Knowledge Graph for CAE-based Development in Vehicle Safety

Dissertation
to obtain a doctoral degree

in the
School of Mechanical Engineering and Safety Engineering

University of Wuppertal

Submitted by:
Anahita Pakiman
from Hamedan [Iran]

Wuppertal 2023

"Aus der Kriegsschule des Lebens.—Was mich nicht umbringt, macht mich stärker," which can be translated as "Out of life's school of war—what doesn't kill me, makes me stronger."

Friedrich Nietzsche

Abstract

This dissertation presents a comprehensive exploration of Knowledge Graphs (KGs) to transform the domain of vehicle development, with a primary focus on improving crashworthiness. The proposed KG leverages insights from diverse structured and unstructured data sources, encompassing critical concepts within the automotive domain. This research serves the industrial goal of capturing and preserving knowledge derived from Computer-Aided Engineering (CAE) workflows, enabling data reuse, improving guidelines, and facilitating Machine Learning (ML) applications.

Creating a domain-specific KG is challenging, especially when dealing with complex CAE data with intricate 3D deformations. This complexity is rarely encountered in most existing text-based KGs. This thesis presents a comprehensive path from data modelling to practical ML applications based on a concrete use case to address this challenge. This approach provides a cyclical feedback loop between data modelling and feature extraction, enhancing the utility of both processes.

The research encompasses several key aspects, including data modelling and ontology development, feature engineering, data visualisation for knowledge discovery, and ML implementations to predict relationships between simulations. It innovatively transforms crashworthiness analysis into graph representations, uses SimRank to analyse weighted bipartite graphs, and abstracts vehicle structures to rank simulations based on load-path similarity.

Additionally, a user-friendly web application for this project is developed using Django and hosted on GitHub[1], ensuring accessibility and ease of use for industry professionals and researchers alike. This work highlights the immense potential of KGs in the automotive sector, facilitating a data-driven approach to vehicle development and significantly improving crashworthiness analysis through the integration of CAE data.

[1] https://github.com/Fraunhofer-SCAI/GAE-vehicle-safety/

Zusammenfassung

Diese Dissertation stellt eine umfassende Untersuchung von Wissensgraphen (Knowledge Graphs, KGs) vor, um den Bereich der Fahrzeugentwicklung zu verändern, wobei der Schwerpunkt auf der Verbesserung der Crashsicherheit liegt. Der vorgeschlagene KG nutzt Wissen aus verschiedenen strukturierten und unstrukturierten Datenquellen und beinhaltet kritische Konzepte in der Automobilindustrie. Diese Forschung dient dem industriellen Ziel, Wissen aus Computer-Aided Engineering (CAE) Workflows zu erfassen und zu bewahren, um Daten wiederzuverwenden, Richtlinien zu verbessern und die Anwendung von maschinellem Lernen (ML) zu unterstützen.

Die Erstellung eines domänenspezifischen KG ist eine anspruchsvolle Aufgabe, insbesondere wenn komplexe CAE-Daten mit komplizierten 3D-Verformungen verarbeitet werden müssen. Diese Komplexität ist in den meisten existierenden textbasierten KGs selten anzutreffen. Um dieser Herausforderung zu begegnen, beschreibt diese Dissertation einen umfassenden Weg von der Datenmodellierung bis hin zu praktischen ML-Anwendungen, die auf einem konkreten Anwendungsfall basieren. Dieser Ansatz ermoeglicht eine zyklische Rückkopplung zwischen Datenmodellierung und Merkmalsextraktion und verbessert den Nutzen beider Prozesse.

Die Forschung umfasst mehrere Schwerpunkte, darunter Datenmodellierung und Ontologieentwicklung, Merkmalsextraktion, Datenvisualisierung zur Wissensentdeckung und die Implementierung von ML zur Vorhersage von Beziehungen zwischen Simulationen. Das Projekt setzt Crash-Sicherheitsanalysen innovativ in graphische Darstellungen

um, nutzt SimRank zur Analyse gewichteter bipartiter Graphen und abstrahiert Fahrzeugstrukturen, um Simulationen nach Ähnlichkeit der Lastpfade zu ordnen.

Darüber hinaus wurde eine benutzerfreundliche Webanwendung für dieses Projekt entwickelt, die mit Django erstellt und auf GitHub[2] gehostet wurde, um die Zugänglichkeit und Benutzerfreundlichkeit für Industrieexperten und Forscher zu gewährleisten. Diese Arbeit unterstreicht das enorme Potenzial von KGs in der Automobilindustrie, die einen datengetriebenen Ansatz in der Fahrzeugentwicklung ermöglichen und die Crash-Sicherheitsanalyse durch die Integration von CAE-Daten erheblich verbessern.

[2] `https://github.com/Fraunhofer-SCAI/GAE-vehicle-safety/`

Acknowledgements

Looking back on my journey, it's challenging to single out just a few individuals who have been unwavering pillars of support. Moreover, I find myself profoundly grateful for the opportunity to pursue higher education during times marked by women's protests in Iran, heartbreaking stories of chemical attacks on girls' schools, and the stifling ban on women's education in Afghanistan.

Foremost, my deepest love and gratitude are reserved for my parents, who never hesitated to stand by me and embrace the challenge of sending me abroad for a brighter future. My father's passion for engineering ignited my own, and my mother stands as a paragon of courage and diligence. Despite the formidable challenges they faced, they made tremendous sacrifices to provide me with an exceptional education.

I must also extend my heartfelt appreciation to Manzoume Kherad High School, where I had a transformative educational experience. This period remains the most luminous chapter in my academic journey, thanks to my mother's tireless research in finding that remarkable institution. The values instilled in us there—being a responsible citizen, teamwork, and a scientific outlook—have underpinned all my subsequent achievements.

The indelible memories of that period continue to drive each of us towards a place filled with the joy of teamwork, discovery, creativity, exploration, and art. Among the exceptional educators, Mariam Mokhtari stands out for her boundless energy and insatiable curiosity in the realm of physics, which profoundly influenced my scientific journey.

Transitioning to the world of industry, it was Jens Weber, my Master's thesis supervisor, who rekindled my belief in finding the same motivation and enjoyment. Moving on to my doctoral work, the inception of my research took root at CEVT (China Euro Vehicle Technology AB). I owe a debt of gratitude to Hans Merkle and Martin Larson for their invaluable mentorship, which enriched my programming skills and semantic reporting expertise. Special thanks are also due to Dag Thuvesen for introducing me to the world of machine learning and helping shape my thesis. Dag's efforts in facilitating access to CEVT's data for my research were instrumental.

When financial support for my idea at CEVT proved elusive, my dearest friends Mattias Ericsson, Zahra Alikahi, and Farzin Jahangiri provided unwavering encouragement, inspiring me to explore alternative avenues.

Last but certainly not least, my heartfelt appreciation goes to my advisors, Prof. Dr. Jochen Garcke and Prof. Dr.-Ing. Axel Schumacher, as well as Dr. Victor Rodrigo Iza Teran, Dr. Daniela Steffes-lai, and Mandar Satyanath Pathare. Their unwavering support, intellectual engagement, and incisive questioning were instrumental in nurturing and refining my research idea. They pushed me to grow, and I take pride in what I've accomplished, knowing that it wouldn't have been possible without their guidance and assistance.

Contents

List of Abbreviations

2D two-dimensional

3D three-dimensional

AD Autonomous Driving

ADAS Advanced Driver Assistance Systems

AVs Autonomous Vehicles

CAD Computer-Aided Design

CAE Computer-Aided Engineering

CAM Computer-Aided Manufacturing

CBG Component-Based Graph

CEVT China Euro Vehicle Technology AB

COG Center of Gravity

DEF Design Exploration Fingerprint

DSM Deformation Space Model

FE Finite Element

FEM Finite Element Method

FMVSS Federal Motor Vehicle Safety Standard

GAE Graph-Aided Engineering

GNN Graph Neural Network

GML Graph Machine Learning

HAV Highly Automated Vehicle

IE Internal Energy

IE_{max} max of IE

IIHS Insurance Institute for Highway Safety

KBE Knowledge-Based Engineering

KDE Kernel Density Estimation

KE Kinetic Energy

KG Knowledge Graph

LHS Left Hand Side

LLM Large Language Model

LM Language Model

LMS Lumped Mass Spring

ML Machine Learning

mPBG multi Part-Based Graph

NCAP New Car Assessment Program

NVH Noise Vibration Harshnesss

PID Property ID

P_e Power of energy absorption

OWL Web Ontology Language

RBE Rigid Body Elements

RDF Resource Description Framework

RDFs RDF schema

RHS Right Hand Side

RMSE Root Mean Square Error

s SimRank

s^{++}_{trgt} SimRankTarget++

s_w weighted SimRank

$s_{w,evd}$ SimRank with evidence

s^{++} weighted SimRank with evidence and spread

SISAME Structural Impact Simulation And Model-Extraction

sPBG single Part-Based Graph

SWT Semantic Web Technologies

t_i initial absorption time

t_n final absorption time

t_{max} last time step

t_{all} all the time steps

List of Equations

Dedicated as a reminder of impermanence
the source of creativity…

1 Introduction

1.1 New data representation for CAE

Over the past 30 years, the focus in vehicle crash simulation has been on the reliability of the Finite Element (FE) method in predicting crash behaviour. This has led to more and more detailed simulations, which have increased the accuracy of the predictions made by the simulations. The increased level of detail in the simulations is an indication of the increased complexity of the data. Therefore, identifying and summarising cause-effect relationships for complex simulations is time-consuming. Using a new representation of the data, it is advantageous to describe the vehicle development process as a sequence of cause-and-effect relationships.

In addition to complexity, the number of simulations has increased enormously, also driven by increased computational power. This escalation has widened the gap between the information described in reports and the information available in the simulations. The more complex simulations, the more time engineers have to spend analysing and reporting the results. As the amount of data increases, the limited amount of engineering time available means that more and more of the data is left unexplored. Therefore, automating the post-processing of analyses has been a great help in overcoming the challenge of exploring data. These scripts are mostly used to extract design performances, such as predicting material properties, e.g. stress, strain and failure. These post-processing scripts have greatly facilitated the engineer's workflow.

However, while automated workflows offer significant benefits, they are

still limited by the lack of semantic connections that can link different analyses together. Semantics play a key role in unlocking the knowledge contained within these analyses. Consequently, there is an urgent need to enhance data representation by providing it with semantics, thereby facilitating meaningful relationships between data elements. In pursuing this goal, computational efficiency becomes a critical consideration, as the system must be able to compare a large number of simulations.

An illustrative example of the benefits of linking simulations is the refinement of design guidelines. Typically, each company formulates its design guidelines based on accumulated engineering experience, translating human injury into structural deformation criteria. These guidelines include factors such as force limits for structural components and intrusion values, along with their respective sequences. However, the design guidelines are often not based on all the analysis that has been during development but rather on a global standard and are, therefore, relatively rough. In addition, the continuous generation and refinement of these guidelines requires a high level of engineering expertise, which can be difficult to sustain within an organisation. As a result, the creation of additional data connectivity is imperative, enabling the identification of cause-and-effect relationships and facilitating the generalisation of design guidelines.

Therefore, the definition of a new data representation based on appropriate engineering principles for the quantification of crash behaviour is of great benefit to the vehicle development process, which is considered to be a largely unexplored area of research. The new data representation allows organisations to move from static engineer reporting to a networked dataset that extracts more features from the data to define design guidelines. In this way, the new data representation will improve knowledge transfer between vehicle development projects. Additionally, such data representation characterising vehicle crash performance will also help Machine Learning (ML) algorithms capture engineering knowledge. The outcome of this research may lead to a breakthrough in vehicle safety analysis processes.

1.2 CAE past, present and future

Mechanical engineering is a field that aims to design, e.g. structures, systems or machines, to find the best solutions to meet the design requirements. This goal has given rise to many topics in the field, all of which have in common the prediction of the mechanical behaviour of the case study. Later, the Finite Element Method (FEM) was born from the need to analyse complex structural and elasticity problems in engineering. The FEM was first developed in 1943 to obtain approximate solutions to vibrating systems [EI17]. The method of analysis in which the field equations of mathematical physics are approximated over simple domains –triangular, quadrilateral, tetrahedral, and so on– and then assembled so that equilibrium or continuity is satisfied at the connecting nodes of the domains [GM96]. A broader definition of numerical analysis was introduced shortly afterwards. However, with several advantages, such as the closeness of the actual structure to the FE model and the ability to mix different element types in the mesh matrix [EI17], FEM remains superior to others.

Subsequently, with the growth of computing power and the advantages of FEM over other numerical analyses, FEM simulation of components and full vehicle functions is becoming state of the art in the automotive industry and is the basis for short development processes [Häg+10]. As a result, FE-simulation allows a reduction in traditional test-based approaches and also enables the study of full vehicle functions earlier in the design process [Häg+10]. The establishment of a new generation of this new type of analysis has shaped Computer-Aided Engineering (CAE) in various industries. However, it wasn't the automotive industry that started this analysis, but the high market demand made the automotive industry a pioneer in the development of methods and the amount of data generated. Today, the number of CAE simulations at automotive OEMs is between 10,000 and 30,000 per week [Sch22]. This volume of data makes CAE in the automotive R&D process one of the engineering domains that can significantly benefit from ML, e.g. Knowledge Graph (KG).

1.2.1 Crashworthiness history

According to the History of Crashworthiness by Tawfik [DB+04], the first fatal accident involving a motor vehicle took place in New York City in 1889. This led to the establishment of research into automotive safety. Over the past century, occupant safety has become a critical criterion in vehicle design. The history of automotive safety can be divided into three distinct periods. In the first and second periods, before 1935 and from 1935 to 1965, the focus was on fundamental improvements and manufacturers introduced many crash avoidance devices. This period saw the introduction of turn signals, dual windscreen wipers, rear-view mirrors and laminated glass to reduce facial injuries. The most significant safety feature of this period was the introduction of seat belts as an option in 1956. The third period saw the introduction of vehicle safety legislation in 1966. This led to improvements in the crashworthiness of the vehicle structure and more effective restraint systems. Examples include Federal Motor Vehicle Safety Standard (FMVSS), New Car Assessment Program (NCAP), Insurance Institute for Highway Safety (IIHS), compatibility tests and tests to ensure the protection of child and adult occupants [DB+04].

The introduction of crash testing was part of this evolution. The first full-scale crash tests were carried out in the early 1930s. These tests included rollover and car-to-barrier [DB+04]. The high cost of these tests drove the demand for CAE analysis in this area. Non-linear FE-models were introduced in the mid-1980s and rapidly gained acceptance among structural analysts [DB+04]. However, the transition within vehicle safety has been a long process due to the high non-linearity of the simulations. However, a major factor in the growth of CAE analysis for crash simulation has been the high cost of crash testing and the late incorporation of testing into the vehicle development process. Figure 1.1 summarises the evolution of CAE analysis from its early beginning to its full integration into vehicle R&D.

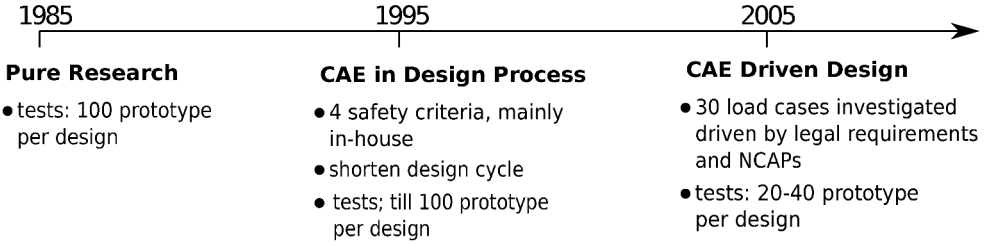


FIGURE 1.1: CAE evolvement into vehicle development processes [BFG05].

1.2.2 Crash simulation complexity

Structural dynamics FEM is the numerical solution of a set of nonlinear partial differential equations of motion in space and time, coupled with material stress-strain equations and appropriate initial and boundary conditions. The solution first discretises the equations in space, which is done by formulating the problem in a weak variational form and assuming an admissible displacement field. This leads to a set of second-order differential equations in time. The time domain discretisation then solves the system of equations. This is done by discretising according to the classical Newmark-Beta method [New59]. The method is called implicit if the chosen integration parameters make the equations coupled. In this case, the solution is unconditionally stable. The solution is called explicit and is conditionally stable if the integration parameters are chosen to decouple the equations. Implicit solutions were mainly used in earlier developments of the nonlinear FE technique [Far70]. Belytschko [Hau83] appears to have first introduced FE-simulation for structural crashworthiness using explicit solvers.

The explicit FE technique for crashworthiness solves a set of hyperbolic wave equations, stress propagation, in the zone of influence of the wavefront and, accordingly, does not require the coupling of a large number of equations. On the other hand, the unconditionally stable implicit

solvers provide a solution to all coupled equations of motion, which requires the construction of a global stiffness matrix. The time step for implicit solvers is approximately two to three orders of magnitude smaller than the explicit time step. For crash simulations involving extensive use of contact, multiple material models, and a combination of non-traditional elements, explicit solvers have been found to be more robust and computationally efficient than implicit solvers [DB+04].

Therefore, vehicle crashworthiness analysis is one of the most challenging non-linear problems in structural mechanics. Vehicle structures typically consist of many stamped and formed thin sheet metal parts. They are then assembled using a variety of welding and joining techniques. Different strengths of steel, aluminium and composite materials may be used in the body-in-white. In the event of a crash, the structure is subjected to high-impact loads. This results in local plastic deformation and buckling. Finally, the contact and stacking of the various components can lead to large deformations and rotations of the structure. These deformations are first-wave effects associated with high stresses. Once these stresses exceed the yield strength and critical buckling load of the material, local deformations occur during a few waves that pass through the structure. Subsequently, inertial effects dominate the subsequent transient response. The integrity of the structure and the associated kinematics and stacking of components are of particular interest here, as the forces transmitted through the different members, the stresses, the strains and the energy absorption. Closed-form analytical solutions to this class of structural mechanics problems present a considerable understatement. Numerical techniques appear to be the practical option at present [DB+04].

1.2.3 CAE workflow limitations

The importance of vehicle safety has grown enormously since the introduction of safety regulations. This growth has been supported by the integration of CAE crashworthiness analysis into vehicle R&D. Crash simulations have become more complex, and the number of simulations

has increased as computing power has increased. However, compared to the full range of real-world crashes, the number of crash scenarios studied today is still limited. The field is therefore facing new challenges.

As the number and complexity of simulations increased, engineers needed more time to analyse each simulation. Several workflows have been defined to help CAE engineers spend more time analysing the data than preparing it. However, linking these analyses and synthesising the simulations still requires a lot of work. One of the challenges is the limited ability of engineers to generalise the simulations to transfer knowledge from one project to another. The ability to extract knowledge from the CAE simulations allows the transfer of knowledge from existing simulations to a wider range of crash scenarios or occupant diversity. Therefore, the importance of data has been emphasised to enable more appropriate reuse of CAE analyses.

Manufacturers currently rely on internally developed design guidelines to ensure the performance and compliance of new vehicles with today's standards. These guidelines have been developed over many years through a process of iterative design and CAE analysis. They serve as a reliable foundation from which carmakers can start to develop new vehicles, incorporating lessons learned from previous projects. However, it's important to note that these guidelines have inherent limitations, primarily due to their reliance on human understanding and their focus on the basic vehicle concept. They do not cover the full range of issues that CAE engineers investigate. In addition, due to the critical importance of crashworthiness in vehicle design, these guidelines tend to include substantial safety margins. As a result, they can lead to unintended uniformity in vehicle appearance across brands, hinder efforts to reduce vehicle weight and restrict the compact arrangement of vehicle components.

In conclusion, due to the need for more analysis, cost constraints, limited engineering time, and challenges in internal knowledge transfer underscore the significant potential of developing an automated knowledge

extraction method for the CAE domain. Therefore, extending the FE method with the power of data science to improve domain knowledge extraction is likely to be the next coming era of CAE. In this thesis, the introduction of semantics to integrate the analysis enables the capture of domain knowledge as an intelligent system to leverage CAE capabilities for the automotive industry.

1.3 Knowledge graph for CAE

The term KG, coined by Google in 2012, refers to any kind of knowledge that is based on graphs. Among the next wave of technologies [Ngu21], KGs are one of the key trends. KGs have become a new form of knowledge representation and are the cornerstone of several applications for specific use cases in the industry. Its underlying abstract structure, which effectively facilitates domain conceptualisation, data management and use as a preliminary driver for several applications of Artificial Intelligence [AS21], is the reason for the growing interest in this technology.

Beyond generic and open KGs, most current KGs are domain-specific, with specific ontologies underlying their design [Li+20]. An ontology, in computing, is a formal representation of the meanings of terms within a specific context, and its adoption within a KG promotes consistent terminology and modelling in that KG [Hog+21]. As there is no "one size fits all" schema or ontology that can be applied to address real-world problems, efforts continue to establish, refine and extend domain-specific KGs in different knowledge domains [Kej19]. A domain knowledge graph is an explicit conceptualisation of a high-level domain and its specific sub-domains. It is expressed as semantically related entities and relations [AS21]. The aim of this research is to extend KG to CAE crash simulation, called car-graph.

1.3.1 Application: Crash scenario recommendation

A multi-vehicle collision is a traffic accident involving a large number of vehicles. The number of crumpled vehicles is highly dependent on the traffic situation and the driving style [SN13]. Research shows that Autonomous Vehicles (AVs) have the potential to improve road safety by eliminating human error [DC20] and optimising traffic congestion [TT21]. The modelling of traffic flow started early with the modelling of the physics of traffic flow [Ker99] and is still a topic of interest with the consideration of AVs [Muz+22]. The existing models for AVs are aimed at avoiding multiple vehicle collisions. However, the vision zero of these methods is difficult to achieve in practice[1]. What is missing from these achievements is the exploitation of crash safety by using the safety characteristics of the vehicle involved in a multi-vehicle collision to make the safest possible decision. Therefore, this work introduces the use of CAE data in AVs modelling.

The safety characteristic is a combination of active and passive safety features. Passive safety includes the structural design of the vehicle, which is the focus of CAE analysis. Active safety, on the other hand, includes software-related features that improve safety during driving. Initially, active and passive safety were developed independently. Active safety acts as an information signal to the driver or the braking system. More recently, there has been an increase in interaction between these two areas. For example, the deceleration rate, the belt force, and passive safety control can be adjusted based on the input from active safety. However, there is still room for improvement in the interaction, and a major achievement will be to include the interaction between several vehicles.

An essential step in the networking of vehicle safety is the existence of an ontology that enables communications between vehicles. In

[1]Vision Zero (VZ) is a public programme aimed at eliminating road deaths and serious injuries. The idea was first introduced to the Swedish Parliament in 1997, and the programme has helped to reduce the number of fatalities, but the number of accidents is still far from zero.

recent years, the focus has been on knowledge-based approaches to the representation of scenarios to support scenario-based evaluation of Advanced Driver Assistance Systems (ADAS)s and Autonomous Driving (AD). Ontologies have become a key component for formalising this knowledge. Scenario representation is also a key aspect of ADASs/AD development and testing [Urb+21]. Rich and realistic scenario representations can lead to the generation of simulated environments that can be used in virtual test procedures (e.g. hardware-in-the-loop simulations). Scenarios may include descriptions of the participants in a road or driving scene, including their interactions, spatiotemporal relationships, etc. [Urb+21].

Optimal understanding of environmental sensing is an essential part of highway ontology, and mostly, the understanding depends on the perception of uncertain traffic conditions in a group of vehicles [Kam+21]. The benefit of understanding the environment would allow multiple vehicles to operate autonomously at high speeds in hazardous conditions and complex highway or urban settings or to perform cooperative manoeuvres [Che+17]. Typically, the initial collisions disrupt the flow of traffic, blocking the road and causing severe congestion. The blockage can sometimes lead to secondary collisions or chain collisions. They are among the worst types of traffic collisions, mostly on high-speed and high-capacity roads, including motorways [Muz+22].

To the best of available knowledge, scenario analyses typically do not include vehicle characteristics related to passive safety, which underlines the need to improve the knowledge of the environment to support the implementation of the safest solutions for multi-vehicle collisions. For example, incorporating details of the structural stiffness of the involved vehicles can improve the ability to effectively use passive safety knowledge. In addition, consideration of the structural characteristics of the environment, such as barrier properties (e.g. crash barrier stiffness), rigid pole placement and road user ergonomics, may also prove beneficial. Unfortunately, there is a large gap between AVs and vehicle passive safety performance in existing ontologies and scenario studies.

The introduction of the Graph-Aided Engineering (GAE) ontology is a first step towards bridging this gap. Below are a number of demonstration examples which may serve as inspiration for further progress in this area.

