part, while phantom nodes are introduced for all nodes on the left side. During this procedure, end vertices are regarded as anchor points, at which a crack path begins and ends. This approach facilitates consistency in the connectivity of neighboring pairs of elements. An example of implementation of these two tasks on the element-intersection graph and phantom node constructions, and shortcut operations in the discretized skeleton curve can be found in [83].

It can be observed from Fig. 4.13 that at junctions, small triangles are created. Elements are defined along the dashed lines, taking the closest magenta skeleton line as the edge of the active domain. As a consequence, the triangles inside the junctions do not belong to the active part of any element. It can occur that junction elements are split in three active parts (e.g., at the junction of paths 1, 2, and 3 in Fig. 4.13). With the proposed procedure for defining the phantom nodes and overlapping elements, no special treatment is required to allow for presence of three or more overlapping element, except

that no cohesive segments are introduced in elements with more than two overlapping elements. Near the junctions it may also happen that there are two overlapping elements that do not touch (e.g., between paths 8 and 10 in Fig. 4.13). Also in this case, no cohesive element is introduced. In other words, elements that contains junction, their contribution to the system of equation for equilibrium solution phase and to the system of equation in Eq. (4.18) are not taken into account.

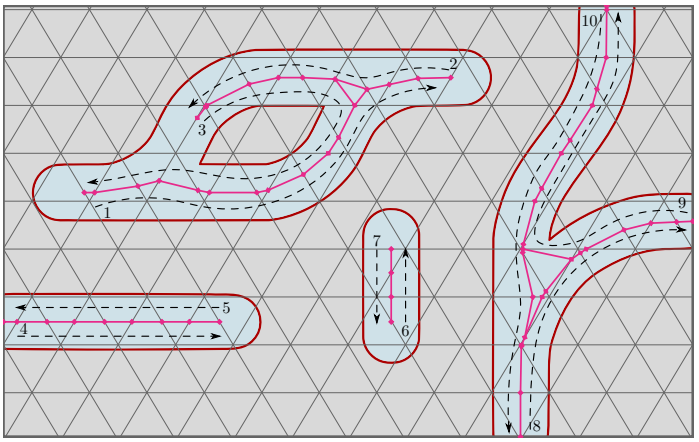


Figure 4.13: Example of phantom node construction. The top left damaged subdomain illustrates the 'global' acyclical property of the atom graph after executing Prim's algorithm.

4.5. RESULTS AND DISCUSSION

The performance of the proposed TLSV2 method is assessed with numerical examples in this section. The method has been developed with Jem/Jive [65], which is an open-source toolkit for finite element analysis, along with the Boost Graph Library [79], which contains minimum spanning tree algorithms, tools to manage graph data structures, and R-tree functionality.

For all numerical examples, unstructured meshes of linear triangles generated with Gmsh [48] are considered. For nucleation, the size of a new damage nucleus ϕ_0 is about the effective element size. The stabilization parameter from Eq. (4.18) is set to $\kappa = 1.2$. Furthermore, the constant ξ used to compute $v_{\max}$ in Eq. (4.24) is set to $\xi = 0.5$, the constants $\bar{\phi}_{\text{init}}$ and $\bar{\phi}_{\max}$ for Y_c-interpolation expression in Eq. (4.26) are, respectively, set to $\bar{\phi}_{\text{init}} = 0$ and $\bar{\phi}_{\max} = 2\pi l_c + 2l_c$. The compression factor in Eq. (4.4) is $\eta = 0.92$, and the user-defined level set value $\phi_\star = 0.5l_c$ (see Eqs. (4.3) and (4.5) and Fig. 4.2). The penalty stiffness terms is $K = 8 \times 10^4\,\text{N/mm}^3$ (see Eqs. (4.12), (4.14) and (4.15)).

Regarding the parameters used for the Skeletonizer algorithm, the denoising angle is $\theta_1 = 125°$, the initial ball has $r_{\text{init}} = 50l_c$ (see Algorithm 4.1). Besides, the threshold parameter $\delta_{\min}$ is about three times the effective element size, i.e., $\delta_{\min} \approx 3h$, and $\delta_{\max} \approx 4\delta_{\min}$ (see Algorithms 4.2 and 4.3 and Fig. 4.6).

4.5.1. COMPACT TENSION TEST

As a first example, the response of a compact tension test under plane stress is considered. Boundary conditions and geometry of the specimen are shown in Fig. 4.14. Young's modulus, Poisson's ratio, and tensile strength are, respectively, $E = 7000\,\text{MPa}$, $\nu = 0.3$, and $f_t = 79\,\text{MPa}$. The critical length l_c is equal to 3 mm. This example is performed with $c = 2$ (see Eq. (4.24)). A refined mesh, with effective element size $h = 0.25\,\text{mm}$, is applied in the region where the crack is expected to develop. The nucleation check is only performed around the notch tip. As this example is a tension-dominant problem, the parameter β in Eq. (4.9) is equal to one, which associated with Eq. (4.8) lead to a symmetric material law that presents the same behavior in tension and compression.

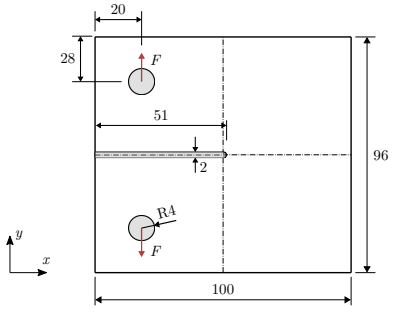


Figure 4.14: Compact tension test: boundary condition and geometry (dimensions in mm).

In Fig. 4.15, the load-displacement curves from Van der Meer and Sluys [29], obtained with a cohesive zone analysis with $G_c = 40\,\text{N/mm}$, and the TLSV2 are drawn together, and the deformed specimen for the final time step is shown. The result verifies the accuracy of the proposed model. Unlike earlier TLS studies [15, 29], in which the parameter related to crack growth Y_c^G (see Eq. (4.26)) was a direct function of the fracture energy G_c, and l_c, Y_c^G in this work is obtained via a fitting procedure at $Y_c^G = 15.6\,\text{MPa}$. The match with cohesive zone analysis in the descending branch shows that during crack propagation the effective fracture energy obtained with the TLSV2 is constant.

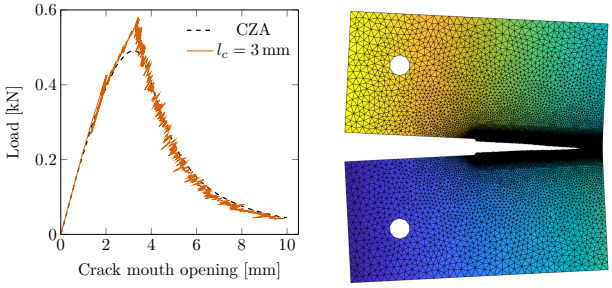


Figure 4.15: Compact tension test: load-displacement curve in comparison with the cohesive zone analysis (CZA) as presented in [29] (left), and deformed specimen with plotted vertical displacement field (right) considering $l_c = 3\,\text{mm}$.

Figure 4.16 shows the load-displacement graphs for different values of l_c: 2 mm, 3 mm, and 4 mm. Observe that the curves (the graph on the left-hand side) obtained with l_c equal to 2 mm and 4 mm, and with the fixed value of $Y_c^G = 15.6$ MPa drift away from the corresponding simulation with $l_c = 3$ mm since they are simulated without their corresponding fitted Y_c^G-value.

The graph on the right-hand side shows results from simulations where Y_c is varied along with l_c in a inversely proportional way. The inverse proportionality is based on the fact that Y_c^G is a direct function of G_c and l_c in TLSV1 models (cf. [15, 29], where $Y_c^G = \frac{G_c}{l_c}$). It can be observed that also with TLSV2, the same effective fracture energy is obtained for different combinations of Y_c^G, and as long as the product of the two is equal.

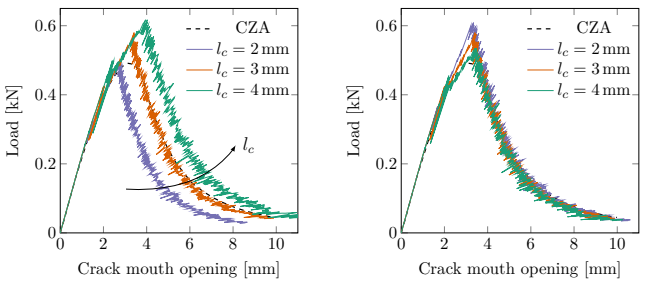
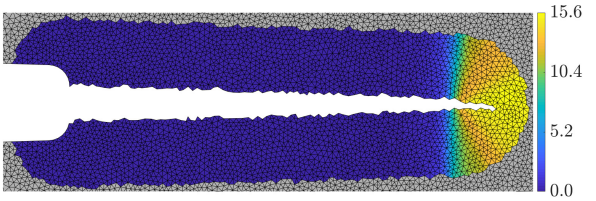
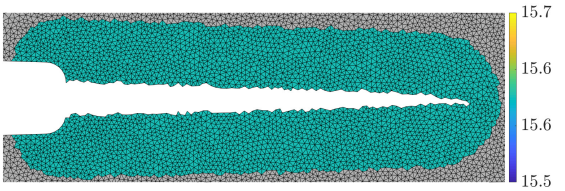


Figure 4.16: Compact tension test: load-displacement curves with fixed values of $Y_c^G = 15.6$ MPa (left), and with inversely proportionally adapted values of Y_c^G (right).