The first example is a motorway multi-vehicle crash, Figure 1.2. In this situation, there are three vehicles with considerable differences in structure: a heavy goods vehicle, a small compact car and a family car. To the best of available knowledge, there is no control within AD that takes into account the size of the vehicles in the control of braking. In the event that it is not possible for all the vehicles to stop in order to avoid the collision, it will be essential to act selectively on the collision. From a passive safety point of view, a compact car has much less space in front to absorb crash forces than a family car. Consequently, scenario 1 in Figure 1.2 can cause a fatal crash. However, if the driver were to stop the vehicle earlier to be on the safe side, it could cause a rear-end collision with the family car, Figure 1.2. A rear-end collision is safer for the compact car because a rear-end collision is less likely to be fatal than a frontal collision, and the family car has sufficient structural support to keep the driver's compartment safe.

Consider the case of a collision between a car and a pedestrian. In Europe, the majority of cars are tested by Euro NCAP, which includes an evaluation of pedestrian safety protocols [VR+16]. These protocols include different impact scenarios, such as head or leg impacts, targeting different areas of the car. As a result, each car is assigned its own performance assessment grid. As shown in Figure 1.3, this grid highlights that the centre of the bonnet is a safer area for head impacts, while in this particular example, there is no designated safe zone for leg impacts. In situations where pedestrian impact is unavoidable, the use of passive safety knowledge can prove invaluable in minimising injuries. For example, as shown in Figure 1.4, the second scenario is preferable as it ensures that the passenger's head strikes the centre of the bonnet.

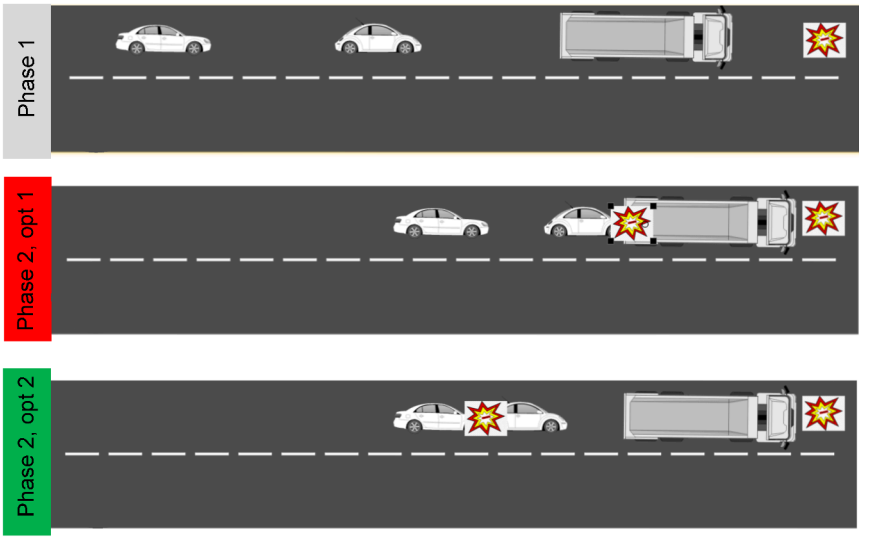


FIGURE 1.2: Multi-vehicle accident on a motorway due to an incident on the road. Presentation of two scenarios of braking leading to a crash.

1.4 Chapters relations

The following chapters begin with an overview of the application of KG in the automotive industry, Chapter 2, followed by related work on KG for CAE. Later, a description of the FE models used in this study is given in Chapter 4. We then focus on the main content of this research in the following four chapters. This thesis content is largely derived from a series of peer-reviewed publications to which I have been the main contributor. The chapters 5, 6, 7, 8 correspond to the papers, [PG22; PGS23b; PGS22; PGS23a] respectively.

[PG22] Anahita Pakiman and Jochen Garcke. "Graph modeling in computer assisted automotive development". In: *2022 IEEE International Conference on Knowledge Graph (ICKG)*. 2022, pp. 203–210. DOI: 10.1109/ICKG55886.2022.00033.

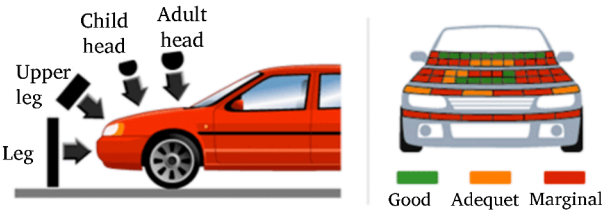


FIGURE 1.3: Pedestrian safety assessment in Euro NCAP [VR+16].

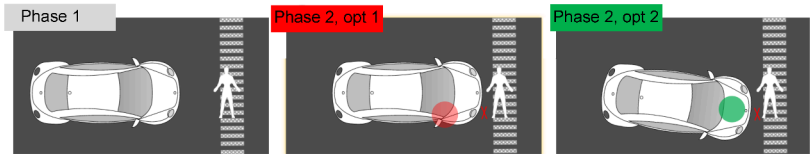


FIGURE 1.4: A crash of a vehicle to a pedestrian. Presentation of two scenarios for managing the situation.

[PGS23b] Anahita Pakiman, Jochen Garcke, and Axel Schumacher. "Knowledge discovery assistants for crash simulations with graph algorithms and energy absorption features". In: *Applied Intelligence* (2023), pp. 1–20.

[PGS22] Anahita Pakiman, Jochen Garcke, and Axel Schumacher. "SimRank-based prediction of crash simulation similarities". In: *Applied Intelligence* (2022). Under review, INS Preprint No. 2210, Institut für Numerische Simulation, Universität Bonn.

[PGS23a] Anahita Pakiman, Jochen Garcke, and Axel Schumacher. "Graph extraction for assisting crash simulation data analysis". In: *Graph-Based Representation and Reasoning*. Accepted. Springer Nature Switzerland, 2023, pp. 171–185. ISBN: 978-3-031-40960-8.

Additionally, Chapter 9 presents the ontology and its use cases, followed by the programming technology of the thesis in Chapter 10 and

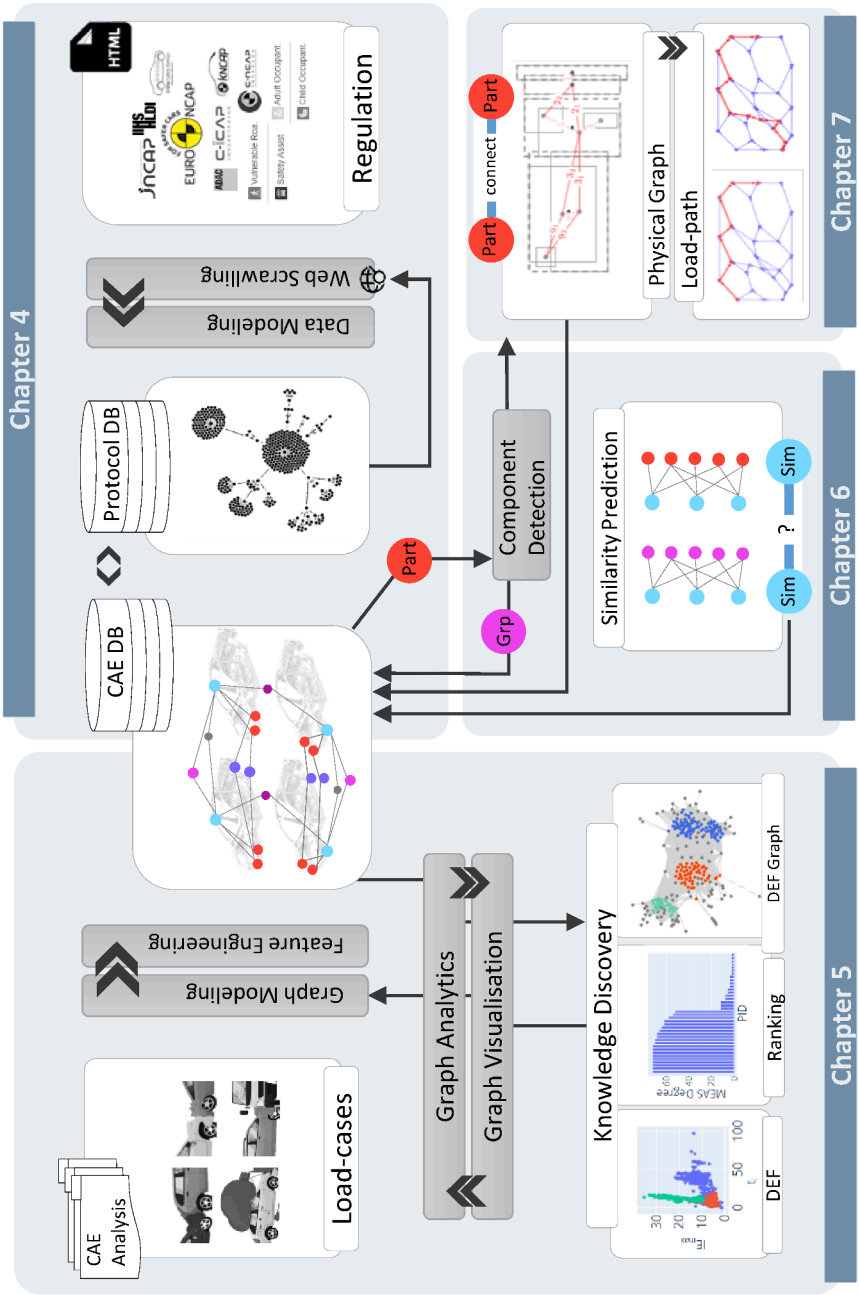


FIGURE 1.5: Reserach method overview in chapters 5, 6, 7, 8. DEF is referring to the Design Exploration Fingerprint presented in Chapter 6.

the conclusion and discussion in Chapter 11.

Figure 1.5 is a summary of the contents of each paper and their relationships. The graph modelling that incorporates the data from the CAE analysis and the regulations in Chapter 5 is first presented. Several CAE disciplines are supported by this data modelling. However, only one of the vehicle safety regulations, Euro NCAP, is included in this thesis.

Chapter 6 presents the feature engineering for the abstraction of the crash analysis problem to a graph in the context of the safety of motor vehicles. This part of the research, together with the feature engineering, presents a new data visualisation that, on one hand, proves the capability of the features and, on the other hand, supports further knowledge discovery from the data. Next, Chapter 7 investigates a graph analysis method that uses a network of simulations and their critical design parameters to find similar simulations. This thesis proves that this similarity prediction can be used to discover clusters of similar behaviour, thus supporting the discovery of different car designs with similar performance. Finally, in Chapter 8, the graph also includes the connectivity of the vehicle parts.

2 Application of knowledge graph to automotive

A fundamental step towards the practical use of Knowledge Graph (KG) is the transformation of the data flow within companies. The available KGs in engineering [AS21] are summarised in a survey of domain-specific KGs. No KG is currently available for the structural design of the vehicle for crash Computer-Aided Engineering (CAE) analysis. Design engineers and attribute managers rely on reports from CAE engineers in today's workflow. Design engineers or Computer-Aided Design (CAD) engineers are responsible for the detailed design of the vehicle. Typically, several parts of the vehicle are designed by each design engineer. A complete vehicle in CAD format is then converted into an Finite Element (FE)-model. Each CAE engineer sets up and runs a specific analysis. Finally, an attribute leader gathers all the available analyses from the CAE engineers, resolves any multi-objective performance issues of the design, and feeds the final decision back to the CAD engineer.

Each simulation output, including time series plots and images of structural deformations with stress-strain counters, is usually summarised in a PowerPoint presentation. These Powerpoints provide an initial assessment of the results and are automatically generated from the simulations. These documents lack connection to the simulations by converting the data into images. Only a small proportion of the data from many simulations is analysed and documented in detail by CAE engineers. This is usually done in the FE-model validation for a complete

new vehicle design, and the iterative designs will skip the detailed investigation by the CAE engineers. These reports, which contain more specific images of the structural behaviour during the crash and limited text, are known as "one-pagers". As a result, much of the simulation data is still missing from these documents, and the primary source of data remains in the simulations. Therefore, the priority here is to extract the data from each simulation to build the KG.

CAE data modelling is challenging because the data is complex and multiple disciplines with different requirements — CAE engineers, CAD engineers and attribute managers — interact with the CAE data. However, the flexibility of graph data modelling allows uncertainties to be taken into account and the modelling to evolve. This thesis presents a first attempt to define a semantic representation that stores information about the different crash scenarios, the vehicle design deviations during the development process, and the quantities of interest that measure the outcome. Therefore, semantic selections follow the development concepts, FE-modelling terminology, crashworthiness assessment quantities, and other relevant entities. The complexity of the data within each analysis compared to the total number of analyses makes the data modelling challenging. Consequently, not all the simulation data is loaded into the car-graph. The car-graph is currently a relatively small graph compared to other areas. Furthermore, these data points can be used as input for Machine Learning (ML) analysis, as graph modelling also allows the storage of ML results. The vision of this work is to support engineers, with a particular focus on automating the safety evaluation of different, uncalculated crash scenarios, by using data modelling and ML techniques to facilitate the assessment of cause-and-effect relationships within the development process.

2.1 From data to knowledge: the semantic web's journey with knowledge graphs

The word KG was used for the first time in 2012, but this method is part of the Semantic Web Technologies (SWT), which started much earlier. The following is a brief overview of this area for a better understanding of the basics of this technology. In the early 20th century, the introduction of the Semantic Web created a technology stack to support a 'web of data' rather than a 'web of documents'. The ultimate goal of the 'web of data' is to enable computers to perform more meaningful tasks and to develop systems that can support trusted interactions across the network. SWTs include various data exchange formats (Turtle, Resource Description Framework (RDF)/XML, N3), query languages (SPARQL, Cypher), ontologies and notations such as RDF schema (RDFs) and Web Ontology Language (OWL). They describe entities and relationships within a given domain of knowledge [PJ21].

Linked data is at the heart of the SWT because it enables large-scale data integration and reasoning. Ontologies are the backbone for structuring linked data and are essential for defining links within and between datasets. They allow users to browse a schematic model of all the data within applications. Ontologies make it possible to combine deep domain knowledge with raw data and to bridge datasets across domains. The KG is another essential component of the Semantic Web. There are many types of KG; some examples are DBpedia, Freebase, Wikidata, YAGO, and so on [PJ21]. However, most of the available KGs and ontologies have been built from textual data, whereas in this research, the FE simulations are the raw data.

2.2 Structuring knowledge graph

The proven power of the KG has attracted much attention in all sorts of fields. Here, a KG is defined as in [Hog+21], as a way of accumulating and communicating knowledge about the real world. To the best of

our knowledge, there is no KG for vehicle development and specifically vehicle safety [Ji+21; AS21]. There are several approaches to structuring a KG that these studies provide a good overview of the nature of the KG and a taxonomy for its construction, Figure 2.2 [Ji+21] and Figure 2.3 [AS21]. Among the various categories of KG, the focus here is on knowledge-aware applications and knowledge representation learning, Figure 2.2. This thesis includes the study of semantic analysis and search engines for knowledge-aware applications.

This thesis covers all levels of knowledge extraction for the FE model in Figure 2.3. However, not all of the properties have been loaded into the database and the process has been carried out selectively. The property graph, Neo4j[1], is the database that is used to have the most flexibility in the modelling of the data. Considering Figure 2.3 in KG construction, this thesis has a hybrid approach for the type of knowledge base. This approach supported iterative improvements to the schema throughout the project. The KG construction method is based on the domain knowledge rule.

KGs are typically created using a top-down approach, where an ontology is created and then populated with data to create the KG. However, an ontology-based approach is time-consuming to initialise and update. Automating the acquisition, processing, and use of knowledge has high value when dealing with large amounts of diverse domain data [BS22]. Moreover, it is essential to consider the questions one wants to address with the data. The question-answer-based approach guides this thesis with an evolving schema that has developed in a feedback loop. This data modelling allows answering some of CAE related questions and focuses on crash FE models.

Figure 2.1 summarises the workflow for loading data into the graph database and exploiting graph mining methods. When evaluating the workflow steps, the aim is to consider the reliability of the methods,

[1] https://neo4j.com/

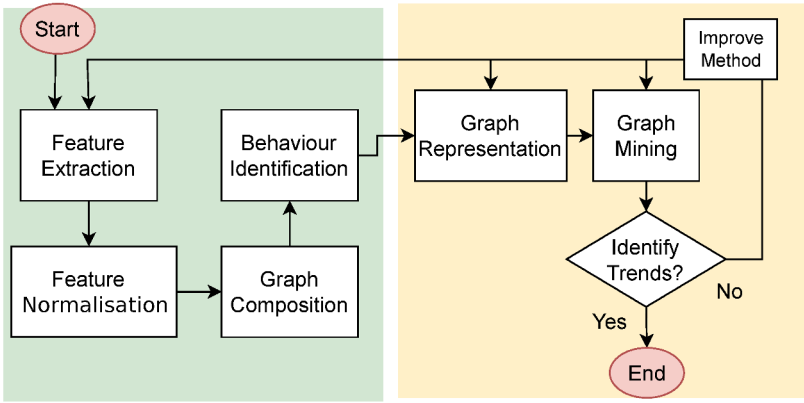


FIGURE 2.1: Feedback process during car-graph development with a focus on the identification of trends in the simulations. Green zone: Stages of the construction and population of the graph representation. Yellow zone: Usage of the graph representation and feedback loops to earlier stages.

particularly for a full-scale CAE crash FE-model, as well as their computational efficiency. A primary concern is that the workflow steps can detect slight and significant crash behaviour deviations in view of the different changes engineers perform during the development stages.

In order to gain insight into their use in the CAE-based development process, this thesis constructs a first graph schema for car crash simulations. Its purpose is to provide searchability and analysis of the data, which should provide the user with additional insights gained from the data. The current graph modelling mainly addresses two outputs: an intuitive graph analysis of CAE data that reflects semantics or behaviour and a graph representation of the data that enables ML methods. The implemented workflow has two stages: first, it processes the data and stores it in a graph database, and second, it extracts the computational graph from the database for visualisation and ML studies.

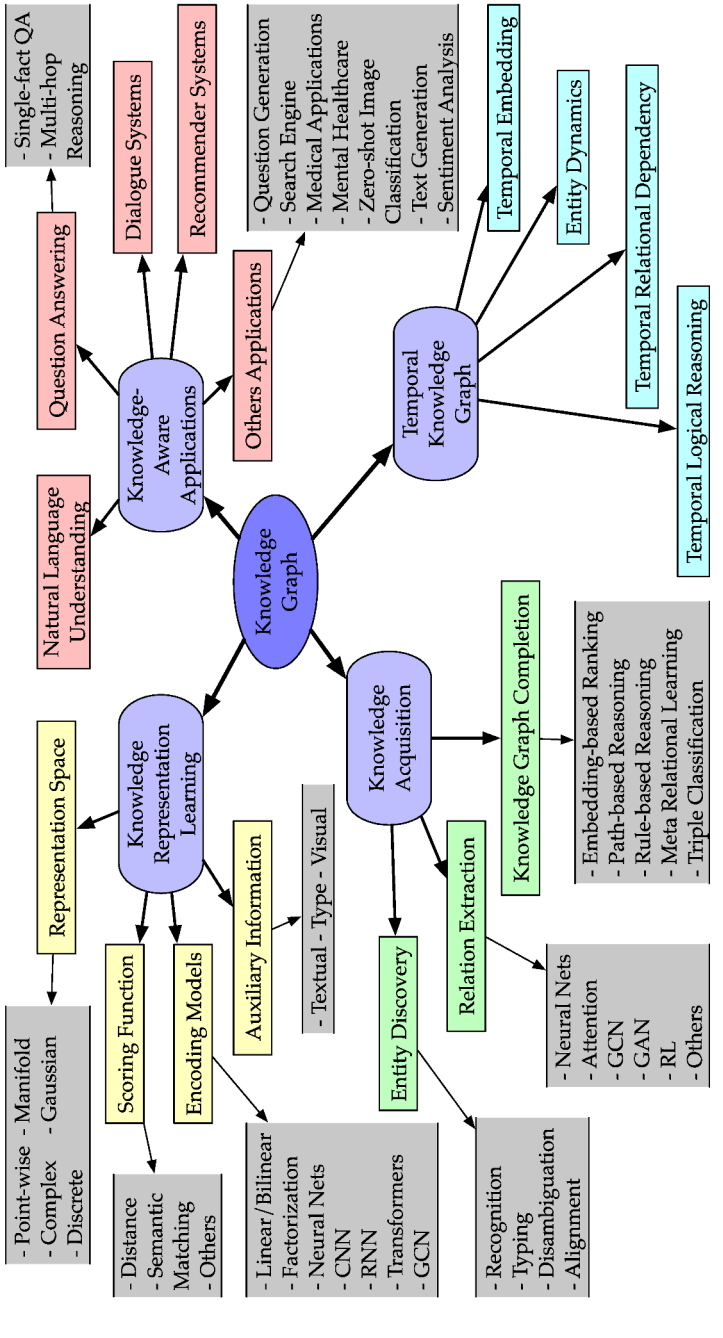


FIGURE 2.2: Overview of the categorisation of research on KGs to provide a map of where this work stands. The grey boxes are examples of methods in the field; more information is available in [Ji+21].

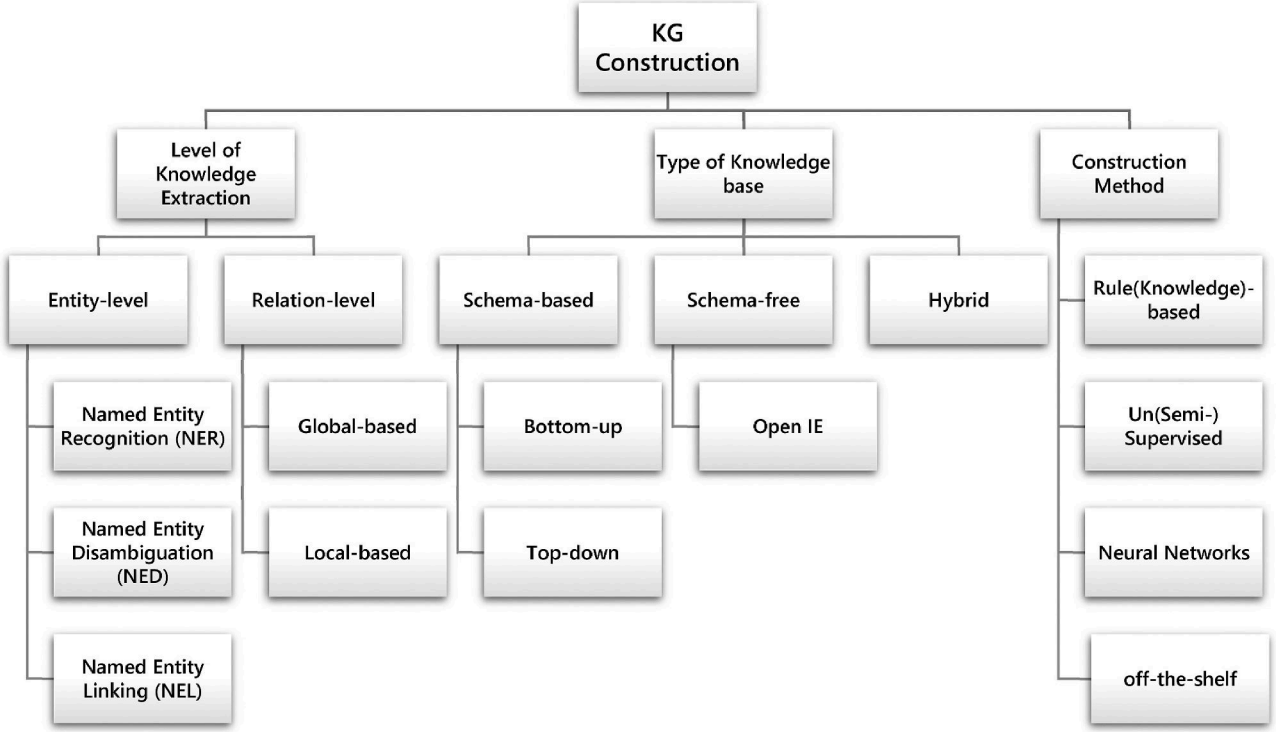


FIGURE 2.3: A taxonomy for constructing KG to give an overview of the possible ways to construct KG and how it relates to the method used in this thesis [AS21].

2.3 Knowledge graph for crash simulations

To minimise injury to occupants and pedestrians, the deformation structures of the vehicle must absorb the kinetic energy. Energy dissipation performance is the essential underlying physics of the crash analysis problem. Therefore, this work focuses on the internal energy absorption of the simulations under various available output options of the solver, Chapter 6. The parts in the CAE model are preferably related to each component of the vehicle. The FE solver outputs the energy per part over time, a so-called energy curve. The number of parts confronts CAE engineers with a practical evaluation of energy curves. Therefore, the use of energy curves in the workflow is limited to, for example, stability investigations (checking the total energy of the simulation) and identification of outlier entities (e.g. parts with negative energy).