Figure 4.17 shows the two fields of the averaged quantities $\bar{Y}$ and $\bar{Y}_c$ obtained through Eq. (4.18). It is important to emphasize the consistency of these results, in which $\bar{Y}$ is zero in the region behind the crack tip, indicating the presence of a traction-free crack and $\bar{Y}_c$ does reach the constant maximum value of 15.6 MPa, the value set in Y_c^G.



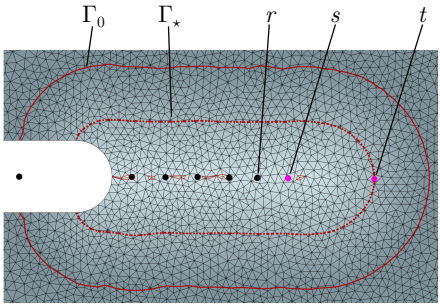
(a) Non-local configurational force, $\bar{Y}$ (MPa).



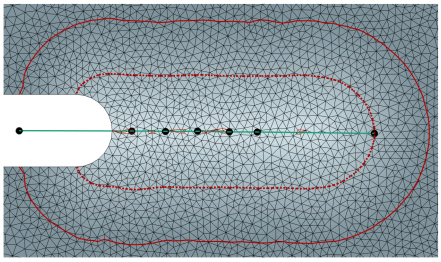
(b) Material resistance to crack growth, $\bar{Y}_c$ (MPa).

Figure 4.17: Compact tension test: averaged quantities at given load step in the post-peak response.

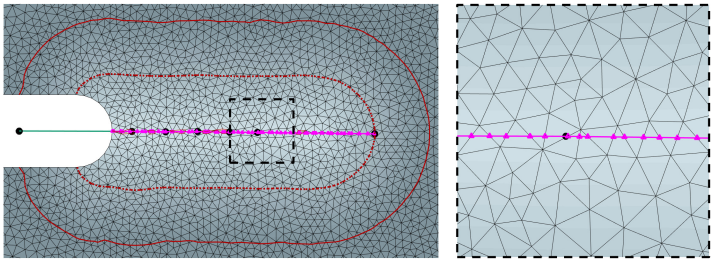
Figure 4.18 shows the outcome of main stages of the Skeletonizer algorithm for one time step from the compact tension simulation. Note that the last atom point obtained by Algorithm 4.2 is the atom s located at $\mathbf{c}_s$; as a result, it would have a lack of cohesive segments between s and the curve $\Gamma_\star$, which in turn gives rise to a non-gradual evolution of traction forces in this region. In order to avoid this absence of cohesive segments in this region, the atom s is moved until it reaches $\Gamma_\star$, in which its new location is indicated by the atom t located at $\mathbf{c}_t$. This shifting procedure is accomplished by moving s on the line oriented by the normal vector defined as $\mathbf{n} = \frac{\mathbf{c}_s - \mathbf{c}_r}{\|\mathbf{c}_s - \mathbf{c}_r\|}$; therefore, the intersection point between such line and $\Gamma_\star$ defines the location of t.



(a) Atom points.



(b) Approximate skeleton curve.



(c) Cohesive segments.

Figure 4.18: Compact tension test: stages of Skeletonizer algorithm with iso-0, iso-$\phi_\star$, and iso-l_c curves of the level set field.

4.5.2. MODIFIED COMPACT TENSION TEST

In the second example, the compact tension test is modified, as shown in Fig. 4.19, in order to show the ability of the proposed framework to deal with crack branching and, consequently, a junction in the skeleton curve. This numerical example is inspired by crack branching case studied by Moës et al. [13] in which the specimen is made of two different Young's moduli. The soft material has the same properties used previously in the first example, and for the stiff material, the properties are $E = 70\,000$ MPa, $\nu = 0.3$, and $Y_c^G = 468$ MPa. The critical length l_c is equal to 1 mm, and the parameter c is increased, $c = 4$, in order to promote more modes along the damage front. The typical element size h is equal to 0.15 mm around the region where the crack is expected to develop. Making use of the previously calibrated value and because Y_c^G is inversely proportional to l_c, Y_c^G is equal to 46.8 MPa in this example.

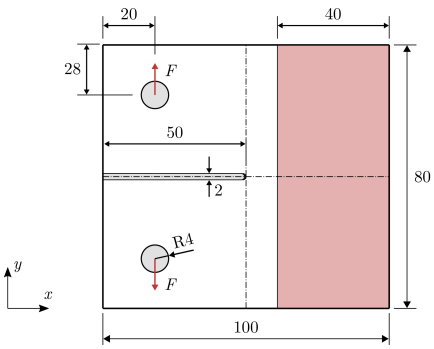


Figure 4.19: Modified compact tension test: boundary condition and geometry (dimensions in mm). Colored area indicates where the stiffer material is set.

Fig. 4.20 shows the load-displacement curve and crack evolution along with its approximate skeleton curve for some time steps. The crack is initiated in the soft material and grows without touching the region where the stiff material is defined.

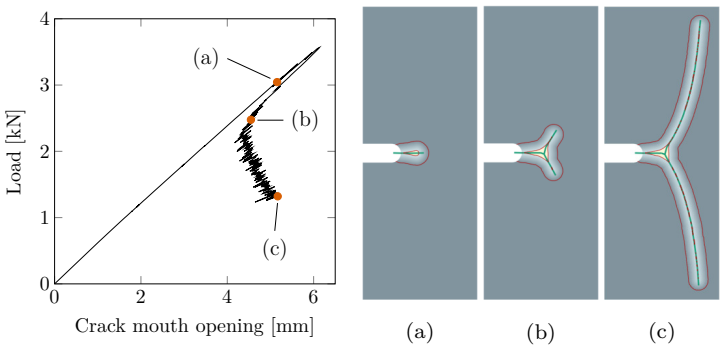


Figure 4.20: Modified compact tension test: load-displacement curve, and crack evolution (close-up) and its corresponding approximate skeleton (green curve).

Figure 4.21 shows the deformed specimen and a close-up figure of the junction.

Crack branching naturally takes place as the iso-0 curve evolves, and consequently, its corresponding skeleton curve. This case would clearly be more complicated to represent with classical XFEM-based models (see Fig. 4.22).

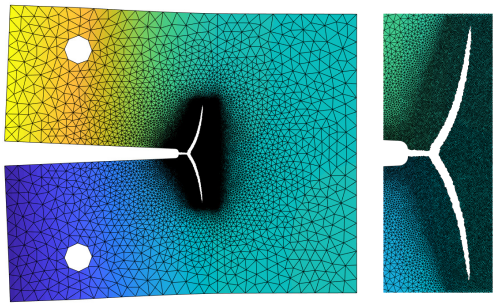


Figure 4.21: Modified compact tension test: deformed specimen (left), and a close-up figure of the crack (right).

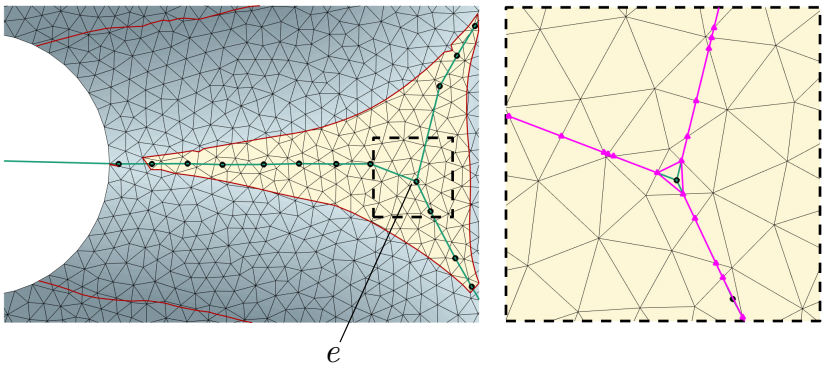


Figure 4.22: Modified compact tension test: zoom of the junction located at element e that has been divided into three overlapping elements.

4.5.3. RAIL SHEAR TEST

The final example is a case where traction-free sliding deformation is desired. The case is inspired by the plane strain rail shear test for mode II failure analysis following Van der Meer and Sluys [29]. The case consists of a weak core sandwiched between two stiff arms, as illustrated in Fig. 4.23. The arms are loaded in opposite direction so that the core is sheared. This setup leads to the crack distribution observed in experimental tests of cusps forming in resin-rich regions of composite material in mode II loading conditions.

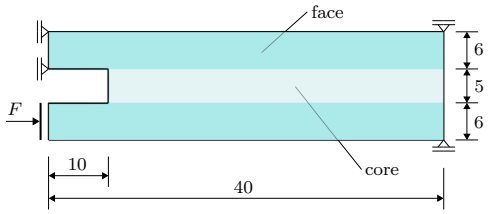


Figure 4.23: Rail shear test: boundary condition and geometry (dimensions in mm).

Young's modulus, Poisson's ratio, and tensile strength of the core material are the same as mentioned in the first example in this paper. For the face material, the properties are $E = 2.1 \times 10^3$ GPa (three hundred times stiffer than the core material), $\nu = 0.3$, and $f_t = 79$ MPa. The typical element size h is 0.15 mm throughout the core region, and l_c and Y_c^G are, respectively, 1 mm and 46.8 MPa. This simulation is performed with $c = 2$, the same value used in the compact tension test.

SINGLE DAMAGE NUCLEUS

Following the approach used to investigate a similar setup with TLSV1 in [29], the problem is first considered to have an initial damage nucleus at the mid-height plane of the core, 15 mm from the left side edge, with radius 0.95 mm, which is slightly smaller than l_c, in order to assess the material laws obtained through Eq. (4.8) by varying the parameter β in Eq. (4.9).

Three different choices for β are investigated. Firstly, β is set to one, which gives rise to a symmetric constitutive model for the bulk in the sense that stiffness degradation is equal under tension and compression. Figure 4.24 shows the load-displacement curve and crack distributions for this case. It is clear that this constitutive model leads to an unrealistic behavior. Similar to what was observed with the TLSV1, initially an X-shaped crack appears, with unphysical compressive branches. However, with the TLSV2, the tensile branches are eventually favored, because there is still tension/compression asymmetry in the cohesive model. Nevertheless, secondary compressive branches appear, and the overall final crack pattern is not realistic. It is concluded that the asymmetry in the cohesive part of the TLSV2 is not sufficient to prevent compressive damage.

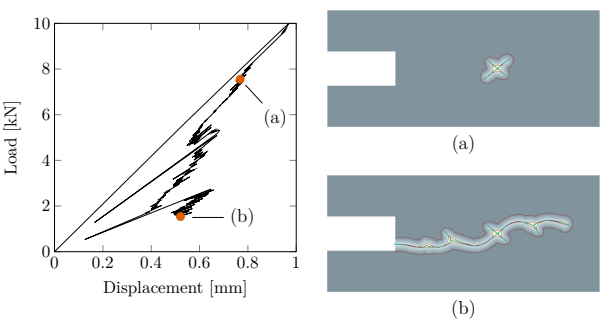


Figure 4.24: Rail shear test with an initial damage nucleus and $\beta = 1$: equilibrium curve, and crack distribution for two different time steps are marked on the load-displacement curve.

The second simulation is performed with $\beta = 0$ where the degradation of material behaves asymmetrically, with complete stiffness recovery under compression. In Fig. 4.25, the equilibrium curve obtained with β equal to zero is plotted, and the final crack distribution and its corresponding deformed configuration are shown. Unlike to what is shown with the TLSV1 in [29], where a hardening phenomenon is obtained with the same constitutive model due to the fact that one of the principal strains in the shear band between arm and core becomes negative leading to stiffness recovery and stress transfer across the crack, the TLSV2 model proposed does not show such undesired behavior. The contact condition in the discontinuity does not prevent sliding deformations, and the surrounding bulk material can completely unload. However, the choice for $\beta = 1$ does lead to more oscillatory behavior in the post-peak response.

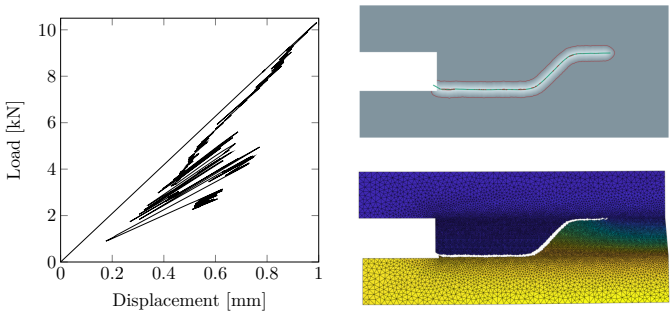


Figure 4.25: Rail shear test with an initial damage nucleus and $\beta = 0$: load-displacement curve, and final crack distribution and its corresponding deformed specimen with horizontal displacement field.

The last one-nucleus simulation uses a constitutive model that combines the benefits from both models that have been investigated previously. The choice for $\beta = 1$ brings robustness, while $\beta = 0$ gives the correct directionality for crack propagation. For the initial transition from a circular damage nucleus to a clean tensile crack under shear loading, full stiffness recovery under compression ($\beta = 0$) is essential. Later, when the asymmetry in tension/compression behavior is partially represented across the displacement discontinuity, stiffness recovery in the bulk becomes less important, although a value of $\beta = 1$ may still lead to undesirable secondary compressive crack branches. For large cracks, an intermediate value of β is sufficient and still provides an improvement in stability over the stricter $\beta = 0$. Because we have information on the size of the damaged zone through $\bar{\phi}$, we can achieve a transition for β from 0 to a higher value as the damaged zone grows. Therefore, β is made into a function of the size of damaged zone $\bar{\phi}$ as follows (see Fig. 4.26):

$$\beta(\bar{\phi}) = \begin{cases} 0, & \bar{\phi} \leq \bar{\phi}_{\text{init}} \\ \frac{\bar{\phi}-\bar{\phi}_{\text{init},\beta}}{\bar{\phi}_{\text{max},\beta}-\bar{\phi}_{\text{init},\beta}}\beta_{\text{max}}, & \bar{\phi}_{\text{init},\beta} < \bar{\phi} \leq \bar{\phi}_{\text{max},\beta} \\ \beta_{\text{max}} \leq 1, & \bar{\phi} > \bar{\phi}_{\text{max},\beta} \end{cases}, \tag{4.33}$$

where $\bar{\phi}_{\text{init},\beta}$ and $\bar{\phi}_{\text{max},\beta}$ are, respectively, the initial size of the damaged zone and the size for which the material is considered to behave as a fully constitutive law with $\beta = \beta_{\text{max}}$.

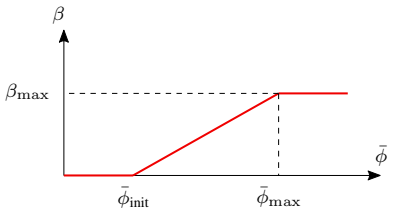


Figure 4.26: Variability of parameter β as a function of the size of damaged domains $\bar{\phi}$.

Therefore, the problem is simulated again with $\bar{\phi}_{\text{init},\beta} = 2\phi l_c + 3l_c$, $\bar{\phi}_{\max,\beta} = 2\phi l_c + 6l_c$, and $\beta_{\max} = 0.88$ for the equation above. Figure 4.27 shows the load-displacement curve with the variability of β, and crack distribution along with its deformed configuration for the last time step. It is clear that the approach proposed for evaluation of β through Eq. (4.33) influences the overall response on this problem. The initial nucleus does not grow into a X-shaped crack, because the material is dictated by the asymmetric constitutive law. In addition, this simulation does not suffer from substantial oscillations when the crack reaches the interface region between core and arms due to the fact that the constitutive law at this point has much less stiffness recovery under compression with $\beta = \beta_{\max}$. Also, note the absence of secondary branches when compared to the final crack configuration obtained with fixed value of $\beta = 1$.

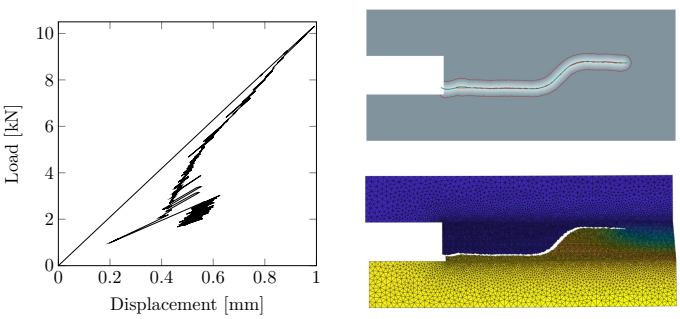


Figure 4.27: Rail shear test with an initial damage nucleus and varying $\beta = \beta(\bar{\phi})$: load-displacement curve, and final crack distribution and its corresponding deformed specimen with horizontal displacement field.

MULTIPLE DAMAGE NUCLEI

With this new way to address the value of β, the shear test is revisited. Now, the TLSV2 model is also applied to a case without predefined initial damage nucleus. The nucleation check is only performed around the mid-height of the core. Besides, the distance between a new damage radius and existing damage fronts is set to be at least 7 mm.

The evolution of damage in the shear test with the varying value of β is shown in Fig. 4.28. As already detailed in a similar test case with the TLSV1 [29], the first nuclei appear before the peak load is reached. As the load increases, the inclined cracks start to form; however, only the left most crack, the main crack, reaches the interface between the two materials, and this crack eventually extends over the whole length of the core.

As the analysis progresses, the load eventually drops as the main crack reaches the right end of the specimen. Furthermore, it can be observed that the skeleton curve of the main crack does not join up with others even though its iso-0 curve does merge with the damage fronts of all inclined cracks. The skeleton curves do not merge because of the fact that when the fronts join up and the main crack moves towards to the right edge of the specimen, these regions experience unloading, after which the front stops evolving.