This thesis shows that energy curves hold information to characterise the simulation crash behaviour. Data analysis on energy curves will simplify data processing to represent the crash behaviour based on a few features. Although other solver outputs are taken into account during a crash simulation analysis, the inclusion of other outputs is beyond the scope of this work. Our focus is on using the primary output of the solver rather than requiring any pre-processing. This method is capable of evaluating simulations that exist within companies from previous vehicle development processes without the need for additional re-evaluation.

Later, this input is used as the main features of the simulations for knowledge discovery, similarity assessment and identification of the essential parts and the load-path. The load-path defines the sequence of part engagement during the crash. The searchability of the simulations is the essential functionality for crash simulation KG, car-graph.

3 Related work for CAE

The proposed car-graph is inspired by the trends in information retrieval and mining, which have moved from a document-centric to an entity-centric approach. The term "Knowledge Graph" was introduced in 1972 as a graph of data intended to accumulate and convey knowledge about the real world [Sch73]. Since IBM Watson's victory in Jeopardy in 2011, Knowledge Graph (KG)s are attracting increasing research interest due to its ability to store knowledge, structured or unstructured, extracted from heterogeneous domains, and to query it to answer questions [Li+21]. In a variety of domains, the proven power of KG has attracted attention. A survey of domain-specific KGs summarises the available KGs in engineering [AS21]; the engineering domain most relevant to this research is manufacturing. Moreover, there is recent research done on the use of KG for Computer-Aided Engineering (CAE) input model [Tha20]. However, there is still no existing KG for the performance evaluation of vehicle design and, in particular, vehicle safety [Ji+21].

Current manufacturing KGs focus on production and manufacturing rather than product development. In addition, most KGs are text-based, and there is limited work on three-dimensional (3D) shapes as a product description [BS22]. Digital twin models for industrial manufacturing, Industry 4.0 and Computer-Aided Manufacturing (CAM) are examples of applications of KGs. In [Buc+21], an overview of available research in manufacturing is provided. Currently, there is no KG available for the CAE/Finite Element (FE) domain, nor specifically for crash simulation, which is the focus of this thesis.

Although there is no KG for CAE data, there are investigations of ontologies for Computer-Aided Design (CAD) and CAE integration [Bou+19], FE-simulation [Kes+19], and crash analysis [FSR19; FS20]. In the context of computing, an ontology is a concrete, formal representation of what terms mean within the context in which they are used (e.g. a given domain). As with other conventions, the usefulness of an ontology depends on the level of agreement about what it defines, how detailed it is, and how widely and consistently it is adopted. Adoption of an ontology by the parties involved in a KG can lead to consistent use of terms and modelling in that KG [Hog+21].

According to [Kes+19], several studies have already applied a knowledge-based ontology system to provide simulation knowledge to FE users. These studies do not consider the extraction of new relationships between data or the answering of an engineer's analytical questions. Some have focused on automating the generation of the FE-simulation [Kes+19] or retrieving simulation solutions from existing simulation [Kes+19]. Moreover, the case studies are simpler [Kes+19] than a full crash simulation. Recent work has characterised the CAE domain and identified unresolved challenges for custom data and metadata management as a graph [Zie+20]. In addition, some studies have explicitly examined a crash simulation ontology and investigated the reasoning structure of engineers, particularly in relation to report generation, as seen in [FS20; FSR19].

In summary, the previous work has had a knowledge management system orientation to understand the data structure and procedures in the company, while the simulation data itself has not been studied. The inclusion of related work on 3D data in the automotive development process, covering CAM, CAD and CAE, reveals the presence of distinct product data streams at different stages of vehicle development. There are more research studies on KGs in CAD and CAM compared to CAE. In addition, automated CAD-CAE model integration has generated ontology models and geometric feature extractions. These studies are of interest to link CAD knowledge to CAE. Kirkwood et al. used design

change vectors to enable sustainable integration of FE mesh and CAD models [KS18]. In a subsequent study, Feng et al. introduced an automated approach for generating simplified and idealised geometry models tailored for CAE simulation [FZL20].

Parametric CAD models, knowledge management, and Knowledge-Based Engineering (KBE) systems have been striving for decades to capture, digitise and automate the application of this type of knowledge in product and production development [Joh+18]. Using the product design KG, [BS22], the authors demonstrate the effectiveness of 3D shape retrieval using an approximate nearest neighbour search. They illustrate the use of KG for design reuse of co-occurring components, rule-based inference for assembly similarity, and collaborative filtering for a multimodal search of manufacturing process conditions. However, the KG should be expanded to include downstream data within product manufacturing and towards improved reasoning and methods to provide actionable suggestions for design bot assistants and manufacturing automation. In addition, in a design context, Huet et al. demonstrated that a cognitive assistant improved participants' ability to select applicable design rules more accurately, allowing them to devote more time to CAD modelling activities [Hue+21]. However, a test protocol is needed to confirm the preliminary results presented in this paper.

When considering the broad application of Machine Learning (ML) to crash data and KGs, it's important to note that its use in current CAE workflows is not as widespread as in more conventional ML domains. There are two primary applications of ML in crash analysis. First, it predicts crash behaviour to replace/support FE-simulation; see [BRK21]. Second, it uses dimensionality reduction on the vehicle component data during crash deformation to explore the FE-simulations, e.g. by identifying clusters [ITG19]. This usually requires an engineer to specify the critical components in advance. While considering all parts together is time-consuming and inefficient, it may also fail to highlight the bifurcation behaviour. This limitation emphasises the importance of automatic detection and filtering of the essential components.

What we have discussed so far is a broad consideration of the existence of KGs for CAE data and ML techniques for the domain. The construction of a KG for a new domain is a considerable amount of work, and there is no end to its development. This thesis focuses on searchability, extraction of crash analysis as a graph, and finally, establishing an ontology and relating it to the other ontologies. The following section provides a literature review for each of these topics.

3.1 Searchable simulations

Dimensionality reduction methods are one of the popular techniques in ML for comparison and gaining an overview of data [LV07]. In crashworthiness, these methods have been studied since 2008 [Ack+08] for exploration and cluster identification [MT08] of the FE-simulations. Note one can consider the distance of the embeddings as a similarity measure of the simulations. However, the focus of the available research has been on outlier detection or behaviour classification [ITG19; Kra+22].

As far as is known, there is no related work where the similarity of simulations is assessed using graph analysis methods. Consequently, other available methods are outlined for assessing the similarity of simulations. Studies using dimensionality reduction usually look into deformations of the FE-model [Gar+22; ITG19; Kra+22; MT08; Sch+22]. The challenge with these methods is their computational cost and sensitivity to capture local differences compared with global deformation. For example, these methods focus on realising an occurrence of a buckling, a global deformation, in comparison to characterising the buckling mode, e.g. its timing, a local feature. Furthermore, these methods' integration into the OEM's workflow has been limited [Sch+22], despite the long period these methods have been available.

Therefore, considering more scalable methods and other input measures is beneficial. One example for crash simulations is the FE solver outputs of energy absorption that gives Internal Energy (IE) per part over time,

a so-called energy curve. Studies show that energy absorption characteristics enable quantifying component performance for the design of experiments feedback in optimisation studies [DZ18; DZ19]. However, as far as is known, there has been no research into the use of features that are generic to the problem, such as energy features, to calculate the similarity of simulations. Another possibility is to use key performance indicators such as firewall intrusion or occupant injury criterion, but these reflect behaviour on a much coarser level, which limits their analytical capabilities. This thesis investigates the use of energy curves and assesses their ability to summarise the differences between simulations. Compared to the deformation-based approaches, the main benefit is computation efficiency.

Another aspect of similarity assessment is FE entities selection for comparison, e.g. node, element, or part. Due to the modelling techniques, most of the FE entities are smaller than a component of the vehicle. Here, a component refers to a group of parts whose structural functionality depends on its parts, e.g. two welded plates of a crash-box, where each plate alone has much less axial stiffness than the welded ones together. Consequently, comparing groups of entities as components can be seen as more physically meaningful. Note that the grouping information is partly[1] available in the CAD data during development. However, this data is lost in today's workflow when generating a FE-model from CAD information. Detecting these components automatically from a FE-model is challenging. In the study by Garcke et al., semantics are introduced for FE entities, enabling the identification of part splits during development [GPP17]. However, the grouping of FE entities from a structural perspective is not addressed. As a result, this thesis also introduces a method for automatically detecting the grouping of FE entities.

[1] If the split is not due to FE modelling.

3.2 Crash analysis as a graph

In crashworthiness, graphs have been used to predict the response of the vehicle [Gra11] or barrier [GG16] with so-called bond graphs. The bond graphs that are available for vehicle crashes represent the problem from the perspective of a mass-spring model [Gra11]. Bond graphs are ideal for visualising the essential properties of a system because their graphical nature separates the system structure from the equations [GB07]. Bond graphs represent the vehicle structure by summarising the physical elements and connections. However, as far as is known, there is no way of automatically extracting the vehicle structure as a bond graph. Incorporating the vehicle structure into the graph structure will enrich the data representation.

Before the growth of computing power allowed large Finite Element Method (FEM) analysis, there were other modelling techniques that simplified the problem to a mass-spring model. The advantage of the mass-spring model is that it can be easily represented as a weighted graph. Structural Impact Simulation And Model-Extraction (SISAME) is a general-purpose tool for the extraction and simulation of one-dimensional non-linear lumped parameter structural models [MRH92]. Using SISAME, mass element weights and spring element load-paths were optimally extracted directly from the test data accelerations and wall forces [Lim17]. However, the Lumped Mass Spring (LMS) modelling is one-dimensional and focuses mainly on accurately modelling the test data rather than representing the structural performance of the vehicle. Later the Deformation Space Model (DSM) was introduced [Lan+19] to compensate for the limitations of the LMS. It can only roughly capture displacements and energy absorption, neglecting connections and interactions with other components.

Another use of graphs in crash analysis is in the structural optimisation of the vehicle [OS13; SS17]. Here, the optimisation method adds vertices and edges to stiffen the structure, starting with a simple graph describing the perimeter of the vehicle. The focus of these studies is to search

with a graph for the optimal solution of the vehicle design. As a result, to complete the vehicle design and ensure safety performance, further processes and CAE analysis are required.

To summarise, automatically converting a crash FE-model in vehicle development to a graph is still an open research question. Depending on the detail required in a graph, there are several ways to represent an FE-model of a vehicle. As a specific application, this thesis investigates how the addition of connections to the graph allows a load-path analysis to be performed for each simulation.

3.3 Automotive ontologies

Ontologies entered the automotive domain with a focus on data integration [MSS+03]. This ontology also established CAE analysis in development processes. However, limited computing power kept ontologies at a concept level, modelling static properties of a product. Later, ontologies evolved with sensor-based data ontologies describing the dynamic behaviour of the vehicle [Klo+18b; Klo+18a]. These ontologies consider a vehicle as a network of sensors and assume that the sensor measurements are synchronised, which is often not the case. As a result, new ontologies are being developed to take into account the stream of observations and events [ACG21]. In summary, there is currently no generally defined strategy for building AI-oriented ontologies for the automotive sector. One of the critical challenges is the need for a standardised global vocabulary [Urb+21].

In terms of vehicle safety, future vehicle developments will require the assessment of relevant accident scenarios that are not covered by today's regulations or consumer crash tests. As future Highly Automated Vehicle (HAV)s are expected to offer new, alternative seating positions, i.e. increased passenger comfort, these positions need to be safety-approved and homologated. As a result, several projects are underway to define the technological developments needed to enable the automotive industry to design and develop new safety systems. A recent focus

has been on knowledge-based approaches that represent scenarios to promote scenario-based evaluation of Advanced Driver Assistance Systems (ADAS) and Autonomous Driving (AD).

For this purpose, ontologies have become a key component for the formalisation of this knowledge [Men+18]. An event-based scenario description for testing has been presented. The OpenPASS [Ope22] software platform is an outcome of these investigations, which enables the simulation of traffic scenarios to predict the real-world effectiveness of advanced driver assistance systems or automated driving functions.

The work by Katsumi et al. presented the potential applications of ontologies in transport research and operations, along with an overview of existing transport ontologies [KF18]. The importance of domain knowledge structures for different purposes has been identified in various works in the transport domain. For example, real-time assessment of traffic scenes [BMM18], automatic support for the design and analysis of performance monitoring of public transport systems [Ben+17], and testing and labelling of applications [Urb+21]. However, based on [Urb+21], there is no standardised approach or formal requirement for defining an ontology other than the W3C group ontology languages. Furthermore, most vehicle design CAE analyses consider passive safety, while transport ontology models focus on autonomous driving and active safety. In addition to the benefits of the vehicle development process, there is the potential to support active safety with passive safety functionality that leads to a safe accident.

There is no ontology for vehicle CAE analysis according to the best available data. This thesis first presents a graph modelling approach that focuses on introducing the semantics essential for exploring CAE data. Before defining the ontology, further development of the graph modelling is considered to assess its ability to answer the questions. These iterations are essential to achieve a useful ontology, especially for an unexplored area of research such as car-graph. The final step is to develop an ontology for graph data modelling.

4 Finite element modelling details

For the Finite Element (FE) simulation data sets, two aspects are considered when constructing the Knowledge Graph (KG) for crash simulations:

- Consideration of recent FE-simulations for scalability of method and feature engineering
- Easy labelling of the data to assess the performance of the Machine Learning (ML) method.

With these two objectives, this thesis involves two main sources of FE simulation data. First, a full-scale model is used to develop feature extraction for ML applications, taking into account computational efficiency, on full detailed models in Section 4.3. Later, an illustrative example is studied to simplify the coupling of the structural components involved in Section 4.1. This makes the analyses easier to understand. In addition, the illustrative example facilitates the evaluation of the method by comparing the predicted result with the engineering-based designation. This illustrative example is a submodel based on the Yaris FE model from the CCSA [Cen15]. This chapter discusses the FE modelling aspect of the sub-model used as an illustrative example in chapters 7 and 8. Finally, Section 4.3 provides an overview of the full-scale model. This data is used for the scalability of the method and feature engineering used in Chapter 6.

4.1 Illustrative example

Using a sub-model based on the Yaris FE-model from CCSA [Cen15], simulation data is generated that allows easy labelling of crash behaviour. Figure 4.1 shows the selected components of the submodel, where the main components are the front bumper beam, crash-boxes, and side-members. Further details of the changes and the data set used for the simulation are given below.

4.1.1 Included parts

Due to the complexity of classifying the crash behaviour, the sub-model is selected along only one deformation axis, the longitudinal axis. This limits the crash modes to the yaw angle. This sub-model includes the central load-path from the bumper to the side-member. The exterior parts are omitted because they do not structurally absorb the impact. The vehicle structure absorbs the impact before it reaches the firewall. A firewall is a part of the body that separates the engine from the passenger compartment. As a result, the structural parts after the side-member are eliminated. Figure 4.1 illustrates the chosen components of the sub-model, with the primary elements being the front bumper beam, crash-boxes, and side-members.

Originally, the Right Hand Side (RHS) and Left Hand Side (LHS) were asymmetric with respect to the xz plane. This asymmetric design is caused by packaging the vehicle components, which usually limits the symmetrical design of the vehicle structure. However, using a symmetric model as a starting point helps in designing the crash modes with more control over the changes and the expected performance. Therefore, the FE model is updated to be symmetric to make it more reliable in labelling crash behaviour. Figure 4.2 is a summary of these changes. The changes include removing the tow hook sleeve from the RHS, Figure 4.3a, and making the bumper beam, crash-box and side-member reinforcement symmetrical. The bumper beam and crash-box are mirrored from the LHS to remove the tow hook sleeve, and the selected

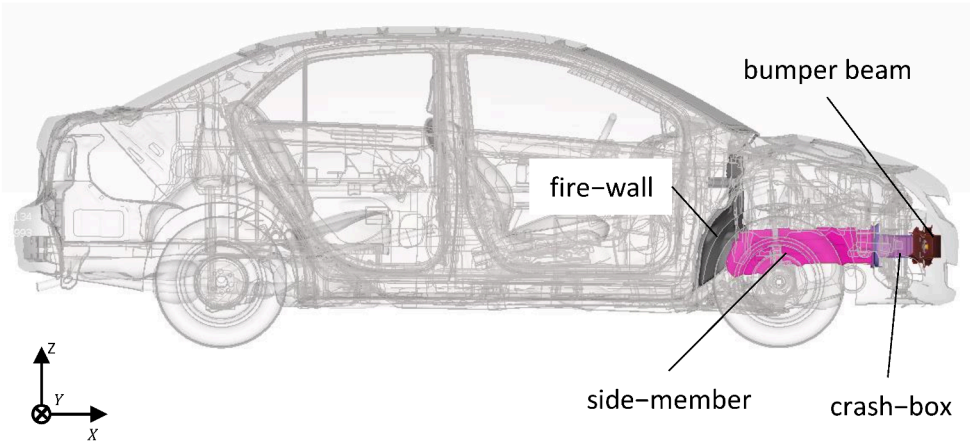


FIGURE 4.1: Included parts in the illustrative example.

side-member is from the RHS due to the higher number of reinforcement plates.

For the defined FE-model, the impact velocity is 56.3 km/h in a frontal impact against a rigid wall, Figure 4.3b. However, the rigid wall is repositioned to be closer to the structure, 80 mm, to reduce the free-flying of the simulation. This gap is generated by removing the front facia. Finally, to achieve deformation in the structure, the end of the side-members are constrained in longitudinal axis displacement and moments, figures 4.3c and 4.4. This boundary condition contributes to the inertia from the total mass of the vehicle. However, this way of modelling is more conservative compared to allowing small deformation by mapping the full model displacement response on these nodes; still, for the purpose of this study to compare the crash modes, it is unnecessary.

4.1.2 Increased deformation

The internal energy flow in the vehicle is characterised by the impact Kinetic Energy (KE). When two objects are considered in an impact, KE

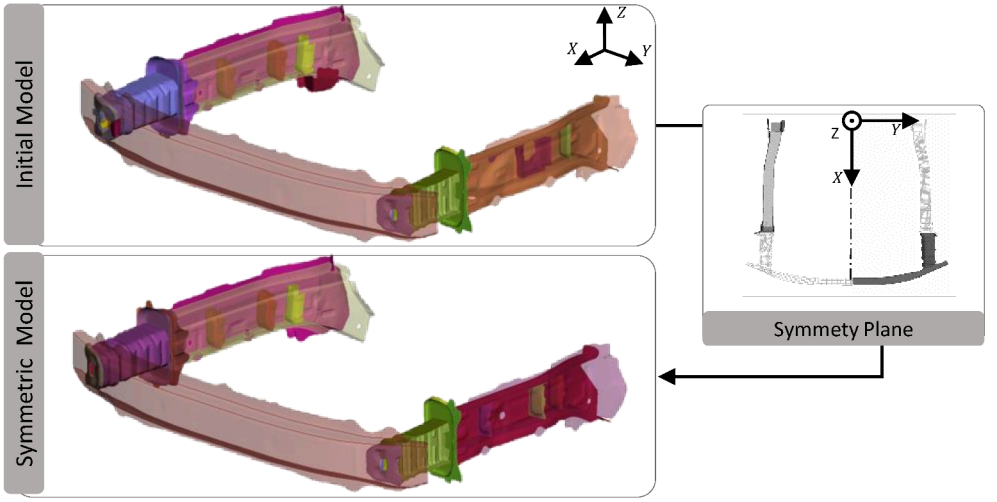


FIGURE 4.2: Changes applied to make the model symmetric: removed the RHS tow hook sleeve, used the LHS crash-box and used three reinforcement plates from the RHS side-member.

has two variables: the impactors' relative velocity and mass. In the full frontal simulation, the vehicle mass and velocity are the only parameters, as the other impactor is the rigid wall with no defined mass and velocity. Consequently, by defining a sub-model that is only part of the vehicle, the KE drops that affect the amount of deformation in the structure are eliminated.

Therefore, it is necessary to add mass to compensate for the absence of the rest of the vehicle mass in the analysis. The best way to add mass is to have a point mass for each deleted component in its Center of Gravity (COG) and keep the corresponding connectivities. This approach requires a considerable modelling effort, which is more necessary for correlating the FE-model with physical crash test data, which is not the focus of this study. This effort is not necessary for these analyses, as the focus is on discovering relationships between simulations. Another option is to add a single-point mass behind the structure, but this is

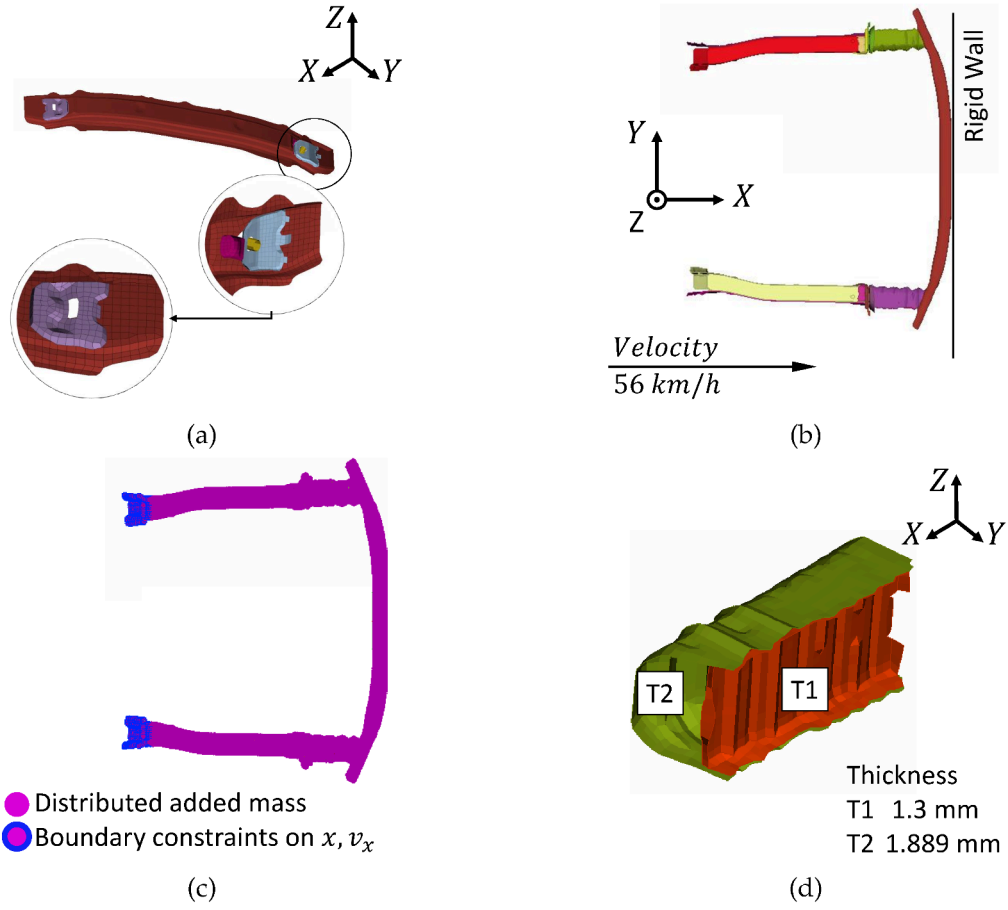


FIGURE 4.3: FE-sub-model setup, (b) top view of the included components with applying symmetry on crash-box, side-member and bumper beam, (c) add boundary conditions and mass, (a) removing tow hook sleeve from the bumper beam. (d) crash-box thickness parameters.