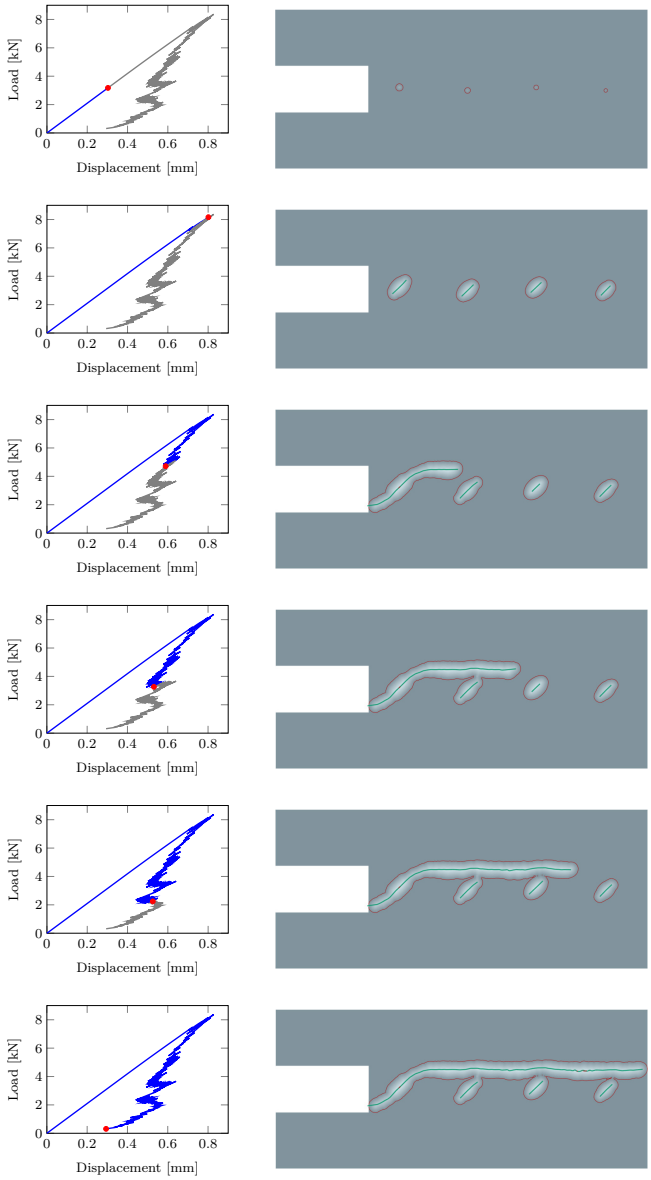


Figure 4.28: Rail shear test: damage front and approximate skeleton curve evolution.

4.6. CONCLUSIONS AND DISCUSSION

In this chapter, the TLSV2 method is extended to provide the ability of dealing with free-form skeleton curves for crack growth problems in solids that bring branching and merging events. The TLSV2 method by Lé et al. [1] has been used as a basis model for the proposed version. A skeletonizer algorithm equipped with the ball-shrinking algorithm in the version by Peters [71] and Prim's algorithm has been designed. The phantom node method has been used to account for the discontinuous part of the TLSV2. Once the skeleton curve has been mapped into the mesh, the construction of the phantom node method including the possibility of crack branching is achieved by traversing along crack branches one side after the other.

After implementing these features, the TLSV2 method proposed was successfully assessed in three numerical examples. The first example, the compact tension test, was used to define some input parameters, and to verify the ability of the TLSV2 to represent crack growth at a constant fracture energy. Even though the fracture energy is not a direct input in the proposed method, this shows that the model can be calibrated to fracture energy measurements. It was demonstrated that it is necessary to extend the last atom point obtained from the ball-shrinking algorithm.

The proposed framework was shown to work for the case of crack branching and merging in the second and third test cases, although merging was only found for the damage front and not for the skeleton. In the shear test, the influence of the stiffness recovery parameter β on the global response was investigated. Variability of β as a function of the size of damaged zone was proposed to benefit from the advantages of the constitutive laws with and without stiffness recovery under compression preventing unrealistic compressive crack branches as obtained with $\beta = 0$, as well as undesirable oscillations as encountered with $\beta = 1$. Furthermore, the special interphase material used in the earlier TLS versions [29, 34, 52] was not necessary, since the crack could grow in a mode II manner without artificial hardening.

The work in this chapter is restricted to unstructured meshes of linear triangles and to 2D simulations. However, the proposed framework, especially the algorithm to locate the skeleton curve in conjunction with the phantom node approach, is expected to work as well for meshes of quadrilaterals without substantial changes.

On the other hand, a 3D version of the proposed framework would require extra attention, mainly for construction of crack surface. Although the ball-shrinking algorithm behaves the same in 3D as it does in 2D [71], it still provides a point-based representation of the medial surface of the iso-0 surface in 3D simulations, i.e., an unstructured set of medial atoms, which would need to be transformed into surfaces with boundaries in order to represent cracks in 3D. An extensive overview of methods for constructing this surface can be found in [84]. Once the medial surface of the iso-0 surface has been determined, the corresponding discretized skeleton surface can be determined, which can subsequently be used in conjunction with the phantom node method for constructing a discontinuity in the displacement field.

5

A PARALLEL IMPLEMENTATION OF THE THICK LEVEL SET V2 METHOD

5.1. INTRODUCTION

In this penultimate chapter of this thesis, the main concepts and features of the numerical models based on the TLS method that have been presented in Chapters 2 to 4 are brought together in order to have a parallel implementation of the TLSV2 method. As already emphasized in Chapter 4, the sequential implementation for the TLSV2 inherits most of the sequential framework designed for the TLSV1. Therefore, the global solution scheme for the TLSV2 outlined in Chapter 4 can readily be plugged into the parallel framework detailed in Chapter 3. In this chapter, we focus only on the highlighted functions shown in Fig. 4.1, which are those functions that have been specifically designed for the TLSV2.

The main premise employed in the parallel framework presented in Chapter 3 is adopted here, namely, that different parts of the TLSV2 are performed with different levels of parallelism. In that spirit, the `skeletonizer`, `updateMesh`, and `computePhiBar` functions are kept in a global context, i.e., only the root process is responsible for handling them, whereas the `IntSchemeUpdate` function is performed in a local context, where each process executes the update on its own subdomain without any data exchange.

In line with the collective communication routines addressed in Section 3.3.2, the `skeletonizer` and `updateMesh` functions are therefore sandwiched between a single `Gather-Scatter` execution block where the root executes their corresponding tasks as follows. Once the root receives the ϕ updated by `updateLevelSet` from each process (see Fig. 4.1), it computes the global skeleton curve through `skeletonizer`, which is subsequently used to determine the updated global mesh by means of `updateMesh`. This execution block ends with the root sending the new mesh quantities (i.e., global IDs of elements and nodes) back to processes possessing the skeleton curve.

As new nodes are dynamically added into the system to be used in the phantom node

approach, the operators **R** and **Q** (see Section 3.3.2), which are used by processes to exchange data in a collective operation and depend on the number of nodes in the mesh, have to be updated accordingly. This update is carried out as soon as the mesh has been changed in the `updateMesh` function.

Regarding the `computePhiBar` function, the root executes a global fast marching algorithm in order to compute $\bar{\phi}$ following the same steps detailed in Section 4.2.5 and schematically illustrated in Fig. 4.5. Despite being a global operation, this task is not positioned between a `Gather-Scatter` pair since the root has already had the global skeleton curve and updated ϕ at this stage. Therefore, only a `Scatter` call is executed in order to send $\bar{\phi}$ back to processes that have damaged regions, i.e., processes that form the set $\mathscr{P}_{\mathrm{d}}$ (see Section 3.3.3). With new nodes at hand, the `IntSchemeUpdate` function is executed defining the integration schemes of the cracked elements and the new displacement DOFs.

In order to investigate the performance of this parallel version of the TLSV2, two numerical examples assessed in Chapter 4 are considered. Firstly, the compact tension test presented in Section 4.5.1 is investigated where the material is still modeled as elastic and, hence, allowing the use of the secant unloading scheme. The final example is chosen as the case that has been shown cusp crack patterns, which have strongly motivated the development of the TLS methods in this thesis, the rail shear test. For this final example, the version presented in Section 4.5.3 is considered, however, unlike in Chapter 4, the core is modeled with plasticity, while the arms are kept elastic.

5.1.1. COMPACT TENSION TEST

The main objective of the first example is to assess the performance of the TLSV2 in its parallel version in terms of accuracy and scalability. For this purpose, load-displacement and speed-up curves considering different numbers of subdomains and cores for the same analysis are considered. For each partition scheme, the problem is run three times, and the average runtime is computed.

The setting parameters, boundary conditions, and geometry of the first example are the same as presented in Section 4.5.1, except for the effective element size, in which a unique value is used for the whole mesh, $h = 0.3\,\mathrm{mm}$, and $l_c = 2.3\,\mathrm{mm}$. In this case, the initial mesh consists of 278380 elements, 140098 nodes and 280196 DOFs. Regarding the overlapping region, one layer of elements is used during the partitioning of the mesh. Making use of the calibrated value (15.6 MPa) from Section 4.5.1, and to the assumption of the inversely proportional characteristic of Y_c^G, $Y_c^G = 20.4\,\mathrm{MPa}$ in this example.

The load-displacement curve for a reference solution, obtained without the parallel framework, is compared with the result of using 24 subdomains in Fig. 5.1. Unlike what was observed in previous test cases in Section 3.4.1, the load-displacement curves are not perfectly matched, however, they do present the same trend, even for the post-peak part.

Figure 5.2 shows for the case of 24 subdomains that the continuity of the level set field and skeleton curve are ensured even when the crack crosses the subdomain boundaries, just like what was found for the damage band with the TLSV1 in Chapter 3.

Figure 5.3 shows the total runtime as well as the total time spent in each analysis phase (level set update, equilibrium solution, and front evolution) as a function of num-

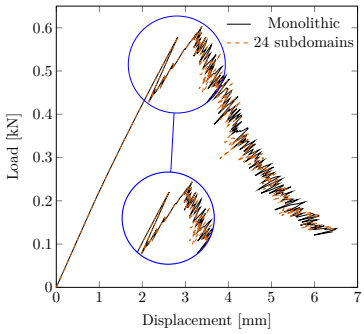


Figure 5.1: Compact tension test: load-displacement curve.

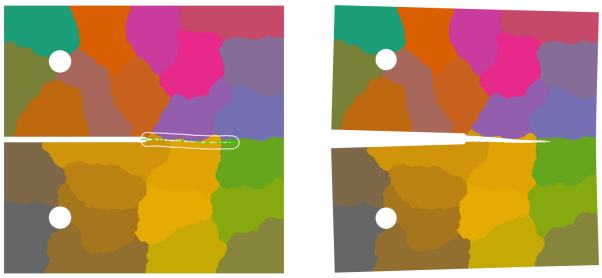


Figure 5.2: Compact tension test: the iso-0 and skeleton curves (left), and deformed configuration (right) in the mesh partitioned into 24 subdomains. The colored set of elements represent the 24 subdomains.

ber of subdomains. By using 24 cores, the parallel framework accelerates the monolithic response by a factor of 12.2. Once again, as already reported in the examples of Chapter 3, the equilibrium solution phase is the most time consuming and presents the best scalability among the three analysis phases. On the other hand, the level set update and front evolution do not scale as well as the equilibrium solution phase, mainly because they rely on collective communication strategies.

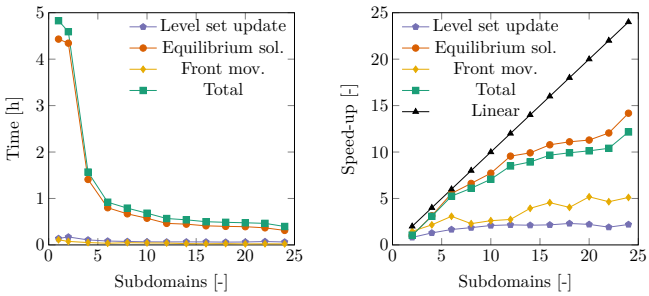


Figure 5.3: Compact tension test: wall clock time (left) and speed-up (right) graphs.

5.1.2. RAIL SHEAR TEST

The main objective of this final example is to show the ability of the parallel implementation of the TLSV2 to handle problems with hardening plasticity. The boundary condition and geometry of Section 4.5.1 are considered. Young's modulus, Poisson's ratio, tensile strength, and hardening curves for the core material are the same as mentioned in Sections 2.3.2, 3.4.2 and 3.4.3. Although Y_c^G has not been calibrated, Y_c^G is set equal to 0.96 MPa, the penalty stiffness is $K = 8 \times 10^2$ N/mm, and the parameter that accounts for stiffness recovery in the bulk material is $\beta = 0.25$ (see Eq. (4.33)). The viscous parameter against crack growth and displacement rate are, respectively, $\eta = 1\,\mathrm{s\,mm^{-1}}$ and $\dot{u} = 0.025\,\mathrm{mm\,s^{-1}}$. For the face material, the properties are $E = 376$ GPa (approximately one hundred times stiffer than the core material), $\nu = 0.3$, $f_t = 79$ MPa, and $G_c = 10$ N/mm.

The typical element size for the core is 0.15 mm while the faces are meshed with element size of 0.8 mm, leading to an initial mesh of 26902 elements, 13587 nodes and 27174 DOFs. The critical length l_c is 1 mm, which is the same as for the corresponding example in Chapter 4. The nucleation check is only performed in the mid-height plane of the core. The SPR operator is used for transferring history.

Figure 5.4 shows the load-displacement curve for 15 subdomains. Figure 5.5 presents the final damage distribution and deformed configuration. Note that the continuity of the iso-0 and skeleton curves are guaranteed on shared regions once again. Also, the compatibility of duplicate mesh quantities for the phantom node approach is ensured on overlapping regions. The whole analysis takes 3.1 h.

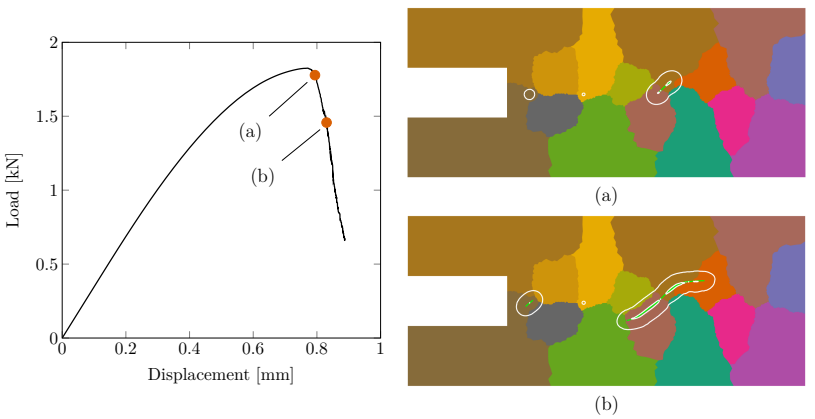


Figure 5.4: Rail shear test: load-displacement curve. The colored set of elements represent the 15 subdomains. The two bullets on the graph correspond to the two figures that show the level set field and its skeleton evolution.

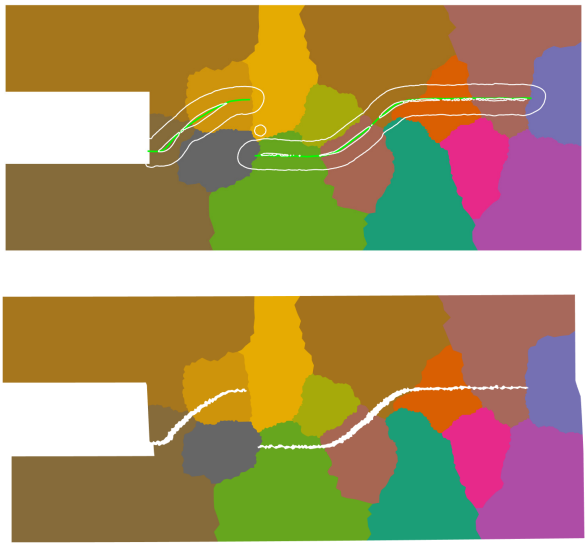


Figure 5.5: Rail shear test: the final iso-0 and skeleton curves (top), and the final deformed configuration (bottom) in the mesh partitioned into 15 subdomains.

5.2. CONCLUSIONS

In this final chapter of this thesis, it has been shown how to obtain a parallel framework for the TLSV2 method based on the numerical tools proposed in Chapters 2 to 4. Minor changes, the majority of them related to the level set update phase, have been required to achieve a parallel version of the TLSV2 method, which shows the versatility and modularity of the numerical tools and implementation that have been proposed in this thesis.

Remarks can be made based on the two examples analyzed in this chapter. As the proposed parallel framework inherits the same design pattern detailed previously in Chapter 3, there is clear resemblance in terms of performance between the two parallel approaches. The equilibrium solution phase presented the best scalability among the other analysis phases, as shown with the speed-up study in the first example. Besides, the presence of the `skeletonize`, `updateMesh`, and `computePhi` functions did not bring much complexity in terms of scalabilty to the level set update phase, even though it involves more complex operations and data storage management compared with the corresponding analysis phase for the TLSV1. Although a speed-up study has been performed only for one test case in this chapter, based on the results shown in Chapter 3, the same performance of the proposed framework for test cases with plasticity is expected. In both simulated examples, the continuity of the level set field and its corresponding skeleton curve were ensured even for cracks crossing multiple subdomains.

6

CONCLUSIONS AND DISCUSSION

A numerical framework based on the Thick Level Set (TLS) method for failure analysis in solids has been extended and assessed. The existing TLS-based methods have been adopted as bases for the proposed framework. Hardening plasticity, parallel computing, and cohesive cracks have been incorporated into the TLS approach as new features within a single implementation architecture. With the proposed framework, a new numerical tool has not only been made to more realistically assessed the complex process of cusp formation in resin-rich regions of composite material in mode II delamination condition.

NEW DEVELOPMENTS

What follows next is a chapter-wise summary of the main contributions that have been made to the collection of existing TLS-based models for the computational modeling of damage and crack growth in solids in a quasi-static context:

- Plasticity has been combined with the TLS method. Two elasto-plastic models have been considered and assessed, the model by Melro et. al. [37] for epoxy resins and the classic Von Mises formulation. A new criterion for damage initiation based on the ultimate yield surface for ductile fracture simulations has been proposed. A new loading scheme that takes into account the permanent strain in the presence of plasticity has been introduced.

- A parallel framework built on the concepts of domain decomposition technology has been designed for the framework. Collective communication routines have been designed to handle TLS-specific analysis phases. A point-to-point communication routine has been employed to handle the enrichment scheme.

- A collection of algorithms and methods have been used and designed to bring to life the full capabilities of the TLSV2. A skeletonizer algorithm equipped with ball-shrinking and graph-based algorithms has been designed for robust identification

of free-form skeleton curves for the TLSV2 method. The phantom node method has been applied to deal with strong discontinuities in the TLSV2 method.

- The proposed TLSV2 model has been incorporated into the parallel framework originally designed for its predecessor version, showing the versatility and modularity of the model components and their computational implementations designed in this work.