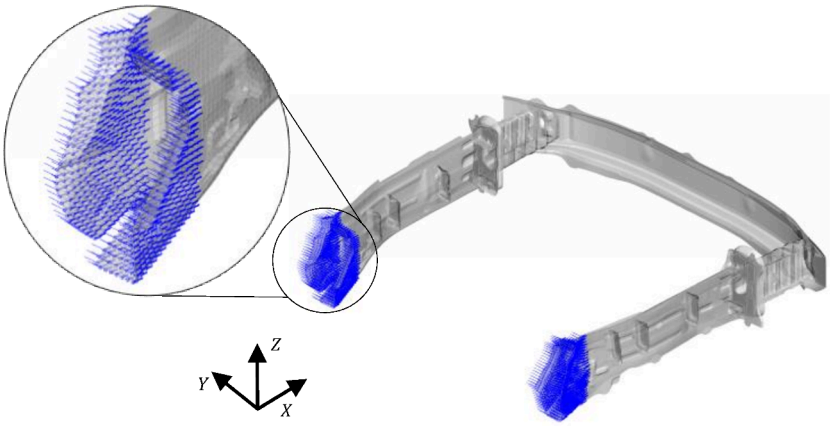
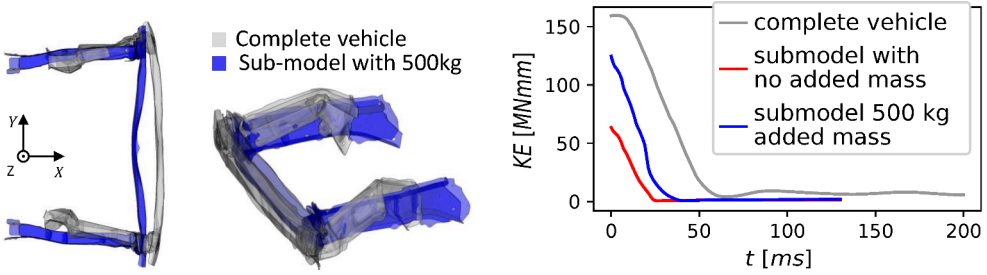


FIGURE 4.4: Selected nodes at the end of sid-member to apply the boundary condition shown in Figure 4.3c.

problematic due to the defined boundary conditions and the combination of nodal-rigid constraints that are needed to connect the mass. The presence of a mass behind the structure will constrain the longitudinal motion in some way, but due to the lack of lateral structure, the side-member will then move inward, which is not desired. Consequently, the added mass is distributed on the structure, Figure 4.3c.

Finally, in order to obtain a slight deformation also in the longitudinal members, a total of 500 kg is added to increase the KE. The mass is distributed over all nodes of the submodel, Figure 4.3c. The amount of this mass is determined by gradually increasing it until the desired deformation is obtained while the model is still stable. The aim of this investigation is to look for different deformations that result in complete crushing of the crash-boxes and slight deformation of the side-members. Figure 4.5a shows the overlay of the final deformation for the sub-model parts for the full model and the sub-model with 500 kg added mass. In the full model, the side-members have plastic deformation, but for this analysis, the combination of full crush and minor plastic deformation is of interest, crash-boxes and side-members, respectively. Figure 4.5b



(a) Deformation comparison.

(b) Kinetic energy comparison.

FIGURE 4.5: Comparison of deformation a and Kinetic energy (KE) b for the full model and sub-model with 500 kg added mass.

compares the kinetic energy for crash-boxes for the full model, the constrained sub-model, and the sub-model with added mass.

4.2 Deformation modes and simulation data sets

There is no standard way to characterise crash behaviour. The deformations are classified on the basis of the stiffness distribution of the vehicle structure and the axis of rotation of the deformed structure. The longitudinal stiffness deviation does not cause a corresponding rotational mode in a frontal crash due to its direction. However, the structural stiffness of the vehicle along its vertical and lateral axis causes deformations with pitch and yaw, respectively, Figure 4.6. For example, for YARIS, the analysis along the vertical axis examines the stiffness distribution of the vehicle structure along three main load-paths in the vehicle's vertical axis, which are the lower, middle and upper load-paths, Figure 4.6. The different stiffness distribution affects the pitch angle of the deformed structure. In addition, along the lateral axis for a symmetrical load-case, the yaw angle of the structure is of interest, e.g. caused by the unsymmetrical stiffness of crash-boxes.

This study consists of 66 simulations of a full-frontal impact against a

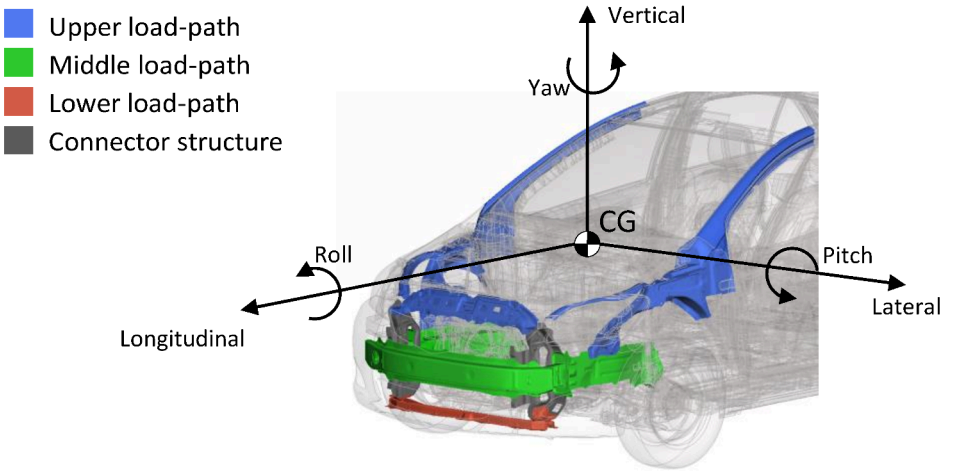


FIGURE 4.6: Yaris load-paths and the connecter structure of the three load-paths.

rigid wall with a speed of 56.3 km/h for the middle-load-path structure presented in Section 4.1.1. The simulation setups for the submodel have minor design changes that are replicating real development process changes. Consequently, the differences in the outcome of the simulation are in the scale of Computer-Aided Engineering (CAE) development processes. The simulations vary in crash-box plate thicknesses, where the crash-box is built of two sheet metal thicknesses, Figure 4.3d. For all simulations, the crash-box outer plate thickness, T_1, is dependent on the inner plate thickness, T_2, by

$$T_2 - T_1 = 0.6. \tag{4.1}$$

Here, T_2 increases from a minimum value of 1.6 $[mm]$ in equidistant steps of 0.1 $[mm]$ to maximum of 2.6 $[mm]$. These variations are implemented equally on RHS and LHS. In a symmetric setup, T_2 of RHS and LHS increase equally in thickness, while for an asymmetric setup, RHS and LHS thicknesses increase unsymmetrically. Figure 4.7a shows the

employed distribution of the T_2 thickness value for LHS and RHS of crash-boxes for 66 simulations[1].

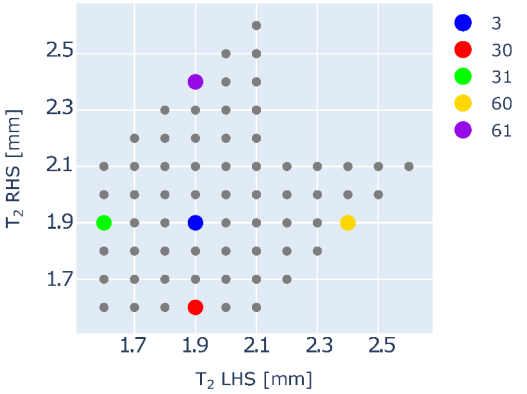
These changes cause asymmetrical and symmetrical absorption, which results in three crash modes for the deformation, Figure 4.7b. The crash mode indicates the yaw angle of the bumper beam. For this symmetric load-case, a symmetric structural stiffness results in a yaw angle of zero. In Figure 4.7a, the simulations with equal thicknesses on LHS and RHS are on the diagonal. For simulations below the diagonal, the LHS is stiffer, causing the crash mode to have a negative yaw $-v_z$. For those above, the stiffer RHS causes a positive yaw $+v_z$ for the crash mode.

Out of these 66 simulations, five are chosen as reference simulations. Figure 4.7c summarises the crash modes for the selected reference simulations, including one zero mode simulation and two simulations for each negative and positive mode. Simulation three is the base model with zero crash mode. The negative and positive modes have mirrored thicknesses, i.e. 30 with 31 and 60 with 61. They also have different stiffness ratios, $T_{2_{RHS}}/T_{2_{LHS}}$, i.e. 60-61 is stiffer than 30-31. The reference simulations will later be used to find the most similar ones; see Chapter 7.

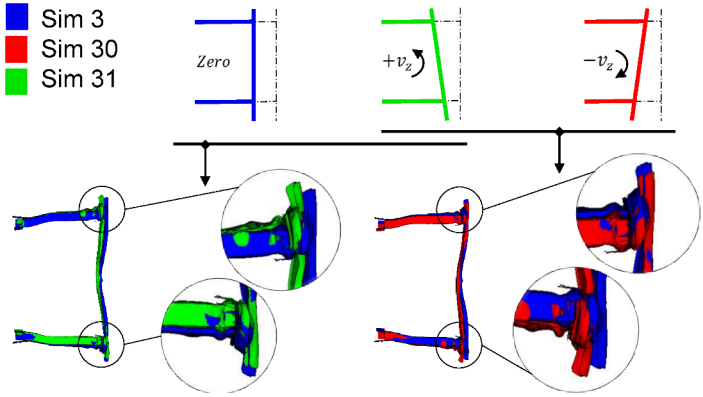
4.3 OEM data from CAE development stages

Besides the illustrative example, this thesis also explores industrial data for method scalability and feature engineering. In this way, The proposed data representation is evaluated and the resulting data exploration approaches are applied to industrial data derived from a vehicle development project carried out at China Euro Vehicle Technology AB (CEVT). In particular, four development stages and three load-cases for frontal impact analysis are under investigation. The development stages are so-called primary, early, middle, and late development stages,

[1]The simulations and databases are available at:
github.com/Fraunhofer-SCAI/GAE-vehicle-safety



(a) Spread of thickness T_2 of crash-boxes RHS vs.LHS for reference simulations and data set.



(b) Defined label for crash mode based on Yaw axis rotation.

Crash Modes					
Ref. Simulations	3	30	31	60	61

(c) Crash modes for the reference simulations.

FIGURE 4.7: Simulation setup consisting of 66 simulations. Coloured points in (a) are the reference simulations. FE-simulation result follows the colour coding of the crash modes (b). Five simulations are chosen as reference simulations (c).

where the names reflect the sequence of the stages. The considered development window covers roughly one-third to two-thirds of the complete R&D development phase (before the first real crash test). Table 4.1 summarises the three load-cases and the number of included simulations.

Specifically, it aims to assess the scalability and feasibility of the introduced energy features and Graph Machine Learning (GML) methods. The focus is on data visualisation to summarise the behaviours and trends. Note that data confidentiality hinders illustrating the developed vehicle platform or giving details about the FE-model. However, the crash behaviour is discussed using the component name. So, the general knowledge of crashworthiness helps to interpret and evaluate the results.

TABLE 4.1: Properties of the investigated CEVT data. The total number of all simulations from four development stages in each load-case. The deviation of the KE is due to the changes in vehicle mass during development.

	Name	No. Sim	KE_i* range $[MNmm]$	Velocity $[km/h]$
ffo**	full front overload.	215	.3 - .4	64.0
foU	front oblique overlap, new US-NCAP	121	.3 - .4	90.0
foI***	front small overlap, IIHS	275	.8 - .9	66.9

* Initial kinetic energy

** Internal load-case, with an additional 5 $[mph]$, 45 $[mph]$ overspeed compared to US-NCAP full front impact speed.

*** Over loaded speed, requirement is 40 $[mph]$

Generally, the positions relative to the direction of the barrier are essential in the analysis of crash behaviour with regard to geometry. Therefore, one can divide the components into the early, middle, and late

energy absorbent components, i.e. bumper beam, crash-box, and side-member, respectively. Additionally, the vehicle's vertical axis positioning includes middle and lower load paths in the absorption, i.e. crash-box and lower load path component, respectively.

The crash-box and the side-member are thin-walled structures with optimised cross-section shapes and crumple points, e.g. ditches and crash beads. They may collapse in a particular pattern to absorb energy efficiently. A side-member is longer and stiffer in comparison to crash-boxes. Further, the deformation modes of longitudinal beams include folding, tearing, and bending. Here, reinforcing components strengthen the beams and optimise the absorption of energy. However, the lower load path component is a thin-walled structure positioned vertically lower than the crash-box. It distributes the load in the sub-frame. Finally, the sub-frame is a structural component with a discrete structure that supports the axle, suspension, and powertrain. This component has minor absorption crashworthiness design aspects among the mentioned components due to the required durable performance.

So far, the components of the ffo load-case were introduced that are studied in-depth in Section 6.3.3. Additional essential components for the foU and foI load-cases are the A-pillar, cowl, front fascia, wheel arch, and wheel rim. A-pillar is the most forward vertical support of the vehicle (among A, B, C, and D pillars). The cowl separates the front compartment from the passenger cabin between two A-pillars. The rest of the components are none structural. For further background on crashworthiness, see e.g. [DB+04].

5 Graph modelling in computer assisted automotive development

5.1 Graph modelling for CAE knowledge graph

An essential step in Knowledge Graph (KG) is the modelling of graph data and the architecture of the graph database. Graph databases allow maintainers to postpone the definition of a schema, allowing the data – and its scope – to evolve more flexibly than typically possible in a relational setting, particularly for capturing incomplete knowledge [Hog+21]. This data structure flexibility is an ideal fit for the automotive industry, which is a fast-evolving industry due to the short development phase of the product caused by the high market demands. At the foundation of any KGs is the principle of first applying a graph abstraction to data, resulting in an initial data graph [Hog+21]. Here, the proposed graph data modelling for the crash analysis domain includes the Finite Element (FE)-model, FE-simulations, and requirement protocols[1].

Computer-Aided Engineering (CAE) data modelling is challenging since the data is complex, and several disciplines with different requirements — CAE engineers, Computer-Aided Design (CAD) engineers, and attribute leaders — interact with the CAE data. However, the flexibility of graph data modelling reflects existing uncertainties and allows

[1]The databases are at github.com/Fraunhofer-SCAI/GAE-vehicle-safety.

the modelling to evolve. This chapter presents a first attempt to define a semantic representation that stores information about the different crash scenarios, the vehicle design deviations during the development process, and the quantities of interest that measure the outcome. Consequently, the proposed semantic selections follow the development concepts, FE-modelling terminology, crashworthiness assessment quantities and other relevant entities. Data loading is selective due to the complexity of the data within each analysis in relation to the total number of analyses available. As a result, the graph here is relatively small compared to other areas. Additionally, these can be used as input for Machine Learning (ML) analysis, where graph modelling allows for storing ML results. The vision is to use data modelling and ML to auto-assess the cause-and-effect in the development process to assist engineers and, in particular, to assess the safety of different, unforseen crash scenarios.

After summarising the data modelling specification in Section 5.2, Section 5.3 describes Euro New Car Assessment Program (NCAP), an example of crash test requirements [VR+16], where a large amount of CAE data is generated during the development process to meet these requirements. These protocol examples help to find semantics to shape the evaluation structure with two objectives: firstly, to benchmark the vehicles and secondly, to help engineers independently understand the CAE reports by linking the performances to the requirements outlined in Section 5.4. Finally, Section 5.5 presents the application of this data modelling, enabling semantically connected dynamic reporting with CAEWebVis and graph analytics. Note that this chapter presents the first applications of graph modelling for the domain, and later chapters 6 and 7 will provide examples of use cases for this graph modelling.

5.2 Graph modelling specification

A good data model should enable developers to query and extract knowledge from a database without deep expertise in the CAE domain.

This section defines a set of competency questions that will later be used as a means to evaluate graph modelling in Chapter 9. These questions are divided into four different aspects to analyse. First, the design of experience discovery. In vehicle development, analyses are developed with a step-by-step approach to find the effect of each change independently and to find the best solutions. These changes are linked together in a development tree. Changes are usually applied to a specific part.

- What is the most common design change in a development tree?
- What are the top three strategic parts for a particular load-case?
- What are the top essential analyses of the development?
- What are the essential vehicle parts for all load-cases?
- What is the Design Exploration Fingerprint (DEF), Chapter 6, of vehicles with the same platform, Table 5.1?

Some examples of questions that can be used to further analyse existing data and find further links between analyses are,

- What are the input changes in the FE-model from analysis A to B?
- What is the essential time step for a particular load-case?
- What is the distribution of specific sensor data (node, element, part output)?
- What is the cross-domain strategic part?
- What is the closest simulation result to a given analysis?

The third category looks at the safety performance of the vehicles available on the market, the Euro NCAP regulations [VR+16] and the defined safety regulations,

- What are the benchmark vehicles[2] for the vehicle under development?

[2]Vehicles on the market that are similar in design to a vehicle under development.

- What are the safety protocols for the vehicle on the market in a given year?

Finally, there are load-case-specific questions that are more domain-specific in nature. Here, the analysis of pedestrians will be considered as an example of a use case. Pedestrian analyses are a complex set-up from a data point of view, as for each vehicle design, several analyses have to be performed for different positioning of the impactor. Consequently, for each vehicle design, several simulations are analysed. However, not all analyses need to be re-evaluated, as most of the vehicle parts that affect pedestrian protection are local, e.g. small changes in the design of the brackets. As a result, previous analyses could be used to assist in the prediction of the final performance.

- What is a pedestrian load-case final performance prediction for an incomplete analysis set?
- What is the statistic performance for a specific impact position?
- What is the best probable performance?

The underlying aim of the data modelling presented here is to answer these questions. In the following, details of the graph modelling will be presented. Chapter 9 will discuss some specific use cases of these questions.

5.3 Data model for Euro NCAP website

There are several crash safety regulations for car-makers to follow, which help to assess the safety of vehicles in the market. These regulations cover a series of vehicle tests to represent essential real-life accident scenarios that could result in injured or killed car occupants or other road users. The NCAP is an organisation that designs and conducts such tests [VR+16]. Euro NCAP information is used as the basis for some of the graph modelling.[3]

[3]The data is available at www.euroncap.com/.

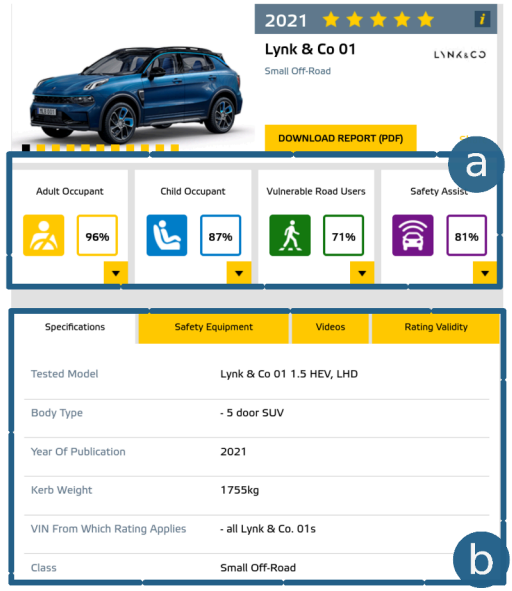


FIGURE 5.1: An example of Euro NCAP result page structure.

Euro NCAP has two primary sources of information that are of interest: test protocols and test results. Test protocols describe the specifications and the assessment methods for each crash test; these are available as PDF documents. Whereas a test result page contains the safety performances of a released vehicle in the market as HTML pages, images, and PDF documents, see Figure 5.1 for an example. Design engineers commonly compare information about the safety performance of similar vehicle classes during a new vehicle development concept phase. However, such an assessment requires a lot of manual data processing since more complex queries of the existing data are impossible, e.g. summarising the pedestrian upper leg performance of vehicles with a specific ride height. The ultimate vision for the Euro NCAP data modelling is to add semantics to them and develop an assistance benchmarking tool for the engineers, where the raw data can be extracted from the available test protocols and test results.

Firstly, the semantics for comparing the safety performance of vehicles are of concern. Such a comparison of vehicle performance is primarily conducted with vehicles from the same class, e.g. large family cars. A (Veh) node represents a vehicle on the market or under development. Thus, the class specification of the vehicle is stored as a (Class) node to ease querying vehicles from the same class. As can be seen, the available information per vehicle includes the protocol classifications for different Euro NCAP assessments, Figure 5.1 (a), and the vehicle configuration, Figure 5.1 (b). The vehicle specification table and the physical crash test media, images and video URL are stored as the properties of the (Veh) node.

In addition, the four test sub-disciplines are modelled, namely **V**ulnerable **R**oad **U**ser / Pedestrian, **A**dult **O**ccupant **P**rotection, **C**hild **O**ccupant **P**rotection and **S**afety **A**ssist with (VRU), (AOP), (COP) and (SA) nodes respectively, Figure 5.2. Additionally, for each pair of sub-disciplines and vehicle, the performance value of a vehicle is stored as the weight of a ◯– RATING –◯ edge. This information is extracted from each tab of the result table, Figure 5.1 (a). The specifications for the sub-disciplines may vary for vehicles depending on the year the vehicle was launched. Therefore, a (Year) node is added to each (Veh). This node identifies the relevant protocols.

In general, the aim is to have a continuous extraction and import of data from the web pages of the Euro NCAP. To achieve this, data is stored for each web page, where (Page) refers to each web page as a node in the database, and the page link, ◯– LINKED_TO –◯, reveals the structure and content of the web page. The URLs of protocols available on the Euro NCAP pages are structured based on the year and the sub-discipline. This structure is used to generate an additional node (Prtcl) for each protocol URL and to link the protocol PDF to its year and sub-discipline with ◯– DEFINED_AS –◯.

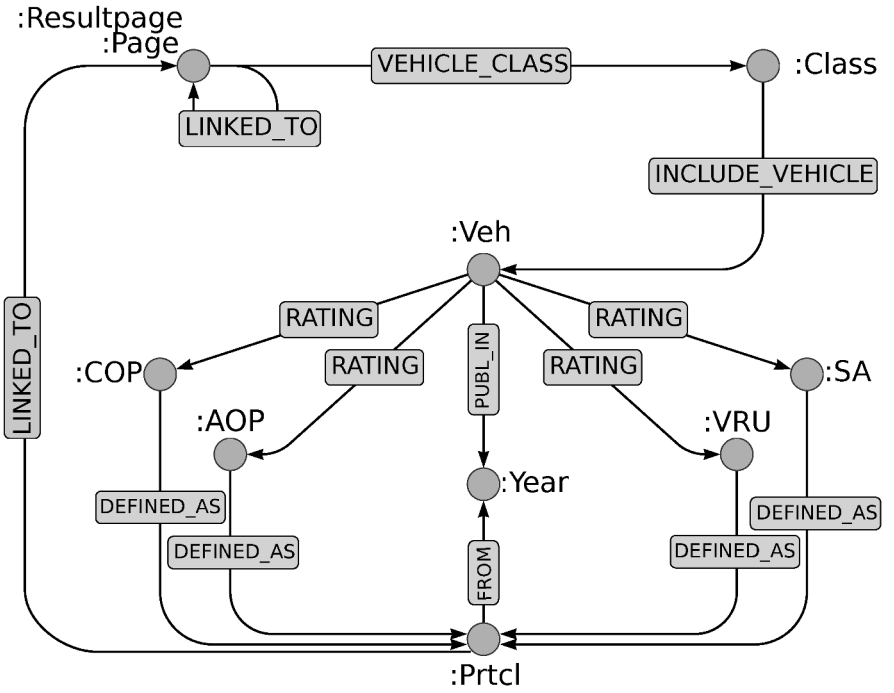


FIGURE 5.2: Euro NCAP graph schema.

5.4 Data model for CAE data

The graph nodes represent the hierarchy of a vehicle development process down to the FE representations for graph modelling of CAE data. The graph modelling consists of several segments, which are split based on the data sources or data processing, Figure 5.3 (a) to (g). It starts with a high-level representation of the development structure relevant to CAE data, as well as the vehicle safety regulations, Figure 5.3 (a) and (b), respectively. The combinations of both describe the setup and input for the CAE analysis. The other segments, 5.3 (c) to (g), are more specific to CAE data. The segments (f) and (c) are mostly adapted from [Tha20] with minor updates for consistency in the names.

For the CAE segments, the two main stages differentiate between the

data from the input and output of the CAE analysis, Figure 5.3 (c) and (d), respectively. The evolving analysis over time links the CAE data to the evolution of the vehicle structure, to which a separate segment is devoted, Figure 5.3 (e), bridging (a) and (c). Finally, the last two segments contain the additional semantics on top of the raw data for input and output, Figure 5.3 (f) and (g), respectively. These two build the basis for and represent the further analysis outcome of the car-graph, e.g. for assisting in processing the results by comparing simulations analysis aiming to connect cause-and-effect or methods for the recommendation of solutions. In the following, further details are given for each segment.

The final output of the company as a vehicle is the starting point and goes down to the smallest entity of the FE analysis, nodes and elements. A vehicle is typically divided into the so-called platform and upper-body. One broad definition of a platform is a relatively large set of product components that are physically connected as a stable subassembly and are common for different final vehicles [ML97]. Using a platform approach, a company can develop a set of differentiated products [WC92]. For simultaneous developments, what deviates between the vehicles, e.g. sedan versus minivan, is the upper-body. This concept is essential to capture in the data modelling as it enables the comparison of related vehicles, i.e. different upper bodies using the same platform. As a result, a vehicle (Veh) node has an upper-body (Ubdy) and a platform (Pltf), Figure 5.3 (a).

As the next segment, let us consider the safety requirements. Part of the vehicle development process is fulfilling crash test protocols, where a third party usually defines the requirements for crash safety, e.g. Euro NCAP. Note that a crash analysis always includes a vehicle and a type of test, specified by a barrier or an impactor. During the car development process, a set of barriers or impactors are studied based on predefined crash test scenarios. Consequently, connecting each vehicle to crash test protocols and the barrier or impactor provides an overview of the type of analysis included in the CAE-data, where the yellow nodes on Figure 5.3 connect to the Euro NCAP schema from Figure 5.2, see

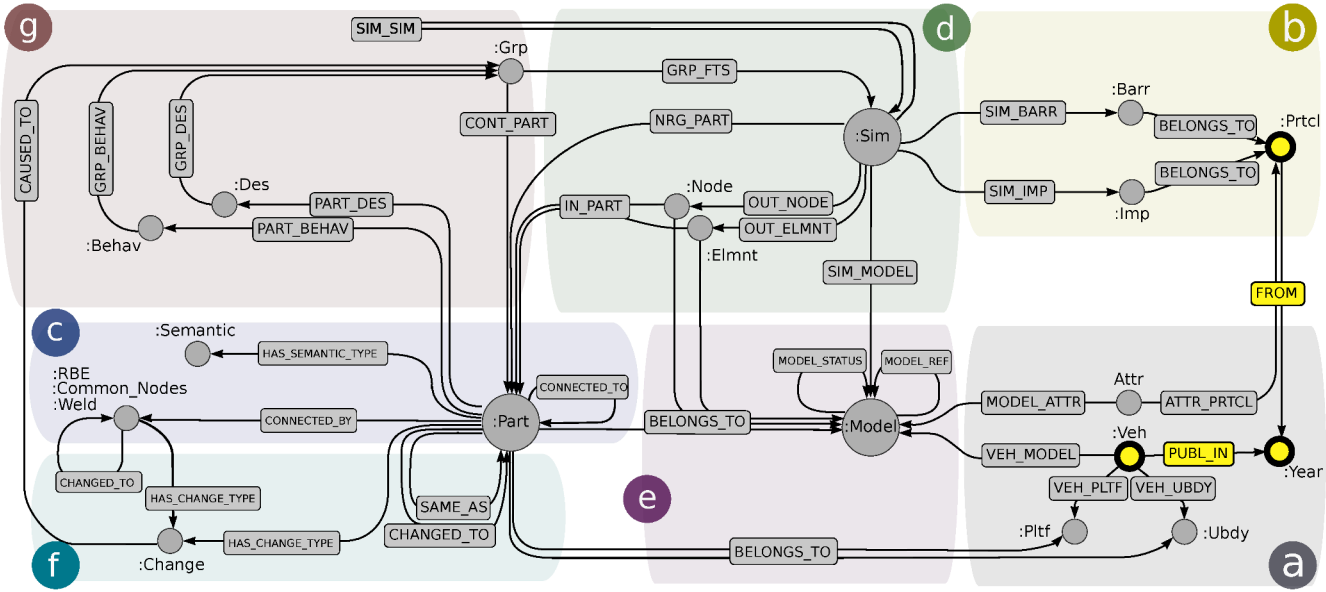


FIGURE 5.3: CAE data graph schema, the lower parts reflect the overall development process, and the upper analysis discipline. (a) a portion of the vehicle development process, (b) safety requirements for a vehicle, (c) FE-model input, (d) FE-simulation result, (e) development history of the analysis, (f) FE-models comparison, and (g) ML applications.

also Section 5.3.

Usually, the FE-model representation of barriers or impactors does not change during vehicle development, while there are slight deviations, such as positioning or mass, for the robustness studies. Technically, however, the vehicle and, say, the barrier are in one FE model; the barrier is separated from the vehicle in the proposed CAE data model. This way, different crash scenarios share the same FE-model from the vehicle design. This separation allows the input FE-model to be reused for different analyses, makes it easy to report the complete safety status of a single design, and reduces the computational time required to compare inputs by ignoring changes due to barriers or impactors. Consequently, an FE-simulation (Sim) is the combination of a (Model) representing the designed vehicle and barrier (Barr) or impactor (Imp).

The input and output of the solver compose the main share of CAE data, dividing the data into the FE-model and simulation levels. The FE-model level represents the configuration at the start of the simulation, Figure 5.3 (c). At the simulation level, deformations and other mechanical properties over time quantify the simulated behaviour, Figure 5.3 (d). Additional metadata can be extracted from the input configuration of the simulation software; these describe the crash scenario, the vehicle characteristics, the development state, or the model/simulation parent. This parent-child relationship of simulations shapes the so-called development tree of a vehicle, ○– MODEL_REF –○, and provides the possibility to reuse former results for the missing result via the edge ○– MODEL_STATUS –○, Figure 5.3 (e). Here, each FE-model (Model) has an edge to the (Veh) node, and several models exist for a vehicle configuration during the development.

For filling the CAE data model with actual data, a challenge is to convert a given (Model) or (Sim) to the corresponding graph representation. Formally, the number of keyword entities in a single simulation is almost 4000 times more than the total number of available simulations for a developed vehicle. Consequently, domain knowledge is utilised

in data processing and appropriate semantics are defined for import to take advantage of the application of KGs to CAE data.

First, (Model) connects to all entities in the FE-model. When modelling the FE-model, the semantics are studied to classify changes, with the future goal of generating and recommending new models based on the development tree. The semantics of the FE model are designed to keep the resolution at the part level and its connectivity in this state. Roughly speaking, parts in an FE-model are a group of elements with the same properties, e.g. thickness and material. Each part of an FE-model resembles a node in the graph and has connections to its neighbouring parts based on its connectivity type, Figure 5.3 (c). In addition, to capture the structural role of the part, an edge of parts is added to (Pltf) or (Ubdy).

The specific FE-model entity selection strategy follows the result of the ModelCompare software [GPP17]. ModelCompare compares two similarly discretised FE models and organises their differences in a semantic fashion. Its outcome contains the pairwise comparison of the models and summarises the changes. Accordingly, (Change) is semantic for the changes of compared models. With this concept, it could be sufficient to model each (Part) to the database only if there is a change in the part. However, this way of modelling is not appropriate for the simulation data. Deformation of the same designed part differs due to its neighbouring parts, as long as slight changes are applied. Consequently, changes detected by ModelCompare are stored as ○— CHANGED_TO —○ edges, whereas all remaining, unchanged parts are connected via ○— SAME_AS —○ for the compared models, Figure 5.3 (f). Finally, note that (Semantic) nodes are generated as a parts container. This enables capturing the case of a part being split into several parts from one FE-model to the next.

Second, for the data modelling of simulations, a (Sim) node reflects a FE-simulation outcome, where its properties stem from global entities of the simulation, e.g. total mass or impact energy. Similar to the FE-model data, parts are the main entities representing the simulation, (Part)

Figure 5.3 (d). However, this connection reflects the CAE analysis characteristic and may differ based on the type of analysis. For crash simulations, the core problem is assessing the impact's energy absorption and managing the energy flow to prevent human injuries and energy features summarise the development fingerprint in Chapter 6. Consequently, as a piece of important outcome information, certain energy features of the parts are used to connect them via ○— NRG_PART —○[4] relationship to the (Sim). Note that thereby (Part) contains properties and relations from both the FE-modelling and simulation level, e.g. thickness and energy absorption features, respectively. Here, simulation states can consist of the energy absorption feature defined in Section 6.1, while the FE-modelling info consists of Property ID (PID), box centre, material (name and ID), the Center of Gravity (COG) or, when it applies, thickness (average and distribution). Geometrical features are a part's length, width, and height, along with the coordinate system of the FE-model (L-x, W-y, H-z).

The highest resolution level of a simulation is node and elements data. These entities' deformations and mechanical properties are outputs in each defined time step. Considering all these details may miss, say, small critical deformations overlayed by the overall deformation while analysing only at the part level is too coarse. Therefore, in the CAE modelling setup, some nodes and elements are defined as output sources for engineers, e.g. elements for cross-section output or nodes for accelerometer or intrusion assessment. For now, the storage is limited to these selected nodes and elements, (Node) and (Elmnt) respectively, Figure 5.3 (d).

Note that the eventual analysis aims are to detect simulation outliers, assess similarities of simulations, predict the so-called crash load-path, and provide a rough prediction of properties from one vehicle to another, in particular those sharing a platform design. ML analyses mainly

[4]NRG is an abbreviation for energy.

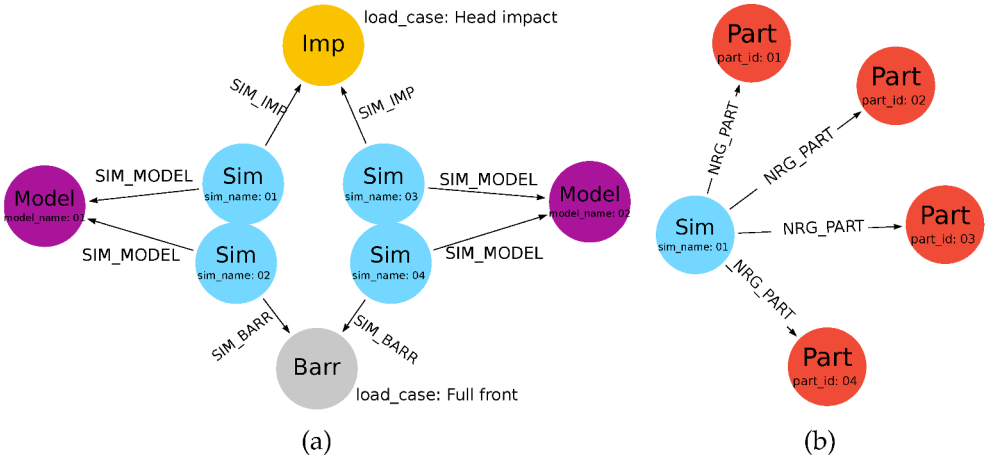


FIGURE 5.4: Illustration of the building blocks of the graph modelling. (a) A high-level simulation consists of one model and one impactor/barrier depending on the crash scenario (b).

extract features from the simulations or FE-models to assess similarities, e.g. feature engineering or dimensionality reduction. Extracted features depend on the analysis grouping setting, e.g. group of parts or simulations. To store the outcome of these analyses, three additional nodes are included (Des), (Behav), and (Grp) Figure 5.3 (g). Design (Des) and behaviour (Behav) bundle parts with similar features at FE-model and simulation, respectively. (Grp) is a group of different graph nodes in the database. Figure 5.5 illustrates examples of loading entities for each simulation into the graph modelling presented.

Several scenarios exist for (Des) and (Behav) connectivity. First, they connect to the stand-alone entities that address the entity-based bundling between many simulations. In this case, the features are independent of the grouped study. Furthermore, performing the analysis for a group of entities is sometimes beneficial, e.g. parts as a component or simulations as a development tree. A component is a group of parts whose role in the crash analysis is the same. In this case, the (Des) and (Behav)

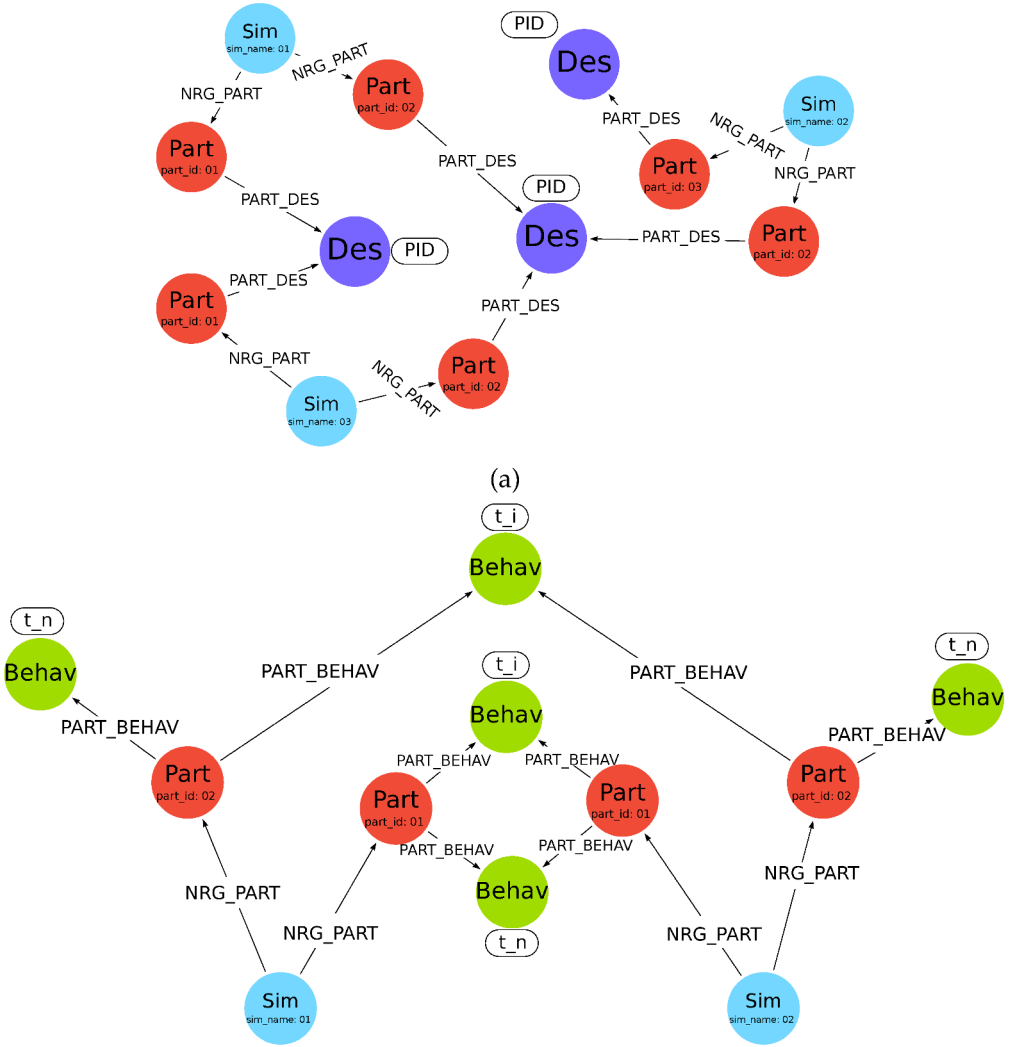


FIGURE 5.5: Continue, illustration of the building blocks of the graph modelling. More detailed components of a simulation are the parts that are filtered as energetic parts (a), while design and (b) behaviour nodes connect parts of simulations based on the similarity of the design or the crash behaviour, respectively.

are connected to the (Grp) instead and contain the information about the grouped entities.

Finally, the edge for (Grp) — GRP_FTS — (Sim) is added as a weighted edge to store feature extraction for grouped entities in the simulation level in addition to the possibility of direct aggregation of features extracted per part. To summarise, if (Des) and (Behav) connect to the FE-model or simulation entities directly, they are entity dependent, and if they connect to (Grp), they are group dependent. These three nodes can store the ML analysis output on any CAE data, e.g. (Node), (Elmnt), (Sim), (Model). Secondly, as a weighted edge, ○— SIM_SIM —○ is introduced, where the weight refers to a similarity prediction between the simulations and is the outcome of a graph analytics algorithm, for example, SimRank [JW02]. Note such similarity predictions and their usage as edge weights are described in Chapter 7.

Finally, the observation shows that it is common to share the FE-model between different disciplines during vehicle development. The graph modelling's initial target is crash safety, as the most complicated analysis in model size and computational time compared to other solid mechanics FE analyses. However, the graph modelling is still applicable for other solid mechanics CAE domains, with corresponding semantics for the requirements and the output quantities to assess the simulation, segment (b) and (d), respectively. An (Attr) is added to enable this extension and identify the analysis discipline with its attribute, e.g. safety and durability, in Figure 5.3 (a). Table 5.1 summarises the nodes and relationships of this schema.

TABLE 5.1: Node Lables of the graph modelling presented in Figure 5.3.

Nodes		Describtion
Label	Full Name	

Table5.1 (Continued).

Veh	Vehicle	The subject of the CAE analysis.
Pltf	Platform	A relatively large set of product components that are physically connected as a stable sub-assembly and are common for different final vehicles [ML97].
Ubdy	Upper-body	What deviates between the vehicles in simultaneous developments, e.g. sedan versus a minivan.
Year	-	The year that the vehicle will be on the market. This clarifies which regulations should be followed.
Attr	Attribute	The analysis discipline, e.g. safety and durability.
Prtcl	Protocol	The documents that specify the requirement for the safety test assessment.
Model	FE-model	FE-model of the vehicle disregarding the load-case.
Sim	FE-simulation	An FE-model simulation output that includes a vehicle and the load-case, e.g. a barrier or an impactor for safety analysis.
Barr	Barrier	The FE-model of a barrier for safety analysis, e.g. pole, rigid wall, moving barrier.

Table5.1 (Continued).

Imp	Impactor	The FE-model of an impactor that hit several positions of a vehicle in several simulations, e.g. head and lower leg.
Part	-	The main entities representing an FE-model and simulation.
Change	Changes	A change applied from FE-model one FE-model to the next.
Semantic	-	Parts container that enables capturing the case of a part split in comparing FE-models.
Connection	-	Different FE-model entities to model connections in the model, e.g. Rigid Body Elements (RBE), common nodes and welds.
Des	Design	Introduced class to connect parts of FE-models based on the similarity of the input data.
Behav	Behaviour	Bucketing the FE-simulation output features.
Grp	Grouping	Grouping the parts and their feature to be analysed for a desired functionality.

5.5 Applications

This work introduces an initial graph for crash simulations and Euro NCAP safety assessments as an example of CAE data and its requirements. Graph modelling enables two main applications: dynamic semantic-oriented reporting of the data and ML assistance in analysing the results. Consequently, the development and uptake of car-graph is supported by a web-based platform called CAEWebVis[5] to enable semantic reporting for CAE.

A web-based reporting for CAE is direct usage of the data. Web-based platforms for CAE data were introduced as early as CAE-bench [Häg+10] and recently as an open-source CAE platform [Lee+20]. Still, even after two decades, documentation using web technologies is not sufficiently integrated at most OEMs. The obstacles are the high overhead for implementation, lack of software independence from the company data structure, and usage of relational databases, which are not as agile as graph databases. A web-based user interface is used to enable project members from different teams to access the CAE results, taking advantage of web standards for ease of use, particularly without detailed CAE experience. Consequently, a framework is proposed that auto-generates and organises results at a middle level and presents it as a web page on the company server. CAEWebVis envisions a data organisation following web semantics, where it connects simulation results to attribute requirements and design limitations. Such a connection of documents will increase the learning rate between the different disciplines. Ultimately, these semantics enable the establishment of the car-graph. CAEWebVis's ultimate vision is to have a tool with low overhead to support the developments required to build safer cars.

The developed CAEWebVis framework for an enhanced exploration of the data is based on the graph modelling. Component-based development is used to have similar visualisation of data for different attributes and analyses. The platform's target is to link data from requirements

[5] Accessible at CAEWebVis.scai.fraunhofer.de/.

to assess CAE performance and competitive market performance. The concept of graph modelling combined with the micro-component web development concept provides enormous flexibility and efficiency in reporting. This flexibility is an outstanding capability since changes to CAE data are often, e.g. improvement of an impactor modelling or a new crash scenario with a bicycle. Moreover, ML methods are not yet established in this domain, which needs flexibility in data modelling. In addition, the new flexible reporting capability allows users to interactively and efficiently select and combine different visualisations and dynamically compare analyses across many simulations. This is to be seen in contrast with the traditional way, where it is time-consuming to collect different reports from several engineers.

As an example, let us consider pedestrian safety requirements; these assess the vehicle's safety for four types of impactors: lower leg, upper leg, child head, and adult head impact. For each impactor, a range of probable impact positions is employed. Consequently, each vehicle design leads to about 250-350 simulations, which highlights the benefits of dynamic reporting. Here, consider the assessment measures for head impact and upper and lower leg. The connection of this visualisation to the protocols, e.g. Euro NCAP, supports the CAD engineer in a better understanding of the problem and assures attribute leaders that the project is following the correct variant of assessment. Additionally, the usage of the introduced representation of different data sources in one graph model enables a quick comparison of the vehicle under development with the market performance.

To achieve this, CAEWebVis has two types of views: zoom-in and zoom-out views. These views maintain classical reports and ML outcomes. The zoom-in view provides visualisations that are easy for humans to evaluate, e.g. a single detailed visualisation such as a one-pager with animations and curve views, e.g. for accelerometer or section forces, or a combined view for less than ten simulations. On the other hand, the zoom-out view is for looking at many simulations simultaneously, such as a scatter-plot view for ML embedding results or a summary status

view for protocols assessment for many simulations. Based on the proposed graph modelling, these visualisations can be agile and efficiently adapt for different crash scenarios, ML analysis, and other CAE data in broader aspects, e.g. visualising the status of a model, entities in segments (d) and (g) in Figure 5.3, based on its connectivity to different barriers and attributes.

Regarding ML applications, previous methods [ITG19] and ongoing research [SlPG21] are available to be transferred to exploit the introduced data model. Furthermore, one of the properties often used in graph mining is the node's degree and corresponding ranking of the nodes. For example, following a high degree (Change) can be a good recommendation for inexperienced engineers. Moreover, (Des) and (Behav) nodes with a high degree reflect common parts and CAE analysis features, respectively, in a development stage, which highlights fundamental parts and essential times during the crash. On the other hand, low degree (Des) nodes reflect components that are outliers or in unexplored design space. Additionally, cross-domain parts are easily identified with querying nodes with high-degree changes that are common between attributes.

Nevertheless, using Graph Machine Learning (GML) methods to gain insights from CAE data is still challenging. Potential analysis goals include predicting the similarity of simulations via ○— SIM_SIM —○ or cause-and-effect predictions for each simulation ○– CAUSED_TO –○. These are weighted edges, where the weight refers to the similarity predictions between the simulations and the strength of linkage of the cause-and-effect from model changes to the simulation behaviour, respectively.

In the past years, the research on GML has focused on methods for analysing data, such as social media interactions or fraud detection, where the graph is homogeneous. For CAE data, a significant challenge is to apply GML methods to a heterogeneous graph, a graph with multiple types of nodes. However, a change in the data modelling required

merging two nodes' information to generate a bipartite graph to make existing methods applicable. In addition to the difference in the data entities and interactions, CAE data has in comparison also different proportions of data size. Recently, Graph Neural Network (GNN) methods have been influential in graph analysis. However, the data size typically required for these methods is not fulfilled in the CAE domain.

5.6 Summary

This chapter introduced data modelling for CAE data, which enables searchability, filtering, recommendation, and prediction during the development process. Besides, the CAE data was linked to the required protocol, and the comparison of vehicle safety performance was considered. The presented graph modelling uses CAE crash simulation as a source for the vehicle development process and the Euro NCAP web page as the source for assessment and benchmarking. Consequently, the platform CAEWebVis is developed as an example of semantic reporting based on graph modelling to illustrate to automotive engineers the advantage of dynamic reporting. The engagement and feedback of OEMs shall enrich the semantics, which would empower car-graph in exploring and predicting CAE data and step toward safer vehicles. However, it is still the early stage of car-graph research, and further research is ongoing for empowering ML algorithms on CAE data that will extend this graph modelling. The following chapters use this data modelling and focus on enriching the data with feature extraction and enabling knowledge discovery with GML methods.

6 Knowledge discovery assistants for crash simulations with graph algorithms and energy absorption features

Finite Element (FE)-modelling improvements resulted in more and more detailed simulations with continuously intensifying complexity of the data. Moreover, the growth of computing power has increased the number of simulations. Due to this data and complexity growth on the one hand and limited availability of engineering time on the other hand, simulation result data is often unexplored. Furthermore, the complexity and size of the simulation data also prevent the direct application of most Machine Learning (ML) methods on the simulation data, e.g. to capture and detect patterns and trends. It is also observed that determining a simulation data representation based on engineering principles, which in particular helps to quantify crash behaviour, is a largely unexplored research area. Here, crash simulation data is represented in a graph database, including suitable physical properties, to empower graph-based ML algorithms.

There are various options available for the selection of physical properties from the different Computer-Aided Engineering (CAE) outputs. Most of these depend on the simulation setups and are too costly for comparing many simulations. Hence, this work proposes the usage of a few scalar features stemming from the Internal Energy (IE) of components, both in the graph representation of simulation data and for the analysis of simulation results. The fundamental physics of a crash

accident is to absorb the energy of the impact through structural deformations, causing an increase in the IE. The close physical connection of the IE properties to the crash problem and their ease of use, with no need for further simulation pre-processing, makes them a compelling candidate for further investigations. This chapter first demonstrates the capability of the IE features to capture the differences between simulations with minor changes. Later, it shows that the chosen IE features are sufficiently characteristic for any given simulation (with 10-12 million elements) to allow Graph Machine Learning (GML) methods to perform well with the comparatively small number of simulation data (200-300).