CONCLUSIONS

The accuracy and ability of different approaches and components of the proposed model to failure analysis including complex topological crack patterns have been tested in several numerical examples. The main conclusions from the simulations are:

- In a medium with hardening plasticity, the characteristic length, l_c, and the viscous resistance against crack growth, η, have a great influence on the global response. The amount of plasticity can be controlled via these two parameters.
- The superconvergent patch recovery has been found to be an accurate operator to transfer history when the integration scheme in the mesh is changed.
- The influence of geometry of resin-rich regions in fiber reinforced polymer composites on cusp development is qualitatively validated. The cusp pattern is more pronounced in curved-profile configuration, in agreement with experimental observations. Different load-displacement curves and fracture morphologies are obtained by varying the profile geometry of such regions.
- Substantial speed-ups are achieved with the developed parallel framework using a moderate amount of cores. The most time demanding part of the TLS method in terms of computational cost, the equilibrium solution part, scales very well with the number of cores used for the analysis. The less demanding parts of the TLS scales less favorably.
- The proposed skeletonizer algorithm along with the phantom node method give the TLSV2 the ability to handle complex skeleton events, including branching and merging.
- Free-sliding deformation is achieved in test setups with cusp development with the TLSV2, without the necessity of including information of the geometry orientation of problem being investigated.
- The tasks related to the TLSV2 method do not delay considerably the parallel framework, despite having complex operations.

FUTURE PERSPECTIVE

Although this work presents a TLS-based framework to simulate failure in solids upon mechanical loads, open questions remain. In the following, a number of improvements and applications for future research are listed.

- In the testing of the proposed framework in a parallel context, several issues have been encountered that can be improved for better scalability, mainly in the analysis phases of the level set update and front evolution. The fast marching operations could be replaced with a system of equations for velocity extension or reinitialization over the mesh, such that parallel iterative solvers can be applied in these solution phases as well. However, an alternative to parallelize the way averaged quantities (e.g., $\bar{Y}$ and $\bar{Y}_c$) are computed would still be necessary to completely parallelize all main solution phases. Regarding the parallel version of the TLSV2 method, skeleton curves could be computed on each subdomain with subsequent smoothing/matching processing on overlapping regions.

- The certainty that no energy is dissipated when a cohesive segment is inserted along the skeleton curve is neither guaranteed nor investigated in this work. Perhaps, an initially rigid formulation can be adopted. To what extent that is necessary and how it can be achieved are remaining questions for now.

- A 3D implementation is undoubtedly a great challenge, which requires several changes and new features. Firstly, the damage nucleation criterion based on the ultimate yield surface should be rewritten in terms of principal strain components, and the out-of-plane plane component would need to be taken into account. Secondly, the skeletonizer algorithm by means of its ball-shrinking component will still provide a point-based representation of atoms, which should subsequently be transformed into surfaces to represent cracks. The complexity of this issue is further amplified when complex crack events arise during the analysis, such as branching and merging.

- The problem that has motivated the development of this work, the process of cusps in a resin-rich region in mode II delamination, can be investigated more realistically. This process has been reported as one of the reasons for the differences in terms of fracture energy between mode I and mode II crack growth since forming cusps requires the creation of more crack surface than forming a single straight crack. For this purpose, virtual experiments based on well-known setups can be used, such as three-point bend end-notched flexure (ENF) and single leg bending (SLB). A better understanding of this *in situ* process can allow us to develop proper physics-based criteria and traction-separation relations for macroscale simulations.

- Despite having a different context, the models that have been developed in this thesis could be extended to dynamics and fatigue analyses, specially the consideration of cohesive cracks through the specific algorithms that have been designed for the TLSV2. Models by Moreau et al. [19], for dynamics, and by Latifi et al. [21] and by Niessem [23], for fatigue, could be used as bases for such extensions.

ACKNOWLEDGEMENTS

The research outlined in this thesis has been carried out in the Department of Materials, Mechanics, Management & Design (3MD) in the Faculty of Civil Engineering and Geosciences of Delft University of Technology (TU Delft). There are quite some individuals without whom this work would not have been finished.

First of all, I would like to express my deepest gratitude to one of my promotors Frans van der Meer for advising me on my entire PhD research, and for his patience, motivation, encouragement, and dedication. He shared with me his enormous knowledge and scientific insights. I am grateful for the freedom, support, and confidence he has given me, without which this work would have been less relevant, challenging, and pleasant.

Second in order, but absolutely not in relevance, I would like to express my genuine gratitude to my other promotor Bert Sluys for giving me the opportunity to join his computational mechanics group, and for the uninterrupted support of my PhD project. His insightful advice on writing this thesis has been precious to me.

I have been fortunate to have some very nice friends and colleagues to whom I would like to thank you. I appreciate the support by Erik Jan Lingen on the use of Jem/Jive C++ library in a parallel environment. I would like to thank my friends and colleagues Richard Dekker, Lu Ke, Dongyu Liu, Dragan Kovacevic, Ali Paknahad, Tiziano Li Piani, Prashanth Srinivasan, Erik Simons, Yaolu Liu, Suman Bhattarai, Debashis Wadadar, and all the others I might have missed out, a big thank you. I will definitely miss our football matches in the X sport center. My sincere apologies if I have missed anyone here. I make a special mentions to Iuri Rocha and Cristyna Araújo for the company, help during my move to the Netherlands, and all the good time we spent together during my stay in Europe.

I would like to thank my family for the love and support, especially my mother and father, who could not see the end of this journey, and sisters. Much gratitude to my in-laws. Last but not the least, I would like to express my biggest gratitude to my wife Ana for her love, companionship, encouragement, and patience without which this adventure would have been much harder to endure. The research has been carried out in TU Delft and, hence, my final thanks goes to the city of Delft, now, one of my favorite places.

BIBLIOGRAPHY

[1] B. Lé, N. Moës, and G. Legrain. Coupling damage and cohesive zone models with the Thick Level Set approach to fracture. *Engineering Fracture Mechanics* **193** (2018), pp. 214–247.

[2] N. Moës, J. Dolbow, and T. Belytschko. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering* **46** (1) (1999), pp. 131–150.

[3] N. Moës and T. Belytschko. Extended finite element method for cohesive crack growth. *Engineering Fracture Mechanics* **69** (7) (2002), pp. 813–833.

[4] A. Hansbo and P. Hansbo. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering* **193** (33) (2004), pp. 3523–3540.

[5] J.-H. Song, P. M. A. Areias, and T. Belytschko. A method for dynamic crack and shear band propagation with phantom nodes. *International Journal for Numerical Methods in Engineering* **67** (6) (2006), pp. 868–893.

[6] Z. P. Bažant, T. B. Belytschko, and T.-P. Chang. Continuum Theory for Strain-Softening. *ASCE Journal of Engineering Mechanics* **110** (12) (1984), pp. 1666–1692.

[7] G. Pijaudier-Cabot and Z. P. Bažant. Nonlocal Damage Theory. *ASCE Journal of Engineering Mechanics* **113** (10) (1987), pp. 1512–1533.

[8] N. Triantafyllidis and E. C. Aifantis. A gradient approach to localization of deformation. I. Hyperelastic materials. *Journal of Elasticity* **16** (1986), pp. 225–237.

[9] R. H. J. Peerlings, R. de Borst, W. A. M. Brekelmans, and J. H. P. de Vree. Gradient Enhanced Damage for Quasi-Brittle Materials. *International Journal for Numerical Methods in Engineering* **39** (19) (1996), pp. 3391–3403.

[10] V. Hakim and A. Karma. Laws of crack motion and phase-field models of fracture. *Journal of the Mechanics and Physics of Solids* **57** (2) (2009), pp. 342–368.

[11] C. Miehe, F. Welschinger, and W. Hofacker. Thermodynamically consistent phase-field models of fracture: Variational principles and multi-field FE implementations. *International Journal for Numerical Methods in Engineering* **83** (10) (2010), pp. 1273–1311.

[12] C. Miehe, M. Hofacker, and F. Welschinger. A phase field model for rate-independent crack propagation: Robust algorithmic implementation based on operator splits. *Computer Methods in Applied Mechanics and Engineering* **199** (45) (2010), pp. 2765–2778.

[13] N. Moës, C. Stolz, P.-E. Bernard, and N. Chevaugeon. A level set based model for damage growth: The thick level set approach. *International Journal for Numerical Methods in Engineering* **86** (3) (2011), pp. 358–380.

[14] C. Stolz and N. Moës. A new model of damage: a moving thick layer approach. *International Journal for Numerical Methods in Engineering* **174** (2012), pp. 49–60.

[15] P. E. Bernard, N. Moës, and N. Chevaugeon. Damage growth modeling using the Thick Level Set (TLS) approach: Efficient discretization for quasi-static loadings. *Computer Methods in Applied Mechanics and Engineering* **233-236** (2012), pp. 11–27.

[16] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Envolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Material Science*. 2nd ed. Cambridge, United Kingdom: Cambridge University Press, 1999.

[17] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. 1st ed. New York, United States of America: Springer-Verlag, 2003.

[18] A. Salzman, N. Moës, and N. Chevaugeon. On use of the thick level set method in 3D quasi-static crack simulation of quasi-brittle material. *International Journal of Fracture* **202** (1) (2016), pp. 21–49.

[19] K. Moreau, N. Moës, D. Picart, and L. Stainier. Explicit dynamics with a non-local damage model using the thick level set approach. *International Journal for Numerical Methods in Engineering* **102** (3-4) (2015), pp. 808–838.

[20] M. Latifi, F. P. van der Meer, and L. J. Sluys. An interface thick level set model for simulating delamination in composites. *International Journal for Numerical Methods in Engineering* **111** (4) (2017), pp. 303–324.

[21] M. Latifi, F. P. van der Meer, and L. J. Sluys. Fatigue modeling in composites with the thick level set interface method. *Composites Part A: Applied Science and Manufacturing* **101** (2017), pp. 72–80.

[22] L.O. Voormeeren, F.P. van der Meer, J. Maljaars, and L.J. Sluys. A new method for fatigue life prediction based on the Thick Level Set approach. *Engineering Fracture Mechanics* **182** (2017), pp. 449–466.

[23] L. Niessen. Fatigue analysis using the The Thick Level Set method. Bachelor's thesis. Delft University of Technology, 2020.

[24] F. Cazes and N. Moës. Comparison of a phase-field model and of a thick level set model for brittle and quasi-brittle fracture. *International Journal for Numerical Methods in Engineering* **103** (2015), pp. 114–143.

[25] A. P. Gómes, N. Moës, and C. Stolz. Comparison between thick level set (TLS) and cohesive zone models. *Advanced Modeling and Simulation in Engineering Sciences* **2** (1) (2015), p. 18.

[26] C. E. Rogers. Investigating the Micromechanisms of Mode II Delamination in Composite Laminates. PhD thesis. Imperial College London, 2009.

[27] E. S. Greenhalgh, C. Rogers, and P. Robinson. Fractographic observations on delamination growth and the subsequent migration through the laminate. *Composite Science and Technology* **69** (14) (2009), pp. 2345–2351.

[28] T. K. O'Brien. Composite Interlaminar Shear Fracture Toughness, G_{IIc}: Shear measurement or sheer myth? *Composite Materials: Fatigue and Fracture* **7** (1998), pp. 3–18.

[29] F. P. van der Meer and L. J. Sluys. The Thick Level Set method: Sliding defomations and damage initiation. *Computer Methods in Applied Mechanics and Engineering* **285** (2015), pp. 64–82.

[30] A. Simone, G. N. Wells, and L. J. Sluys. From continuous to discontinuous failure in a gradient-enhanced continuum damage model. *Computer Methods in Applied Mechanics and Engineering* **192** (41) (2003), pp. 4581–4607.

[31] C. Comi, S. Mariani, and U. Perego. An extended FE strategy for transition from continuum damage to mode I cohesive crack propagation. *International Journal for Numerical and Analytical Methods in Geomechanics* **31** (2) (2007), pp. 213–238.

[32] E. Tamayo-Mas and A. Rodríguez-Ferran. A new continuous–discontinuous damage model: Cohesive cracks via an accurate energy-transfer process. *Theoretical and Applied Fracture Mechanics* **69** (2014), pp. 90–101.

[33] E. Tamayo-Mas, J. Feliu-Fabà, M. Casado-Antolin, and A. Rodríguez-Ferran. A continuous-discontinuous model for crack branching. *International Journal for Numerical Methods in Engineering* **120** (1) (2019), pp. 86–104.

[34] L. A. T. Mororó and F. P. van der Meer. Combining the thick level set method with plasticity. *European Journal of Mechanics - A/Solids* **79** (2020), p. 103857.

[35] K. Moreau, N. Moës, N. Chevaugeon, and A. Salzman. Concurrent development of local and non-local damage with the Thick Level Set approach: Implementation aspects and application to quasi-brittle failure. *Computer Methods in Applied Mechanics and Engineering* **327** (2017), pp. 306–326.

[36] A. P. Gómes, N. Moës, and C. Stolz. Comparison between thick level set (TLS) and cohesive zone models. *Advanced Modeling and Simulation in Engineering Sciences* **2** (1) (2015), p. 18.

[37] A. R. Melro, P. P. Camanho, F. M. Andrade Pires, and S. T. Pinho. Micromechanical analysis of polymer composites reinforced by unidirectional fibres: Part I - Constitutive modelling. *International Journal of Solids and Structures* **50** (11) (2013), pp. 1897–1905.

[38] E. A. Souza Neto, D. Perić, and D. R. J. Owen. *Computational Methods for Plasticity: Theory and Applications*. 1st ed. West Sussex, United Kingdom: John Wiley & Sons Ltd., 2008.

[39] F. P. van der Meer. Micromechanical validation of a mesomodel for plasticity in composites. *European Journal of Mechanics - A/Solids* **60** (2016), pp. 58–69.

[40] F. P. van der Meer, N. Moës, and L. J. Sluys. A level set model for delamination – Modeling crack growth without cohesive zone or stress singularity. *Engineering Fracture Mechanics* **79** (2012), pp. 191–212.

[41] J. Stewart, L. Redlin, and S. Watson. *Precalculus: Mathematics for Calculus*. 6th ed. Belmont, United States of America: Cengage Learning, 2013.

[42] J. D. Lawrence. *A Catalog of Special Plane Curves*. 1st ed. New York, United States of America: Dover Publications, 1972.

[43] G. N. Wells, L. J. Sluys, and R. de Borst. Simulating the propagation of displacement discontinuities in a regularized strain-softening medium. *International Journal for Numerical Methods in Engineering* **53** (2002), pp. 1235–1256.

[44] O. C. Zienkiewicz and J. Z. Zhu. The superconvergent patch recovery and *a posteriori* error estimates. Part I: The recovery technique. *International Journal for Numerical Methods in Engineering* **33** (1992), pp. 1331–1364.

[45] O. C. Zienkiewicz and J. Z. Zhu. The superconvergent patch recovery (SPR) and adaptive finite element refinement. *Computer Methods in Applied Mechanics and Engineering* **101** (1992), pp. 207–224.

[46] J. Shi, D. Chopp, J. Lua, N. Sukumar, and T. Belytschko. Abaqus implementation of extended finite element method using a level set presentation for three-dimensional fatigue crack growth and life prediction. *Engineering Fracture Mechanics* **77** (2010), pp. 2840–2863.

[47] J. C. Simo and T. J. R. Hughes. *Computational Inelasticity*. 1st ed. New York, United States of America: Springer-Verlag, 1998.

[48] Christophe Geuzaine and Jean-François Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering* **79** (11) (2009), pp. 1309–1331.

[49] H. Li, M. W. Fu, J. Lu, and H. Yang. Ductile fracture: Experiments and computations. *International Journal of Plasticity* **27** (2) (2011), pp. 147–180.

[50] C. Miehe, F. Aldakheel, and A. Raina. Phase field modeling of ductile fracture at finite strains: A variational gradient-extended plasticity-damage theory. *International Journal of Plasticity* **84** (2016), pp. 1–32.

[51] H. Badnava, E. Etermadi, and M. A. Msekh. A Phase Field Model for Rate-Dependent Ductile Fracture. *Metals* **7** (5) (2017), p. 180.

[52] L. A. T. Mororó and F. P. van der Meer. Parallel Computing with the Thick Level Set Method. *SIAM Journal on Scientific Computing* **43** (6) (2021), pp. C386–C410.

[53] I. B. C. M. Rocha, F. P. van der Meer, L. A. T. Mororó, and L. J. Sluys. Accelerating crack growth simulations through adaptive model order reduction. *International Journal for Numerical Methods in Engineering* **121** (10) (2020), pp. 2147–2173.

[54] H. Wafai, A. Yudhanto, G. Lubineau, R. Yaldiz, and N. Verghese. An experimental approach that assesses in-situ micro-scale damage mechanisms and fracture toughness in thermoplastic laminates under out-of-plane loading. *Composite Structures* **207** (2019), pp. 546–559.

[55] Y. Saad. *Iterative Methods for Sparse Linear Systems*. 2nd ed. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2003.

[56] P. Gosselet and C. Rey. Non-overlapping Domain Decomposition Methods in Structural Mechanics. *Archives of Computational Methods in Engineering* **13** (4) (2006), pp. 515–572.

[57] V. Dolean, P. Jolivet, and F. Nataf. *An Introduction to Domain Decomposition Methods*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2015.

[58] C. Farhat and F.-X. Roux. A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering* **32** (6) (1991), pp. 1205–1227.

[59] P. Le Tallec, Y. H. De Roeck, and M. Vidrascu. Domain decomposition methods for large linearly elliptic three-dimensional problems. *Journal of Computational and Applied Mathematics* **34** (1) (1991), pp. 93–117.

[60] F. J. Lingen, P. G Bonnier, R. B. J. Brinkgreve, M. B. Van Gijzen, and C. Vuik. A parallel linear solver exploiting the physical properties of the underlying mechanical problem. *Computational Geosciences* **18** (6) (2014), pp. 913–926.

[61] MPI Forum. MPI: A Message-Passing Interface Standard. Version 3.1. Available at: `http://www.mpi-forum.org`, (Accessed: Dec. 2019). June 2015.

[62] E. W. Remij. Fluid driven and mechanically induced fracture propagation: theory and numerical simulations. PhD thesis. Eindhoven University of Technology, 2017.

[63] F. J. Lingen. Design of an Object Oriented Finite Element Package for Parallel Computers. PhD thesis. Delft University of Technology, 2000.

[64] Richard. Barrett, Michael. Berry, Tony F. Chan, James. Demmel, June. Donato, Jack. Dongarra, Victor. Eijkhout, Roldan. Pozo, Charles. Romine, and Henk. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. Society for Industrial and Applied Mathematics, 1994.

[65] Jem-Jive - Software development kit for advanced numerical simulations. `http://jive.dynaflow.com`. Accessed: 10-12-2019.

[66] Thomas Adams, Stefano Giani, and William M. Coombs. A high-order elliptic PDE based level set reinitialisation method using a discontinuous Galerkin discretisation. *Journal of Computational Physics* **379** (2019), pp. 373–391.