The car-graph is built, focusing on the IE in the CAE outputs. Features are introduced based on the IE, which allow visualisations that enable the engineer to extract additional knowledge from and gain insight into simulations. For example, one can study part similarity, summarise development stages, or analyse crash behaviour. Finally, the proposed energy features are integrated into a heterogeneous weighted graph, which allows the identification of outliers and absorption trends, as well as a visual clustering with force-directed GML methods [Jac+14].

Figure 6.1 summarises the approach, where new visualisations are introduced for knowledge discovery with the support of graph databases. First, in the section 6.1, features are extracted for energy curves. Section 6.2 then examines the ranking of design components during the development stages. Later, energy features are explored to identify similarities, and design exploration fingerprint is introduced in Section 6.3. Further, Section 6.4 uses graph visualisations to advance CAE knowledge extraction. Figure 6.1 summarises the approach in this study in introducing new visualisation for knowledge discovery with the support of graph database and GML methods.

6.1 Energy features

The deformation structures of the vehicle shall absorb the kinetic energy so that the occupants and pedestrians have the least possible injuries. In

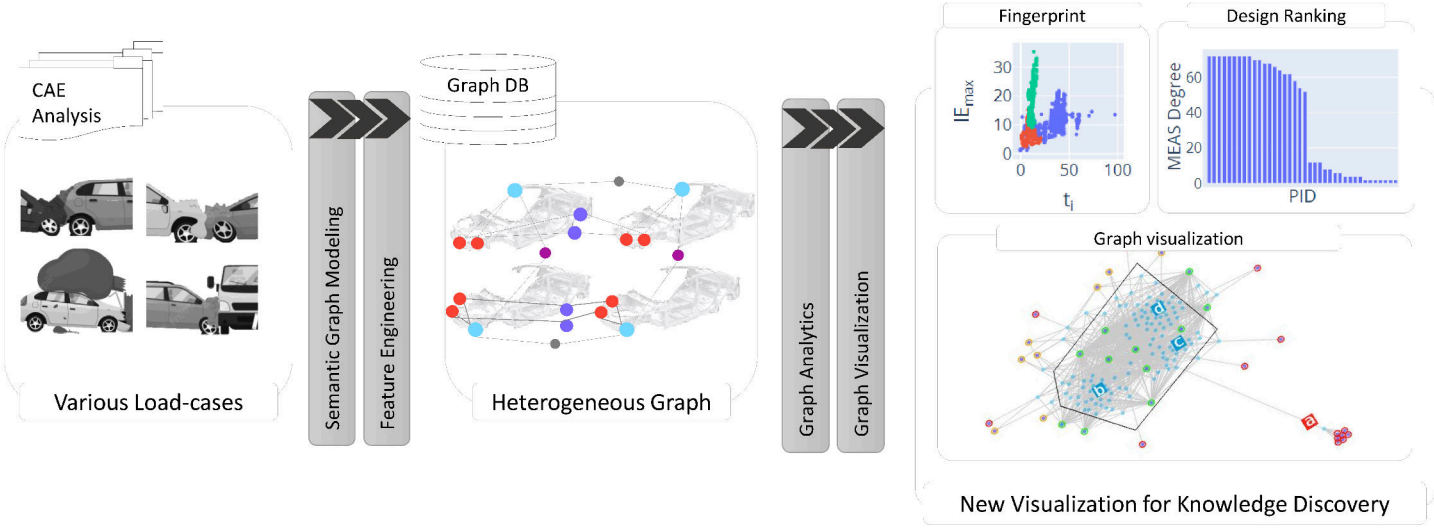


FIGURE 6.1: The graph-based analysis and visualisation workflow involves loading crash simulations to a graph database, including the computation of engineering features and data visualisation for knowledge discovery. The node colouring in the heterogeneous graph reflects different node types and follows the schema in Figure 7.7, fingerprints, degree distributions, or force-directed graph layouts are examples of visualisation approaches.

the case of shell structures, this is done by means of suitable deformation patterns. The main underlying physics of the crash analysis problem is the energy dissipation performance. CAE engineers often determine the crash behaviour mainly by analysing the intrusion, acceleration, force, deformation, and failure. These parameters are assessed and correlated with reality visually and quantitatively. They often do not take into account the energy dissipation performance. However, the energy is a solver output available for all simulation parts, whereas other measures require specific FE-model preparation.

The FE solver outputs energy per part over time, a so-called energy curve. Parts in the CAE model preferably refer to each vehicle component. Despite this, CAE modelling techniques require arranging vehicle components into several properties, for example, due to changes in the thickness and material. Consequently, CAE models have many parts (1500-4000). The number of parts confronts CAE engineers with a practical assessment of energy curves. Therefore, it limits the use of energy curves in the workflow to, e.g. stability investigations (checks the simulation's total energy) and outlier entity identifications (e.g. parts with negative energy).

TABLE 6.1: Introduced scalar features representing an energy curve.

Energy Features		
t_i	initial absorption time	The initial time that the energy absorption starts.
IE_{max}	absorbed energy	The max. IE absorbed by the part.
t_n	absorption time	The time in which IE_{max} is reached.
Δt		$t_n - t_i$

This work claims that energy curves hold information to characterise the simulation crash behaviour. Data analysis on energy curves will simplify data processing to represent the crash behaviour based on a

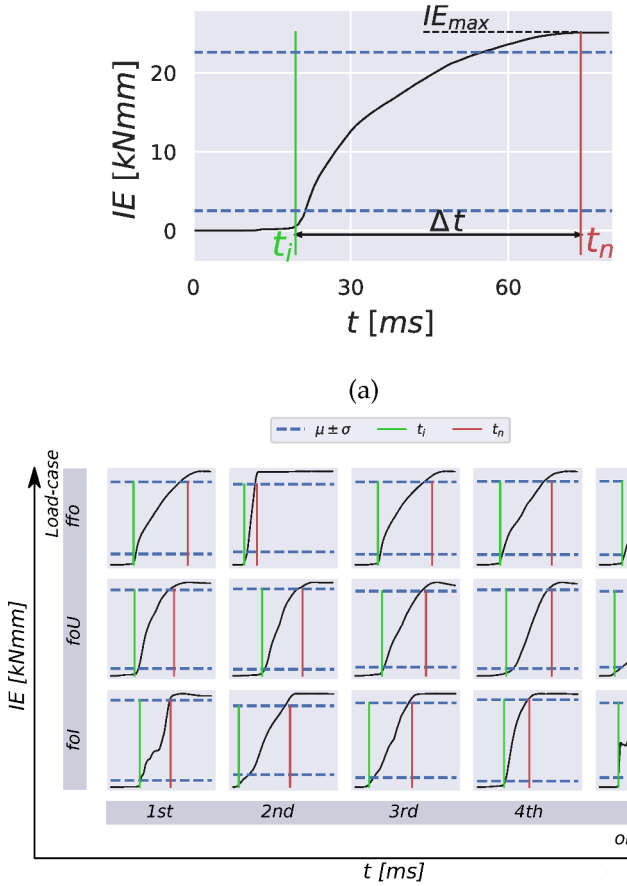


FIGURE 6.2: (a) The internal energy output of the solver over time for a single property with the three features $t_i, \Delta t, t_n$ that characterise the energy absorption. The dashed blue lines that intersect with the energy curve visualise the standard deviation approach. (b) Diversity of internal energy output over time for the five most energetic parts, Table 6.2, in three load-cases, Table 4.1, with calculated initial and end of the absorption time and standard deviations. x and y axes are normalised.

TABLE 6.2: Part name for Figure 6.2b, load-case information in Table 4.1.

load case	1st	2nd	3rd	4th	5th
ffo	SM R-I	CB L-V	SM LHS-I	SF L-U	SF R-U
foU	SM R-I	CB L-V	SM LHS-I	SF L-U	SF R-U
foI	AP L-I-L	fascia-F	WA-F	cowl-L	WR-L-F

SM: side-member, CB: crash-box, SF: sub-frame

WR: wheel rim, WA: wheel arch, AP: A-pillar

R: right-hand side, L: left-hand side

-U: upper, -V: vertical, -L: lower, -I: inner, -F: front

few features. Figure 6.2a shows the energy curve for the most energetic part of a complete vehicle simulation in a front overload load-case, ffo, with a total initial kinetic energy of 453 $[kNmm]$ (initial velocity of 64 $[km/h]$). The shape of IE over time is approximately a sigmoid curve, except for parts with negative energy due to numerical error. From the crash analysis perspective, measures with the potential to analyse the crash behaviour from this curve are initial absorption time, absorption end-time/period, and absorbed energy, Table 6.1. These features indicate three abstract characteristics of the energy curve. Absorption time is defined with Δt and final absorption time (t_n) as relative (to initial time) and absolute measuring, respectively. For now, both features will be kept, and the functionality of each will be studied in different applications.

Figure 6.2b shows representative examples of IE curves and the features extraction over several simulations and parts. These curves belong to three simulations from three different front impact load-cases (Table 4.1) that are selected randomly. For each simulation, the five most energetic

parts are plotted (part names in Table 6.2 and part definition in Section 4.3). In these examples, the shape of the curve during the absorption time (Δt) is nonlinear for some parts (first and fifth part in foI load-case). This nonlinearity indicates the probable necessity of additional features or more complex methods for characterising the absorption interval.

yy The max of IE (IE_{max}) is defined as the maximum of the IE curve, and the following describes the time extraction features initial absorption time (t_i), t_n and Δt. The accuracy of the timings depends on the time interval output of the solver. Here, three approaches are investigated to estimate the features based on the IE behaviour: thresholding, derivative change, and standard deviation spread. With thresholding, one considers the time when the IE crosses a pre-defined threshold value. The derivative-based method calculates the IE derivative ($\dot{IE}$) and determines a significant change. However, using the spread $\mu \pm \sigma$, upper and lower thresholds are considered depending on the mean μ and standard deviation σ of the IE for the part over time.

Figure 6.2 shows the methods for t_i and t_n versus the time standard deviation. To summarise the observations, the derivative method is more suitable for t_i due to its sensitivity in capturing trigger time. However, thresholding performs more desirable for t_n since IE growth is saturating at the end. Furthermore, the standard deviation approach fails for the parts with a long absorption time or negative IE in the initialisation. The following sections compare thresholding and derivative-based methods for t_i and t_n in more detail. For this, visual engineering judgment is used. Both methods are considered on random samples from three complete vehicle front load-cases in four development stages, Table 4.1. For each simulation, the 20 most energetic parts are considered.

As mentioned, the features are not continuous values. Their resolution depends on the solver settings for the timestep output, which can vary from 1 $[ms]$ to 0.001 $[ms]$. Consequently, the features binning is according to the timestep definition. Further and detailed investigation of binning and resolution is beyond the scope of the initial study of energy

features. Further, we focus on these three features, although considering other features during the absorption may contain more component characteristics during a crash simulation. However, extracting more features is out of the scope of this work, where the focus is to investigate the potential of features from energy curves, but not to find the best approach to achieve this. The features described have been chosen for their simplicity of use and interpretation.

6.1.1 Initial time

The initial time t_i for each part reflects when a part begins to absorb the impact energy. In crash simulations, there is a gap between the start time and the time when a specific structural part gets affected by the crash, which makes finding the exact time imprecise. Here, the extracted t_i from the thresholding and derivative-based methods are close for most of the studied parts. Furthermore, the parts with significant differences in the two computed t_i are examined and compared to each other. Figure 6.3 presents an example of such a part with significant differences, together with a part where both approaches give similar t_i. From visual engineering judgment, part two starts to absorb energy earlier than part one, whereas the thresholding method extracts the same time ($t_i \approx 65\ [ms]$) for both parts. However, the derivative-based method computes for part two an earlier time ($t_i \approx 40\ [ms]$) than the thresholding method, which is preferred from an engineering perspective. Therefore, further investigations with the derivative-based method are under consideration.

The derivative-based method requires curve filtering due to the non-smoothness of the curve. Filtering methods from SciPy.signal are examined. From lfilt[1], filtfilt[2] and sosfilt[3] the lflit with FIR filter is selected (lfilt, sample number n=75, b=1/n a=1), which smoothens the

[1]Filter data along one-dimension with an IIR or FIR filter

[2]A digital filter forward and backward to a signal.

[3]Filter data along one dimension using cascaded second-order sections

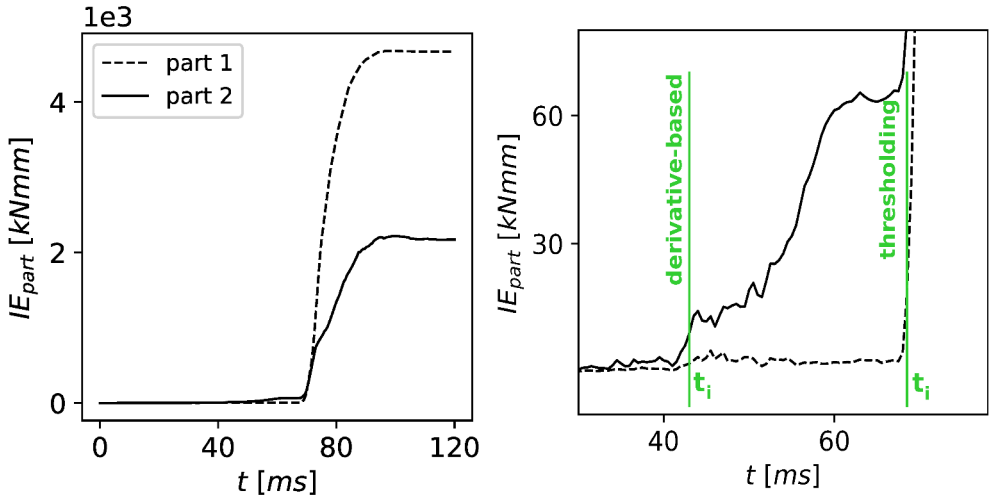


FIGURE 6.3: (a) IE curves for two different parts. (b) zoomed view of (a), the derivative-based method for part two (left marker) gives an earlier, and preferred, t_i than thresholding (right marker).

curve without any time shift, Figure 6.4a. With this filter and a min-max normalisation of the IE derivative, t_i is extracted as the time when the derivative is above 0.005. Both methods' lower limits are selected based on visual engineering judgment and visual tuning of the explored data. Figure 6.4b shows the result for a selected part.

6.1.2 Absorption time

The absorption time interval Δt is defined as the time interval from when the part's IE increases until it stabilises to its maximum. The start time is t_i from the last subsection, and the time at the end of absorption is t_n. The result of the thresholding of IE_{max} and the derivative are compared to extract t_n. For the thresholding, in order to exclude the gradual energy increase at the end of the simulation, an additional factor y is introduced for IE_{max}:

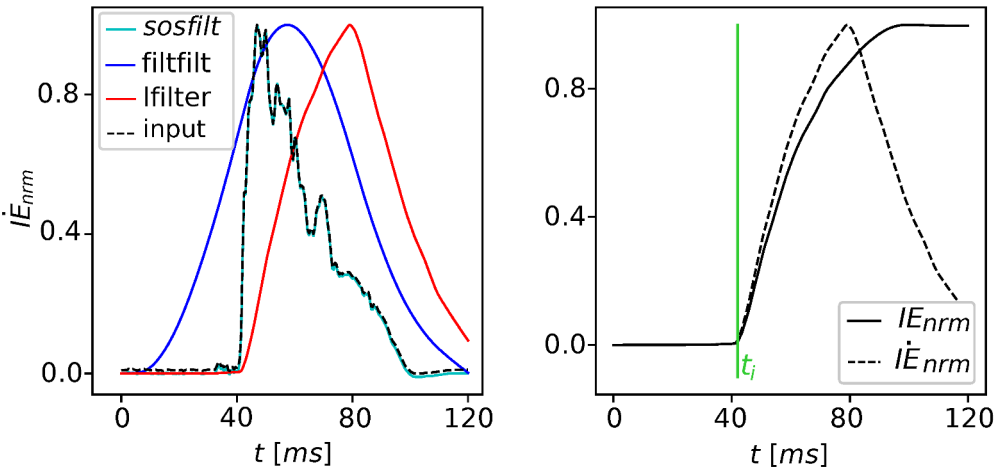


FIGURE 6.4: (a) Several filters are applied to the derivative of the IE over time, where filtfilt[2] and sosflit[3] miss the ramp-up time. (b) Normalized IE and normalised $\dot{IE}$ that is filtered with lfilter[1], t_i extracted based on $\dot{IE}$.

$$t_n = \max_t \{t \mid IE(t) \leq y \times IE_{max}\},$$
$$\Delta t = t_n - t_i. \tag{6.1}$$

The factor y is set to 0.95 based on a visual judgment from an engineering perspective on randomly selected simulations. The derivative method sets the second derivative of IE to zero to find the maximum of IE and its time for t_n. This calculation requires filtering, using the same filters as for t_i. However, here, a time shift of the filtering is inevitable; see Figure 6.4b. Consequently, the second derivative does not allow the reasonable extraction of the absorption time. Therefore, the maximum percentage time provides the best result for t_n.

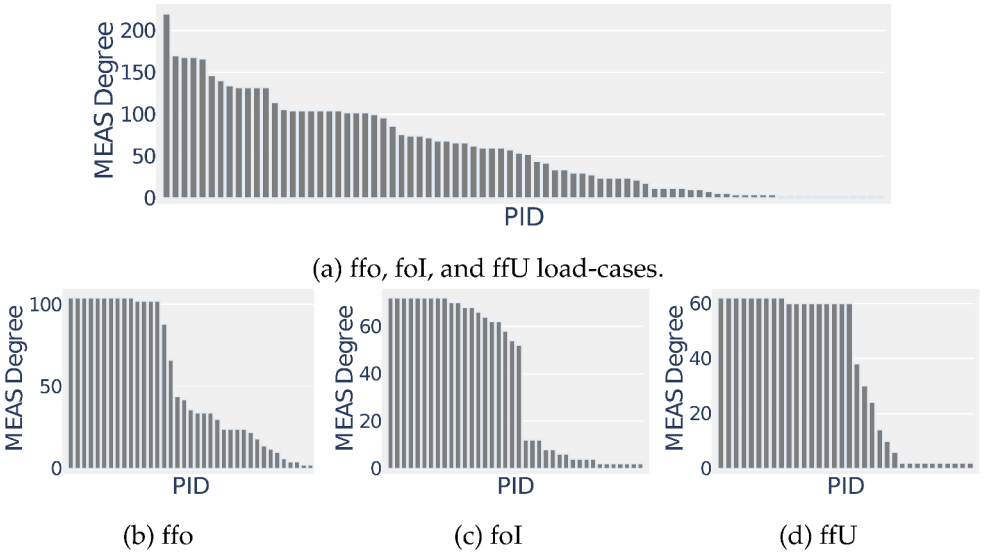
6.1.3 Discussion

Lastly, the standard deviation of IE is calculated for the parts, Figure 6.2b. The crossing of IE with $\mu \pm \sigma$ refers to t_i and t_n. However, there is a deviation in the result for different parts compared to the other method. Two extreme examples, not shown in the figure, show undesired results are curves with long absorption time and the components with spring-back FE-modelling, i.e. negative IE in the initialisation. In both situations, the standard deviation is relatively tiny and causes a higher value for t_i and a smaller value for t_n, respectively.

Note that parts with common t_i and t_n in one simulation are simultaneously involved during the crash. Identifying such simultaneous parts can be used to identify parts for grouping as one absorption block. But, such a grouping is out of the scope of this thesis and part of future work. It also filters out the parts that behave similarly in terms of energy absorption by considering parts that share all three features for multiple simulations.

6.2 Query database

Graph analysis methods allow simple data explorations, discover non-trivial patterns in the data, and reveal behaviours. One of the used properties is the node's degree, and one can rank the nodes accordingly. A ranking of (Des) nodes extracts common parts in a development stage and reflects fundamental parts. Further, low degree (Des) nodes reflect components that are outliers or are in an essentially unexplored design space, graph modelling described in Chapter 5. A degree ordering of (Behav) nodes can, for example, extract common timings of behaviour in a development stage, e.g. using the introduced energy features, which reflects essential times during the energy absorption. Such selection procedures allow automated post-processing scripts to support the CAE-ML workflow instead of requiring interactive user selections. High-ranked parts in a development stage for a load-case identify required parts in energy absorption. High-ranked parts are more reliable than



(a) ffo, foI, and ffU load-cases.

(b) ffo (c) foI (d) ffU

FIGURE 6.5: Degree distribution for the Des nodes for the China Euro Vehicle Technology AB (CEVT) data from the early stages, graph modelling described in Chapter 5.

just filtering the most energetic part in a simulation since outlier parts with high energy might exist due to FE-modelling errors.

A ranking of Des nodes, i.e. according to Property ID (PID), can summarise information relevant to the engineering process for problem-solving. While the node degrees in real-world, large-scale networks often follow a power-law distribution, i.e. a fast decline in the degrees [New05], this is not observed for the Des nodes that reflect PIDs, Figure 6.5. Here, the distributions for each load-case show some parts with high degrees, some in the middle, and the remainder with small degrees.

Generally, for this data, high and low-ranked parts correspond to essential and outlier parts, respectively. A significant degree drop can also help identify the number of essential parts in a load-case. Additionally,

the middle-degree parts are the ones that are not dominant in all the simulations and are neither outliers. Consequently, these parts are interesting structural components that potentially change the crash behaviour and summarise the simulation design scenarios. Such middle-degree parts can be valuable input for inexperienced CAE engineers to identify parts that affect the crash behaviour. For example, the fast transition of the degree in the foI load-case, Figure 6.5c, compared to the rest, indicates that the number of parts affecting the load-case is limited or that the engineer has performed a limited exploration of the design space.

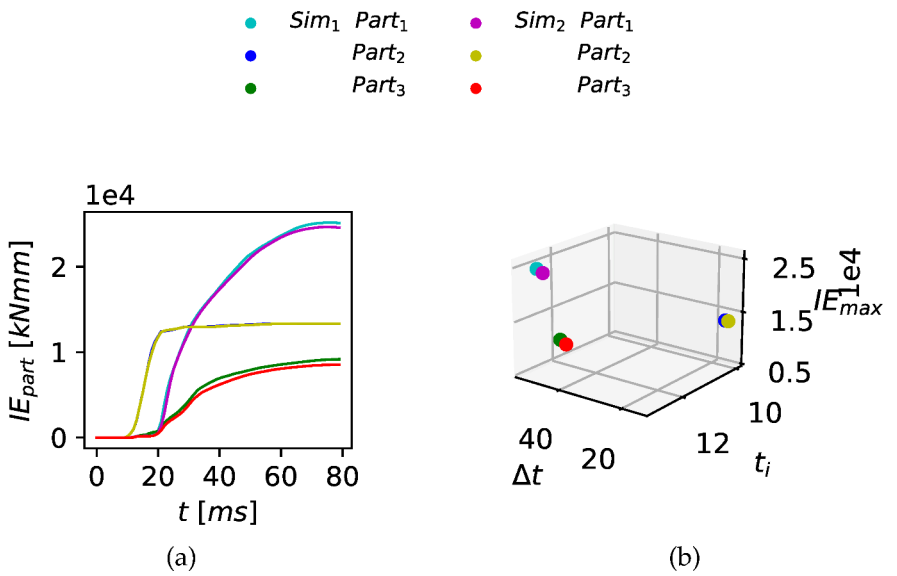
6.3 Scatter visualisation

Data visualisation is a key component in a typical data analytics project. The main aim of data visualisation is to identify patterns and trends that are hidden behind the data. An explorative visualisation of the data rather than descriptive analytics, which describes the data in a summarised way, provides a way for generating insights from the data [SSS18].

Here, several data visualisation techniques were proposed for better data exploration of crash simulation data. In particular, energy features from Section 6.1 are considered as a data representation for each part, where a scatter plot is used for visualisation. Each point in the scatter plot refers to one part of the simulation, and its coordinates are the part's energy features. This visualisation allows for assessing the similarity of the underlying energy curves, identifying outlier parts, finding the similarity in component-wise crash behaviour, and generating a visual design exploration fingerprint for numerous simulations.

6.3.1 Curve similarity

The energy features were selected to extract the main properties of the energy curves. Therefore, they enable the assessment of the similarities of energy curves. For example, Figure 6.6 shows a scatter plot for three



(a) (b)

FIGURE 6.6: *IE* curves similarity with scatter visualisation (a) *IE* curves from three identical components in two simulations with the same load-case and (b) the scatter plot for their energy features.

pairs of parts from two different simulations and the corresponding energy curves. This figure indicates that the similarities of energy features of parts are related to the similarity of the corresponding curves. Consequently, the scatter plot of the parts' energy features facilitates visualising clusters with similar behaviour from an absorption perspective.