[67] Richard Dekker, Frans P. van der Meer, Johan Maljaars, and Lambertus J. Sluys. A level set model for stress-dependent corrosion pit propagation. *International Journal for Numerical Methods in Engineering* **122** (8) (2021), pp. 2057–2074.

[68] A. Biel and Ulf Stigh. Strength and toughness in shear of constrained layers. *International Journal of Solids and Structures* **138** (2018), pp. 50–63.

[69] L. A. T. Mororó, A. Poot, and F. P. van der Meer. Skeleton curve and phantom node method for the Thick Level Set approach to fracture. *Engineering Fracture Mechanics* **268** (2022), p. 108443.

[70] J. Ma, S. W. Bae, and S. Choi. 3D medial axis point approximation using nearest neighbors and the normal field. *The Visual Computer* **28** (2012), pp. 7–19.

[71] R. Y. Peters. Geographical point cloud modelling with the 3D medial axis transform. PhD thesis. Delft University of Technology, 2018.

[72] F. P. van der Meer and L. J. Sluys. A phantom node formulation with mixed mode cohesive law for splitting in laminates. *International Journal of Fracture* **158** (2) (2009), pp. 107–124.

[73] A. Turon, P.P. Camanho, J. Costa, and C.G. Dávila. A damage model for the simulation of delamination in advanced composites under variable-mode loading. *Mechanics of Materials* **38** (11) (2006), pp. 1072–1089.

[74] F. P. van der Meer, L. J. Sluys, S. R. Hallett, and M. R. Wisnom. Computational modeling of complex failure mechanisms in laminates. *Journal of Composite Materials* **46** (5) (2012), pp. 603–623.

[75] Florian Levet and Xavier Granier. Improved Skeleton Extraction and Surface Generation for Sketch-Based Modeling. In: *Proceedings of Graphics Interface 2007*. GI '07. Montreal, Canada, May 2007, pp. 27–33.

[76] A. Tagliasacchi, T. Delame, M. Spagnuolo, N. Amenta, and A. Telea. 3D Skeletons: A State-of-the-Art Report. *Computer Graphics Forum* **35** (2) (2016), pp. 573–597.

[77] K. Siddiqi and S. M. Pizer. *Medial Representation: Mathematics, Algorithms and Applications*. Springer, 2008.

[78] Harry Blum. Discussion paper: A geometry for biology. *Annals of the New York Academy of Sciences* **231** (1) (1974), pp. 19–30.

[79] Boost C++ Libraries. `http://www.boost.org/`. Accessed: 20-07-2021.

[80] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithm and Applications*. 3rd ed. Springer, 2008.

[81] Jeremy Kepner and John R. Gilbert. *Graph Algorithms in the Language of Linear Algebra*. Vol. 22. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2011.

[82] Hana Samet. *Foundations of Multidimensional and Metric Data Structures*. San Francisco, CA: Morgan Kaufmann, 2006.

[83] A. Poot. Laying the Foundations for a New Thick Level Set Method. Master's thesis. Delft University of Technology, 2020.

[84] T. Delame, J. Kustra, and A. Telea. Structuring 3D Medial Skeletons: A Comparative Study. In: *Proceedings of the Conference on Vision, Modeling and Visualization*. VMV '16. Bayreuth, Germany: Eurographics Association, 2016, pp. 1–8. ISBN: 9783038680253.

CURRICULUM VITÆ

Luiz Antonio TAUMATURGO MORORÓ

08-10-1988 Born in Fortaleza, Brazil.

EDUCATION

2006–2010 *Bachelor in Civil Engineering*
Department of Civil Engineering
Federal University of Ceará, Brazil
Thesis: Global analysis of composite tubes
Supervisor: Prof. dr. Evandro Parente Junior

2011–2013 *Master of Science in Computational Mechanics*
Department of Structural Engineering
Federal University of Ceará, Brazil
Thesis: Geometrically nonlinear analysis of thin-walled laminated beams
Supervisor: Prof. dr. Antônio Macário Cartaxo de Melo

2017–2023 *PhD candidate in Computational Mechanics*
Department of Materials, Mechanics, Management & Design
Faculty of Civil Engineering and Geosciences
Delft University of Technology, the Netherlands
Thesis: Extending the Thick Level Set approach: Plasticity, Parallel computing, and Cohesive cracks
Promotors: Dr. ir. F. P. van der Meer and Prof. dr. ir. L. J. Sluys

EXPERIENCE

2012–2013 *Master research assistant*
Department of Structural Engineering
Federal University of Ceará, Brazil

2014–2015 *Structural analyst*
RCM - Structural Engineering, Brazil

2014–Present *Lecturer in Civil and Structural Engineering*
Federal Institute of Education, Science, and Technology of Ceará, Brazil

List of Publications

Journal articles

3. **L. A. T. Mororó**, A. Poot, and F. P. van der Meer. Skeleton curve and phantom node method for the Thick Level Set approach to fracture. *Engineering Fracture Mechanics* **268** (2022), p. 108443.

2. **L. A. T. Mororó** and F. P. van der Meer. Parallel Computing with the Thick Level Set Metho. *SIAM Journal on Scientific Computing* **43** (6) (2021), pp. C386–C410.

1. **L. A. T. Mororó** and F. P. van der Meer. Combining the Thick Level Set method with plasticity. *European Journal of Mechanics - A/Solids* **79** (2020), p. 103857.

Co-authored journal articles

1. I. B. C. M. Rocha, F. P. van der Meer, **L. A. T. Mororó**, and L. J. Sluys. Accelerating crack growth simulations through adaptive model order reduction. *International Journal for Numerical Methods in Engineering* **121** (10) (2020), pp. 2147–2173.

Conference articles

1. **L. A. T. Mororó** and F. P. van der Meer. Simulation of cusp formation in composite materials using the thick level set method. In: *18th European Conference on Composite Materials (ECCM 2018)*. Athens, Greece, 24–28th June 2018.

Abstract conference articles

3. **L. A. T. Mororó** and F. P. van der Meer. A new version of the Thick Level Set method (TLSV2): skeleton curve and phantom node method. In: *11th European Solid Mechanics Conference*. Galway, Ireland, 4–8th July 2022.

2. **L. A. T. Mororó** and F. P. van der Meer. Parallel Computing with the Thick Level Set Method. In: *14th World Congress on Computational Mechanics (WCCM XIV)*. Virtual Congress, 11–15th January 2021.

1. **L. A. T. Mororó** and F. P. van der Meer. Simulation of Cusp Formation in a Polymer Matrix Loaded in Shear Using the Thick Level Set Method with Plasticity. In: *VI International Conference on Computational Modeling of Fracture and Failure of Materials and Structures (CFRAC 2019)*. Braunschweig, Germany, 12–14th June 2019.

PROPOSITIONS

accompanying the dissertation

EXTENDING THE THICK LEVEL SET APPROACH

PLASTICITY, PARALLEL COMPUTING AND COHESIVE CRACKS

by

Luiz Antonio TAUMATURGO MORORÓ

1. You can only affirm that you have well understood a numerical model when you are able to implement it completely.
2. Focusing too much on building numerical tools can keep you away from the actual problem that has originally motivated the development of such tools.
3. Turning a sequential implementation into a corresponding parallel one can be harder than implementing the same parallel framework from scratch.
4. A powerful computer cluster can be useless if your parallel implementation relies primarily on collective-like routines.
5. A PhD journey can be less tortuous by using the *divide-and-conquer* concept from domain decomposition methods.
6. Despite having more potentialities, the Thick Level Set method is less popular than phase field approaches.
7. Fixing a memory-related error in a C++ code within a parallel environment is as thrilling as celebrating a goal of your favorite football team.
8. Most people think all Brazilians are good at football – in fact, the majority of them are.

These propositions are regarded as opposable and defendable, and have been approved as such by promotors Dr. ir. F. P. van der Meer and Prof. dr. ir. L. J. Sluys.

STELLINGEN

behorende bij het proefschrift

EXTENDING THE THICK LEVEL SET APPROACH

PLASTICITY, PARALLEL COMPUTING AND COHESIVE CRACKS

door

Luiz Antonio TAUMATURGO MORORÓ

1. Je kan alleen stellen dat je een numeriek model doorgrond hebt wanneer je in staat bent het helemaal te implementeren.
2. Een te sterke focus op het ontwikkelen van numerieke gereedschappen kan je afhouden van het eigenlijke probleem dat oorspronkelijk aangezet heeft tot de ontwikkeling van deze gereedschappen.
3. Een sequentiële implementatie ombouwen tot een parallelle kan moeilijker zijn dan hetzelfde parallelle raamwerk helemaal opnieuw te implementeren.
4. Een krachtig rekencluster kan nutteloos zijn als je parallelle implementatie primair op gedeelde routines leunt.
5. De reis van een promotie kan minder kronkelig gemaakt worden door het *verdeel-en-heers* principe van domein-decompositie methodes.
6. Ondanks een surplus aan mogelijkheden is de *Thick Level Set* methode minder populair dan *phase field* benaderingen.
7. Het herstellen van een geheugen-gerelateerde fout in C++ code in een parallelle omgeving is even opwindend als het vieren van een doelpunt van je favoriete voetbalteam.
8. De meeste mensen denken dat alle Brazilianen goed zijn in voetbal – in feite zijn de meesten dat.

Deze stellingen worden opponeerbaar en verdedigbaar geacht en zijn als zodanig goedgekeurd door de promotoren Dr. ir. F. P. van der Meer en Prof. dr. ir. L. J. Sluys.