Note that for three-dimensional (3D) visualisation, it is more illustrative to have independent variables, which facilitates the separate investigation of each feature. Therefore, the 3D scatter plot uses Δt as the energy absorption time because t_n includes the effect of t_i, whereas Δt is independent of t_i. However, if one wants to include t_i information, using t_n in two-dimensional (2D) visualisations can be advantageous. Generally, the weighted sum of the energy features can be used to measure the curve similarity. Here an open question is the normalisation and

weighting of the energy features, which likely also depends on the analysis goal in the application. To keep this simple, the focus here is on exploring and understanding individual energy features.

6.3.2 Part similarity

Here, the detection of geometrically matching components with energy features is under investigation. The underlying assumption is that components are geometrically correspondent if they are located symmetrically in the vehicle, their undeformed geometries mainly overlap symmetrically, and their deformations are symmetrical. One straightforward use case is capturing similar energy absorption by symmetric parts of the vehicle structure in a full-frontal impact. The similarity is due to the almost symmetrical design of the vehicle on the Left Hand Side (LHS) and Right Hand Side (RHS). Moreover, the full-frontal load-case affects the LHS and RHS of the vehicle structure symmetrically.

Figure 6.7 illustrates this use case. It contains the four most energetic parts of 50 simulations of a full-front load-case in one development stage. A similar PID of the thereby selected parts implies that their geometries are more relevant than the remaining parts in the vehicles[4]. This data overview shows that the four most energetic parts generate two distinct point clusters. Each cluster holds two parts, and each pair consists of the RHS and LHS of the corresponding geometrical part.

As a final result, the energy features detect a symmetrical behaviour in the crash simulations and summarise the distribution of the design exploration. While the dataset is limited here, i.e. only one load-case and one development stage is considered, this applies more generally. An example, discussed further in Section 6.3.3, is for distinct point clusters, where if the PID changes for a component, one can now connect components between different development stages.

[4] Assuming the PID remains fixed during one development stage.

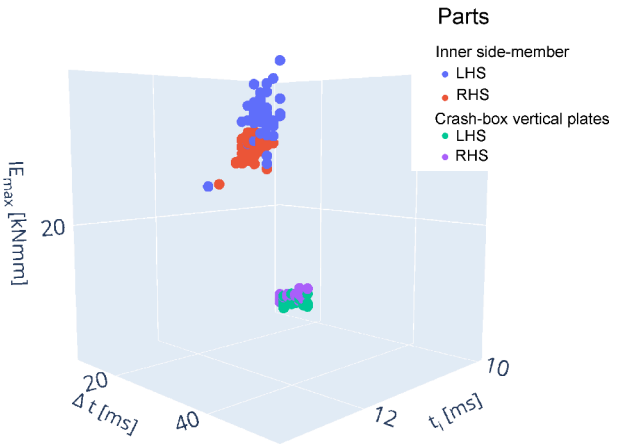


FIGURE 6.7: Energy features for symmetric components, side-member, and crash-box plates for 50 simulations. The colouring of the points is based on the semantics of the parts. The range of design exploration is highlighted by the density of the point clouds for each part, with the distance between points indicating the similarity of the parts in different simulations in terms of energy absorption.

6.3.3 CAE design exploration fingerprint

Summarising the behaviour of design exploration with many simulations is an additional application of energy features. Design Exploration Fingerprint (DEF) is introduced as a data visualisation, which is the scatter distribution of the energy features. The scatter plot contains energy features from energetic parts of many simulations in one or several development stages. DEF of a group of simulations assists in assessing the vehicle's development process. Four colour schemes are explored that visually group the data points differently during the exploration. The colour schemes are according to PID, IE_{max} order, development stage, and load-case, respectively.

The colour schemes reflect different use cases for the data exploration. The PID colour scheme visualises the design space for each part. Nonetheless, due to possible PID variations between load-case or development stages, the PID colour scheme is limited to simulations in one development stage and one load-case. The second colour scheme uses the IE_{max} order in a simulation, which visualises the parts order in the energy absorption for each simulation. This visualisation is informative if coupled with the PID colour scheme to highlight the permutation of parts in absorption behaviour. Additionally, the fingerprint with the development stage colour scheme emphasises load-cases in one/several development stages. Finally, the load-case colour scheme demonstrates the evolvement of the platform in several/single development stages independent of PID change between several load-cases.

Here are data visualisation examples with DEFs for CEVT's real-life development stages. These examples show the types of engineering information that DEFs can visualise. To better demonstrate a 3D plot in a 2D figure, the DEF is presented as a matrix scatter plot; see Figure 6.8. Matrix scatter plots use two features for absorption time (Δt and t_n) since the coupling between t_i and Δt is lost in a 2D visualisation. Additionally, the range of end-time or absorption period difference remains identifiable, i.e. when comparing the spread shape between different platform

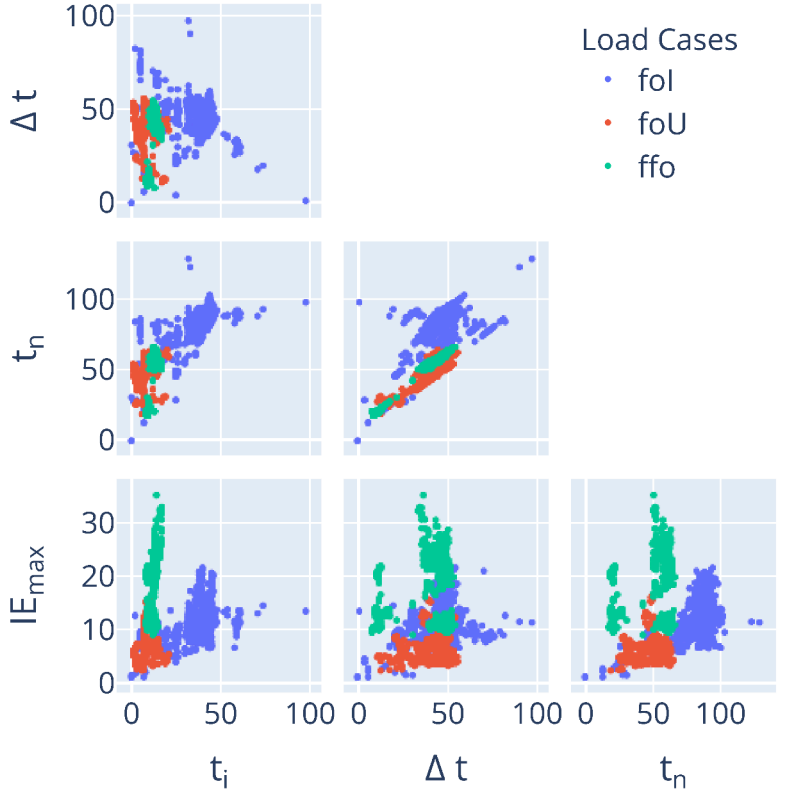


FIGURE 6.8: DEF load-case scheme, five most energetic components in simulations from four development stages, (t_i $[ms]$, Δt $[ms]$, IE_{max} $[kNmm]$). Simulation specification in Table 4.1.

structures, by just considering t_n or Δt.

Note that an interactive application is the most helpful visualisation for exploring the data using DEFs. For example, the application can enrich the data by connecting each point in the scatter plot to additional information such as pictures, deformation videos, or metadata of the part and simulation.

PID scheme

The DEF in each plot is an imprint of the distribution of the energy features independent of the PID. Consequently, the pattern shown by the PID colour scheme conveys the parts between development stages even though the PID has changed. Figure 6.9 uses the PID scheme for an early and a middle development stage in a 2D $IE_{max} - t_n$ fingerprint. This visualisation shows that even though the part numbering differs in these two development stages, the shape of the scatter plot and absorption order identify the pairwise components that correspond in energy absorption, see the point clouds (a) and (b) in Figure 6.9 and Table 6.3. Here, cloud (a) consists of the inner plate of the side-member. For both stages, the cloud includes only two PIDs referring to the LHS and RHS parts. However, an offset along the y-axis shows a decrease in the mean of IE_{max}.

Likewise, cloud (b) contains two components. The upper points belong to the sub-frame, and the lower ones to the outer wall of the side-member. However, this cloud holds many different PIDs. The variation of the PID for the sub-frame highlights the critical components studied in the CAE-based analyses.

Additionally, the cloud distribution shapes a pattern where it addresses the difference between development stages, e.g. a change in the FE-modelling technique or a change in the vehicle concept. In this example, the vertical and horizontal plates of the crash-box have separate

TABLE 6.3: PID part name in two development stages for Figure 6.9.

Early stage	Middle stage	Part Name	(Cloud Label)
10020420	10021870	LHS-I	(a)
10021520	10021320	RHS-I	side-member
10022010	10021830	LHS-O	
10021350	10021220	RHS-O	(b)
18620080	18620090	LHS-V	(c)
18620120		RHS-V	crash-box
18620070	18620070	LHS-H	
18620110		RHS-H	(e)
55021040		LHS	lower load path (d)
55021060		RHS	
55131390, 55132410	55132390, 55131220	LHS	sub-frame (b)
55131400	55131440, 55132820	RHS	
	55131010		

RHS: Right-Hand Side, LHS: Left-Hand Side

-U: Upper, -V: Vertical, -H: Horizontal, I: Inner, O: Outer

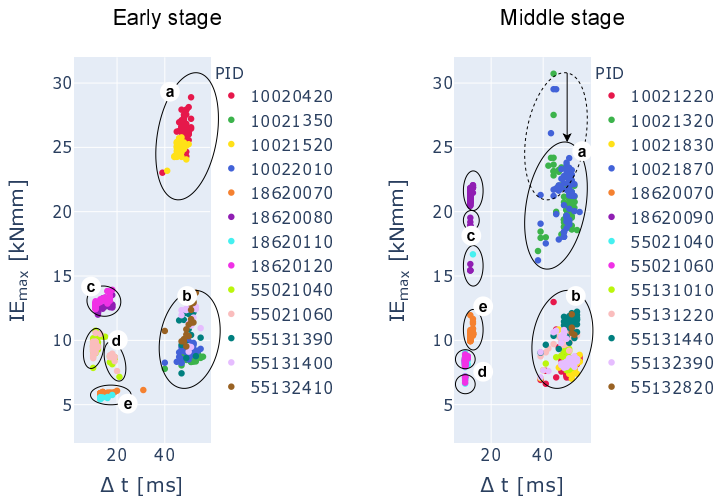


FIGURE 6.9: DEF with PID colour scheme, CEVT data ffo load-case, Part name in Table 6.3.

PIDs for RHS and LHS. However, these are modelled as one in the mid-stage. Consequently, the absorption has doubled; see clouds (c) and (e) in Figure 6.9.

Finally, point cloud (d) belongs to the lower load path components RHS and LHS. It keeps its dual behaviour, but this visualisation summarises that the absorption interval is more stable in the later stage.

Order scheme

The ordering scheme visualises the IE_{max} order for each part in a simulation. The ordering scheme visualises the point cloud for the energy absorption order combined with the PID scheme. Figure 6.10a compares the ffo load-case in two development stages using the IE_{max} order scheme for each simulation's eight most energetic parts. The number of point clouds for each placement captures the number of scenarios for evaluating the permutation of the energy absorption. In the primary

stage, bifurcation exists for the sixth, seventh, and eighth-ordered components; however, in the early stage, bifurcation starts right after the second part. Besides the number of scenarios, the density of the point clouds can reflect outlier simulations or an unexplored design space. For example, a few simulations in the early stage have the fifth and sixth parts in the left point cloud.

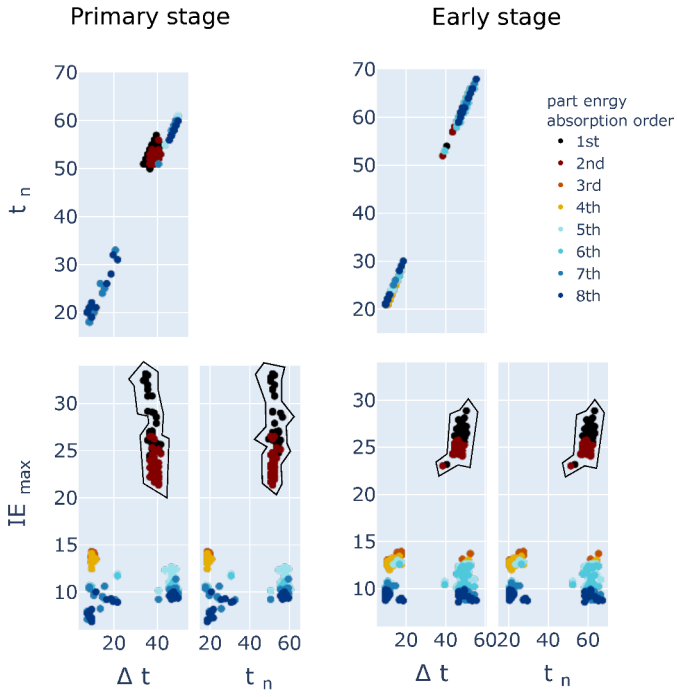
So far we looked at IE_{max}, t_n, and Δt features. Additionally, the t_i fingerprint achieves a different knowledge summary. Figure 6.10b shows the initial time for the same development stages as Figure 6.10a. Here, we see that the two most energetic parts, the side-members, have noticeable differences in the t_i spread. The deviation is also captured in the $t_n - \Delta t$ plot, Figure 6.10a. The early development stage is more stable in trigger time than the primary development stage and limits the exploration of designs. Consequently, the t_n and Δt relation becomes more linear. Therefore, $IE_{max} - \Delta t$ and $IE_{max} - t_n$ provide similar DEFs in the early stage. However, in the primary stage, the relation of t_n and Δt is non-linear for the side-member. Consequently, the point cloud shape of $IE_{max} - \Delta t$ and $IE_{max} - t_n$ differs in the primary development stage.

Development stage scheme

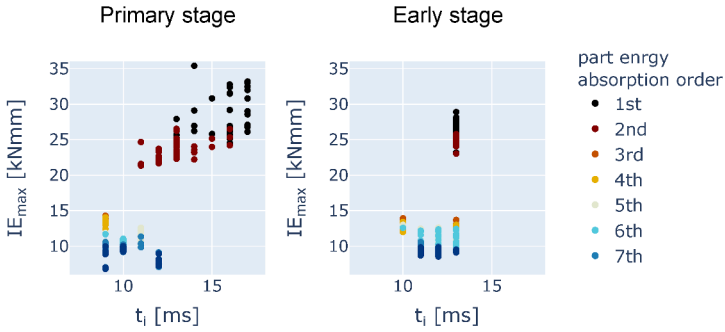
This colouring scheme is beneficial for summarising the trends of the development stages. In this visualisation, t_n is preferable to Δt since an absolute value is better for comparing development stages. Figure 6.11 shows the pair-wise comparison of four development stages with the development stage scheme colouring. In summary, remarkable detections are:

a) The initial time absorption span has been the smallest for the early development stage, and absorption initialisation varies a lot for the rest.

b) The inner side-member part with the highest IE_{max} has been declining in the maximum absorbed energy during the development.



(a)



(b) Initial time distribution for (a).

FIGURE 6.10: DEF with order scheme, ffo load-case considering eight most energetic parts for each simulation. Primary and early development stages 50 and 53 simulations, respectively, (t_n $[ms]$, Δt $[ms]$, IE_{max} $[kNmm]$).

c) The 2d visualisation overlays point clouds in initial absorption time.

d) The inner side-member stays almost steady in absorption time spread.

e) The spread of absorption time declines as the development stages evolve for the rest of the parts.

Load-case scheme

This visualisation enables the comparison of designs between load-cases, which supports detecting multi-disciplinary development challenges with different crash requirements. Figure 6.8 is a matrix scatter plot for three load-cases of the front crash in four development stages of the CEVT data with a load-case scheme, Table 4.1. It includes 611 simulations with five parts with high-ranked IE_{max}. The visualisation indicates that the ffo load-case has discontinuous absorption compared to the other two. This gap exists for t_n values that make two clusters: early ($\approx 10\ ms$) and late absorption ($\approx 60\ ms$).

6.4 Graph visualisation

Graph visualisation techniques are investigated here for knowledge discovery in simulation studies, where energy features are used as weights in these graphs. Visual exploration in an interactive way allows one to apprehend the underlying graph and thereby gain insight. Visual representations of graphs can be classified into three major categories: node-link diagrams, matrix representations, and hybrid methods. Here, the focus is on the application of node-link diagrams. Among node-link diagram methods, the most widely used are force-directed layout algorithms [Che+18]. They have often been preferred over other algorithms since the 1980s. Force-directed algorithms can be divided into classical and hybrid algorithms according to their characteristics and computational modelling. Classical force-directed algorithms are usually based

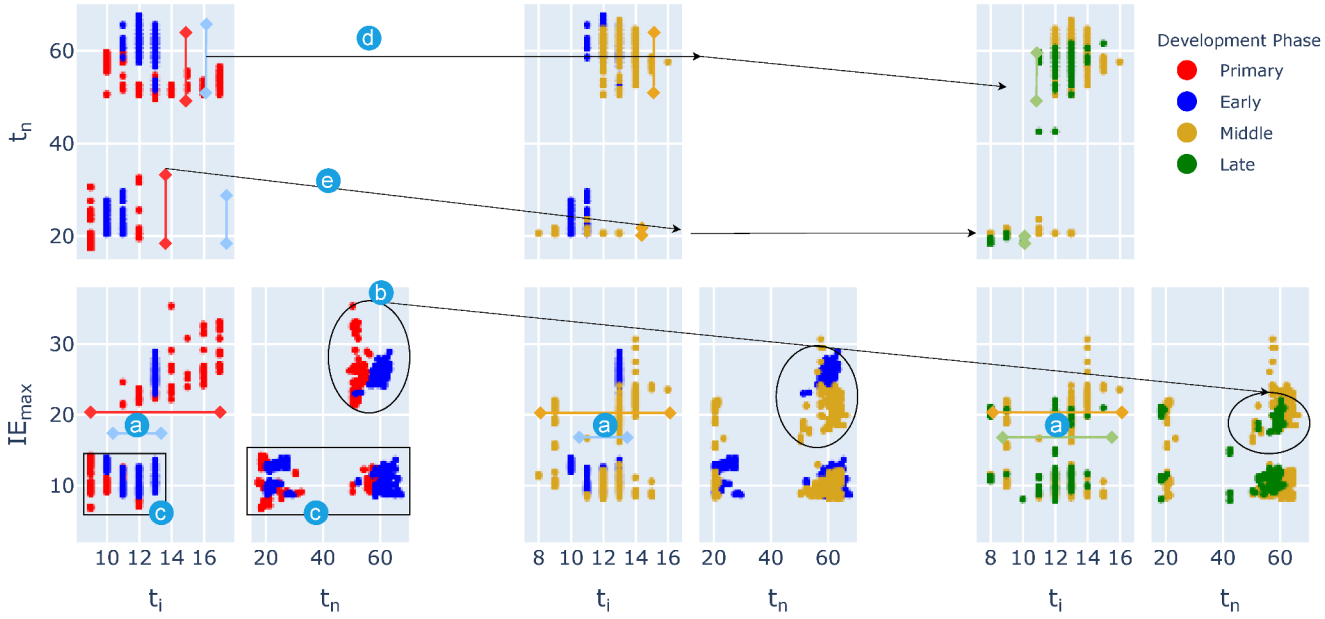


FIGURE 6.11: DEF, development stage scheme, ffo load-case CEVT data. Pair-wise comparison for four different development stages. Primary, early, middle, and late stages with 50, 53, 82, and 28 simulations, respectively. Seven most energetic parts are considered for each simulation, (t_i $[ms]$, Δt $[ms]$, IE_{max} $[kNmm]$).

on physical laws, specifically in ways that simulate a spring system. For large and complex networks, hybrid force-directed algorithms are designed, which use heuristics to improve the performance of classical force-directed algorithms [CS20]. Classical algorithms are still suitable in this case due to the relatively small size of the graph.

Following the survey [CS20], for the purposes of this work, the three methods of Fruchterman–Reingold [FR91], Kamada–Kawai [KK89] and ForceAtlas2 [Jac+14] are suitable. The three methods are investigated on the data, where ForceAtlas2 shows the best results[5]. In general, more successful force-directed techniques are those that have avoided certain principles to show off other structural properties of the graph, such as ForceAtlas2 [Jac+14]. The method still follows the idea of a physical system, but the principle the authors have tried to optimise is one of clustering rather than being concerned with edge lengths or uniform node distributions. The following is a summary of the use of such graph visualisation methods.

Here, a sub-graph is extracted by using nodes with (Sim) — SIM_DES — (Des) edges. The result is a bipartite graph consisting of two types of objects, namely (Des) and (Sim) nodes. The edges of the (Sim) — SIM_DES — (Des) bipartite graph are weighted by the Power of energy absorption (P_e), $P_e = IE/\Delta t$, which can be seen as an aggregation of energy features.

Due to the widespread energy power absorption, specifically in the networks that include outlier simulations, it is challenging to get an interpretable view of the network. Consequently, from the options available to improve the visualisation of the graphs, the gravity option is deactivated. This simplifies the equilibrium of the forces. Instead, the scaling ratio and the influence of the edge weight are investigated as two options of this method. The scaling ratio (R) refers to the repulsion required and is claimed to result in a more sparse graph. Furthermore, the

[5]ForceAtlace2 implementation available at `https://github.com/bhargavchippada/forceatlas2`.

edge weight influence (e_{inf}) scales the weights from zero, for no weight influence, to one as normal.

Note that the graph is relatively small compared to graphs in many other domains, with less than 26000 nodes considering all the node types. Consequently, the primary computational time is loading the data to the graph database, which is done offline in a pre-processing step. The ForceAtlace2 calculation depends on the number of included nodes and the needed number of iterations. For the data in this thesis, both take only a few seconds. The timings for the rest are less than a second, which makes it easy to explore the data interactively.

The visualisations presented in the following are for three scenarios:

- One load-case in one development stage.
- Different load-cases in one development stage.
- One load-case in several development stages.

All approaches mentioned in the following are practical options for an interactive user interface to assist engineers in data cleaning and knowledge discovery.

The first case study considers the eight most energetic parts for 115 simulations in a primary development stage and foI load-case. Figure 6.12a visualises the bipartite graph. The graph has 115 simulation, (Sim), and 33 design, (Des), nodes. The number of design nodes is more than eight due to differences in the most energetic parts of the simulations. This basic visualisation can only distinguish the density of the degree of (Des) nodes.

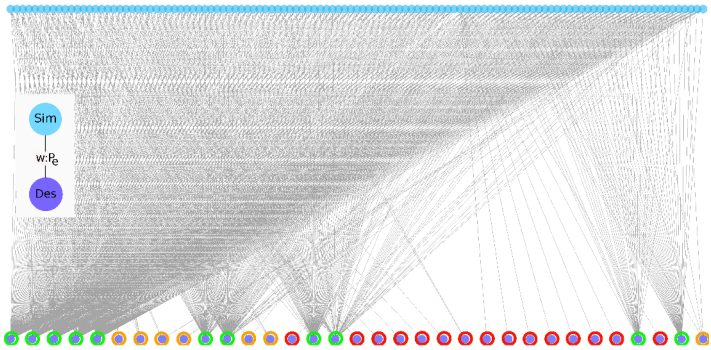
However, the ForceAtlas2 method reveals more information about this network, Figure 6.12b. By positioning them off-centred, this visualisation emphasises the outlier (Des) and (Sim) nodes. Most of the (Des) outliers are related to the connection modelling, which is very sensitive to the modelling. Therefore, the solver tries to rectify it, which causes high

IE in the corresponding connection part. However, connections are not the study object of these FE-simulations. As a result, these are unreliable simulations and designs.

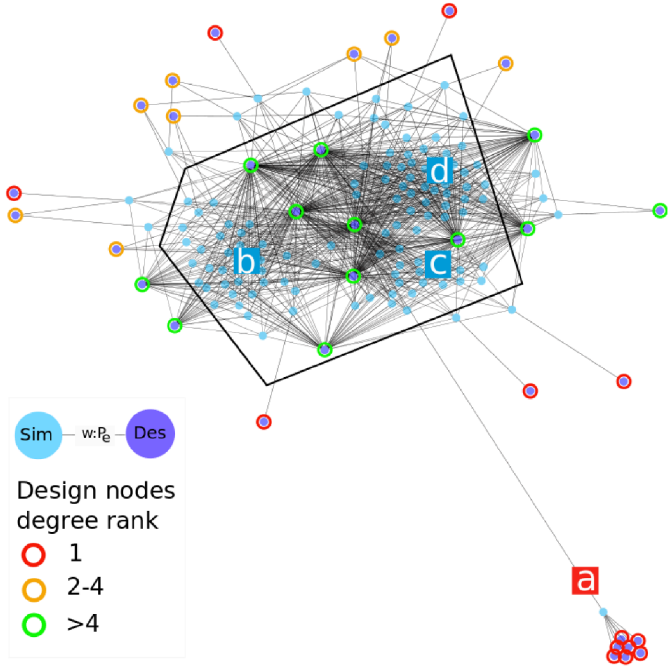
Additionally, the Sim nodes are distributed based on their similar structural connection to Des nodes, which shapes clusters of simulations, zones (b), (c), and (d) in Figure 6.12b. From a simulations clustering perspective, ForceAtlas2 has an outstanding result. The other two methods, Fruchterman–Reingold and Kamada–Kawai, separate only the outlier Des nodes. The Des nodes located in the centre of simulations are the Des nodes with the highest degree. These nodes are essential parts of most of the simulations. This visualisation highlights that Des nodes cause the split of the simulations' point cloud. In this example, each cluster has several design nodes positioned outwards and with high degrees, Figure 6.12b green nodes. Additionally, there are simulations further away from the central simulation clouds. This can indicate less explored design space, Figure 6.12b orange nodes.

The first visualisation, Figure 6.12, contains only eight designs. In addition, the edge influence is initially set to zero, allowing the structural effect of the network to be seen in Figure 6.12. Figure 6.13a shows the same simulations as Figure 6.12, but the number of designs is increased to 20 for each simulation, resulting in a total of 62 design nodes. In this figure, the remaining nodes are positioned close to the centre because the edge weights amplify the effect of outliers. An interactive setting is therefore required to improve the visualisation according to the user's needs, searching for outliers or finding clusters. The other approach is to disable the distant nodes that are outliers, Figure 6.13.

Here, we are comparing the ForceAtlas2 options outcome with removing the outliers. First, two options of e_{inf} and R are considered from ForceAtlas2 to improve the visualisation. Figure 6.14 summarises the effect of edge weight influence for the network above. This figure illustrates that by reducing the influence of edge weighting, the overall design exploration graph is preserved, and the resolution in the graph structure is still maintained. As a result, outliers and design exploration

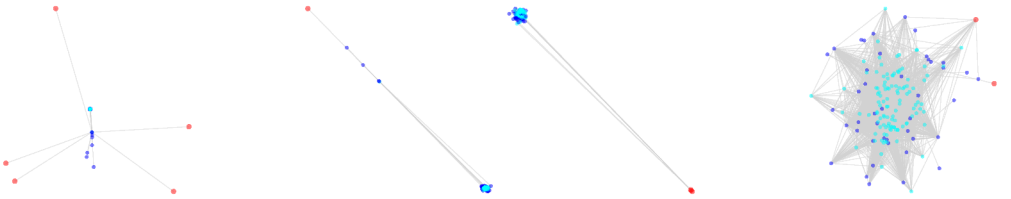


(a) Bipartite



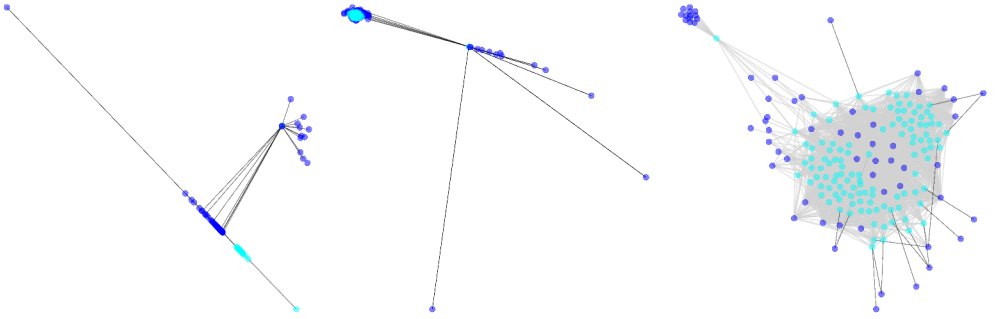
(b) ForceAtlas2

FIGURE 6.12: Information extraction with graph visualisation, fol load-case in the primary stage, 115 simulatio, (Sim), nodes with eight design, (Des), nodes for each simulation and weighted with P_e. Design nodes' edge colours are based on the nodes' degree, graph modelling described in Chapter 5.



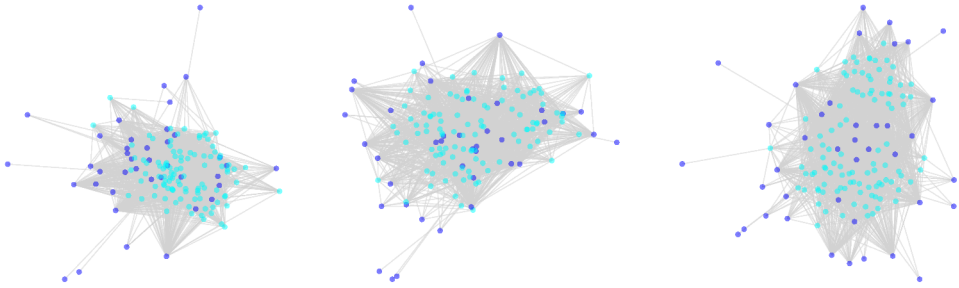
(a) itr 1 (b) itr 3 (c) itr 5 (d) itr 7

FIGURE 6.13: Case one, foI load-case in the primary development stage, the threshold of 0.8, $e_{inf} = 1$, and $R = 1$. Odd stages when iteratively removing distant nodes from the foI load-case in a primary stage, 115 simulation nodes with 20 design nodes for each simulation and weighted with P_e. The red nodes are the outliers identified to be removed.



(a) $e_{inf} = 2$ (b) $e_{inf} = 0.5$ (c) $e_{inf} = 0.02$

FIGURE 6.14: Case one, foI load-case in the primary development stage. Improving visualisation by varying edge weight influence, e_{inf}, from 2 to 0.02.



(a) $e_{inf} = 1$ (b) $e_{inf} = 0.8$ (c) $e_{inf} = 0.5$

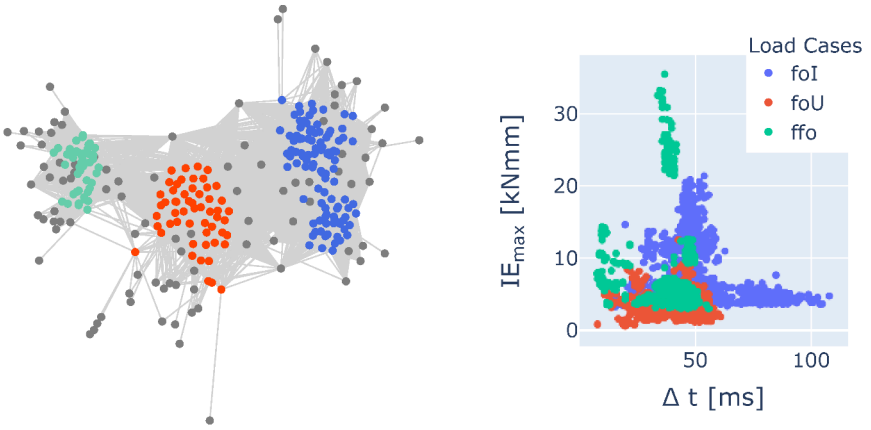
FIGURE 6.15: Improving visualisation for network without outlier simulations with varying edge weight influence, e_{inf}, from 1 to 0.5. Case one, foI load-case in the primary development stage.

clusters remain similar to Figure 6.13. Note that changing the scale factor does not seem to have a noticeable effect on the resulting visualisation of the data.

The next option is outlier removal, which uses an iterative approach to identify and remove distant nodes using the ForceAtlas2 algorithm. In each iteration, all edge lengths are calculated based on the ForceAtlas2 positioning of the nodes. Edges with lengths greater than a specified threshold are then removed. Disconnected nodes are then removed before the positions are recalculated for the reduced network in the next iteration. Figure 6.13 shows several iteration steps where the distant nodes are removed with a threshold of 0.8. The red nodes are the identified outliers and are removed before the next iteration. Thus, this method is able to quickly identify the outliers and clean the data. This method does not take into account the labelling of the nodes, so the identified nodes are either Des or Sim.

The final investigation for this case is the visualisation with ForceAtlas2, where the outliers simulations are excluded from the graph, Figure 6.15. The difference between this visualisation and Figure 6.13 is that when nodes are removed, the node labels are included, so the Sim and its

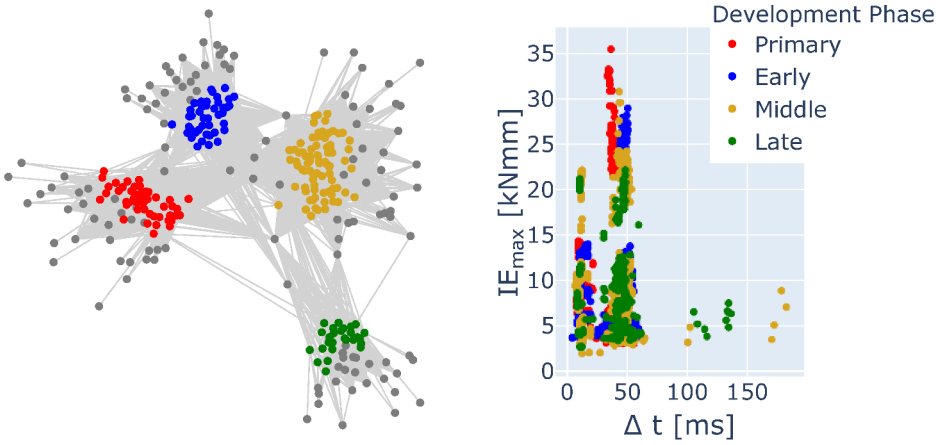
corresponding edges are removed, and so are the disconnected nodes. In this visualisation, keeping the edge influence smaller than 0.5 allows the detection of the clusters. This factor may differ depending on the intensity of the graph and its node numbers. Next, we look into two other study cases of graph visualisation.



(a) ForceAtlas2 (b) DEF

FIGURE 6.16: Case two, several load-cases in the primary development stage. (a) Des nodes are coloured in gray and Sim nodes are following the scatter plot colouring (b). $R = 1$ and $e_{inf} = 0.5$.

The example for different load-cases in one development stage includes the primary development stage of three load-case of foI, foU, and ffo, Figure 6.16. This graph contains 20 Des nodes for each Sim node, resulting in 204 simulations and 94 designs. This network excludes outlier simulations and corresponding design nodes. Figure 6.16 compares the ForceAtlas2 visualisation with the DEF. Similar to Figure 6.8, this visualisation indicates that the ffo load-case contains a limited design of experiments. An additional discovery based on comparing to the DEF is the relation of these three load-cases with each other: foI and foU have more in common than the ffo load-case. This can be visualised with several design nodes between blue and red point clouds vs. green and red.



(a) ForceAtlas2 (b) DEF

FIGURE 6.17: Case three, ffo load-case in several development stages. (a) Des nodes are coloured in gray and Sim nodes are following the scatter plot colouring (b). $R = 1$ and $e_{inf} = 0.5$.

Furthermore, the Des nodes between each load-case cluster identify the essential parts in common.

The last case study is one load-case (ffo) in several development stages, Figure 6.17. This graph also considers 20 Des for each Sim, including 196 simulations and 123 designs. Besides the scarcity of each design exploration compared to others, it is possible to discover the essential parts in common between different development stages. The ones in the centre are in common for several stages. Moreover, the late development stage, green cloud, differs more from the other three. The DEF in Figure 6.17b visualises this difference with the lowest absorption feature of that development stage. An important note is that the commonality of Des nodes between different development stages is uncertain. However, the DEF is independent of this assumption for comparing parts. This network is a potential for knowledge discovery. Note that better semantics are needed for the Des nodes than the PID used.

In the last two use cases, $e_{inf} = 0.5$ is set to visualise the clustering of the parts. However, due to the increase in the graph constraint by having several development stages included, this value needs to be higher to highlight the additional outlier. A good example is the parts in the scatter plot in Figure 6.17b whose absorption time is more than $100[ms]$. These parts are not as noticeable as in Figure 6.17a. However, increasing e_{inf} makes these parts more outstanding.

6.5 Summary

Based on graph representations of the data presented in Chapter 5, this thesis proposed energy features and used them for data visualisation while leveraging them as weights in the data graph to empower knowledge discovery. The sensitivity of energy features was presented for differentiating FE crash simulations during development stages. Moreover, it introduces a simple way of filtering the necessary parts to be studied in ML deformation-based workflows. Besides, applying ForceAtlas2 visualisation further empowered outlier detection, data cleaning, and the clustering of the parts and simulations. This visualisation allows vehicle design exploration knowledge discovery, e.g. by assessing a single load-case in one development stage and comparing different load-cases and development stages.

Overall, DEF, design ranking, and graph visualisation are three new visualisation concepts for CAE data and allow further knowledge discovery[6]. In a broader view, a web-based platform is envisioned to enable semantic reporting for CAE[7] as a practical tool, which targets CAE attribute leaders, CAE engineers, design engineers, and data analysts in automotive R&D. It should enable project members from different teams to access the CAE results, understand the design performance limitations, compare simulations, and use algorithms on the car-graph.

[6]The database example and a user tutorial are at github.com/Fraunhofer-SCAI/GAE-vehicle-safety

[7]Accessible at CAEWebVis.scai.fraunhofer.de/.

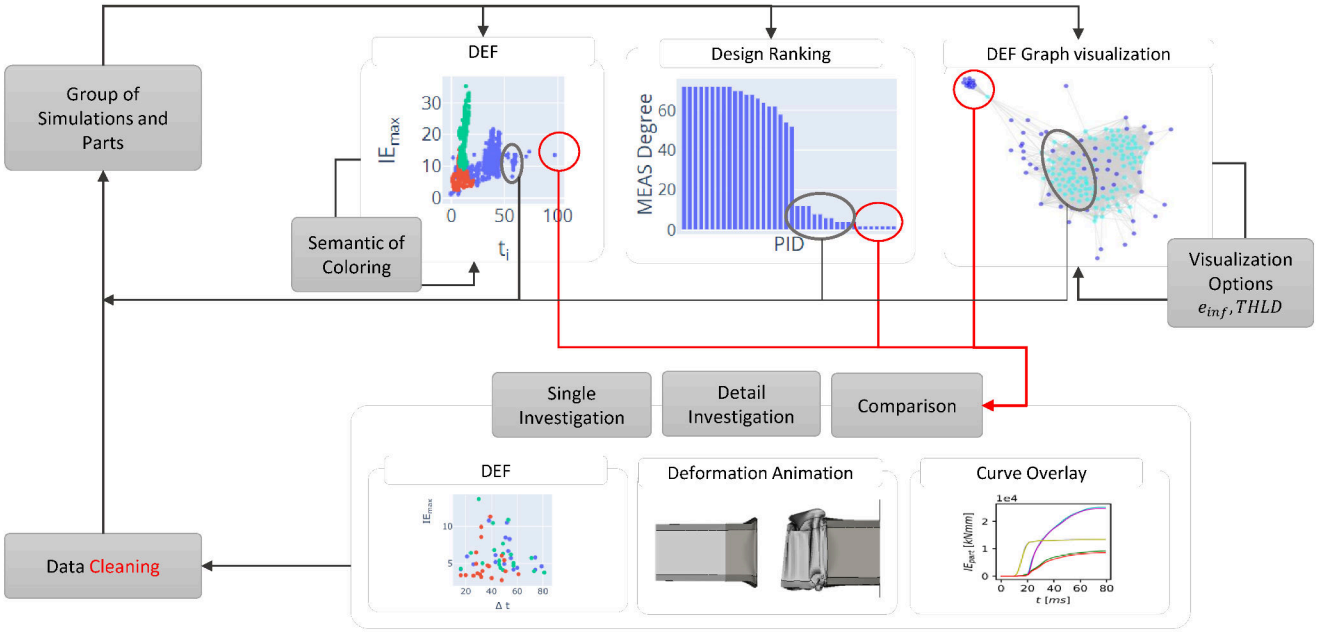


FIGURE 6.18: The workflow of the knowledge discovery assistant for dynamic reporting.

For example, the support of data exploration with 2D and 3D views of the DEF, Section 6.3.3. Interpreting DEF involves further investigation, where a dynamic interaction and filtering facilitate the data exploration. For example, each scatter point can link to the simulations or parts' corresponding energy curve, metadata, pictures, and deformation videos.

Figure 6.18 summarises the interactions of such a workflow. Here, design ranking and design exploration graph visualisation was the use case of trend and outlier detection at a high level. In comparison, the DEF can find some extreme outliers and is best used for summarising the exploration and more detailed investigations. For graph visualisation, additional improvements can still be made. One additional study can be on extending the types of nodes and relations included in the network, for example, including development tree connections, impactor/barrier nodes, and simulation similarity predictions. The edge bundling method can also reduce the visual clutter caused by edge overlaps. It can provide a global overview of complex connection graphs while providing information on the primary connection relationships in the graph by the thickness and colour of the edges [Che+19]. With all the further possibilities of feature extraction and knowledge discovery, this thesis holds back on further investigation and focuses directly on predicting the similarities of the simulations.

7 Simrank-based prediction of crash simulation similarities

7.1 Advantages of CAE data searchability

Having semantics improves searchability as a basic functionality to get more out of the data and can also be used to rank similarities between existing entities. Despite these technological developments, many engineering domains do not use semantics and graph modelling for searchability. For example, automotive OEMs now run between 10,000 and 30,000 Computer-Aided Engineering (CAE) simulations per week [Sch22]. This volume of data makes CAE in the automotive R&D process one of the engineering domains that can benefit from significantly improved searchability. The current lack of sophisticated searchability in CAE data disconnects data and hinders multidisciplinary collaboration, reducing efficient problem-solving. In addition, a significant part of the available data is not reusable. It is therefore perceived as so-called dark data [SD20]. By evaluating Graph Machine Learning (GML) methods for predicting the similarity of simulations, this chapter introduces improved searchability to this domain. Predicting the similarity of simulations will assist engineers by allowing them to search for the simulations most similar to a given one. Such similarity links different design solutions with corresponding behaviours, highlights limited explored designs, classifies explored behaviours and allows identification of outliers as these simulations are dissimilar to most others.

Previously, a Knowledge Graph (KG) for the automotive industry was

introduced, focusing on the use case of CAE crash simulations, chapters 5 and 6. This chapter uses semantics to predict the similarity of simulations. The case study is in crash simulations and uses the most energetic parts and derived internal energy features. However, a similar process can be implemented for other CAE domains by introducing simulation-based semantics that characterise the mechanical properties of the analysis, e.g. high strain elements for fatigue analysis and eigenmode structural frequencies for Noise Vibration Harshnesss (NVH) analysis. Currently, there is no physics-aware method that focuses on searchability in CAE simulations based on available information. Here, car-graph modelling is extended to include link prediction between simulations based on crash behaviour, making the simulations searchable. The car-graph is a heterogeneous weighted graph, and the relationships between simulation results are missing. Accordingly, the problem is limited to GML methods for unsupervised learning on weighted heterogeneous graphs. Note that the labelling of the data that will characterise the crash behaviour is complex; the behaviours are typically unclassified and multi-criteria.

Currently, limited methods are available for unsupervised learning on weighted heterogeneous graphs [Kum+20]. Therefore, the car-graph is reshaped to suit the best available algorithms. It is downsized to a weighted graph with two types of nodes, i.e. a bipartite graph, and allows the use of the widely used SimRank [JW02] method. This method evaluates the similarity of two types of nodes based on the connectivity of the nodes. To use the edge weights, SimRank++ [AGMC07] is also investigated, which is a SimRank extension that includes edge weights in the similarity prediction. In addition, a modification of SimRank++ is introduced that is more suitable for predicting the similarity of simulations. Note that the graph nodes of the weighted bipartite represent simulations and their main energy absorption entities. Consequently, the commonality of absorption entities between simulations controls the similarity score.

Chapter 6 introduced energy features for individual parts. Here, the

illustrative example, Chapter 4, motivates energy features such as the weights in the bipartite graph. In particular, crash simulation behaviour can be visualised in a diagram, their similarities labelled, and their behaviour classified. Furthermore, a grouping of parts is introduced, for which a method is introduced to automatically detect the components in the vehicle, taking into account the loading direction of the analysis, i.e. impact direction for crash simulation. Therefore, there are two variants of entity selection in the GML methods for similarity prediction: the Finite Element (FE) parts individually and a group of parts representing components.

The performance of predicting similarities is first examined for all SimRank-based methods on labelled crash behaviour from the illustrative example. Then, the proposed part grouping and its energy features are integrated into the initial car-graph, providing an alternative way of predicting similarities between simulations. Finally, the similarity prediction approach is explored on unlabelled industrial data from different development stages in a project of China Euro Vehicle Technology AB (CEVT).

First, there is an introduction to the SimRank methods and the extension of the method in Section 7.2. Section 7.3 presents a component detection method and shows results for the illustrative example. Next, an energy diagram is introduced that follows the crash behaviour in Section 7.4, and Section 7.5 uses these labels to evaluate the similarity predictions and rankings. There is also a summary of the results of the similarity prediction for the labelled data, the illustrative example and the unlabelled data (CEVT development stage data) in Section 7.5. Figure 7.1 summarises this chapter's approach.

7.2 Simulation similarity prediction

Identifying similar objects based on the link structure in a graph is a fundamental operation in various domains such as web mining, social

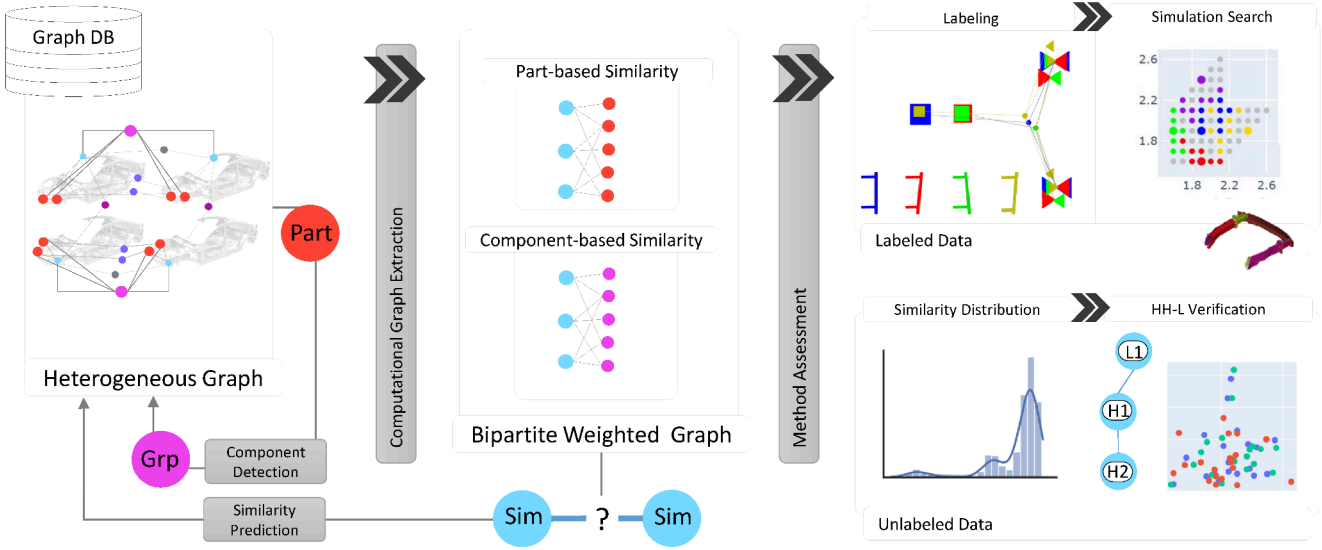


FIGURE 7.1: Similarity prediction workflow for labelled and unlabelled data. This workflow considers parts and part groups (components) and predicts the similarity based on two variants of a bipartite graph. The nodes' colouring in the heterogeneous graph reflects different node types.

network analysis, and spam detection [Wan+21]. Amid the existing similarity approaches, SimRank [JW02] has emerged as a powerful tool for assessing structural similarities between two objects. Similar to the well-known PageRank [Ber05], SimRank scores depend merely on the link structure, independent of the textual content of objects. The major difference between the two methods is the scoring mechanism. PageRank assigns an authority weight for each object, whereas SimRank assigns a similarity score between two objects.

SimRank is an approach that is applicable in any domain with object-to-object relationships. It measures the similarity of the structural context in which objects occur based on their relationships with other objects. Effectively, it computes a measure that says, "two objects are similar if they are related to similar objects" [JW02]. The similarity $s(a,b) \in [0,1]$ between objects a and b is defined by a recursive equation. If $a = b$ then $s(a,b)$ is defined to be 1, otherwise,

$$s(a,b) = \frac{C}{|E(a)||E(b)|} \sum_{i \in E(a)} \sum_{j \in E(b)} s(i,j), \tag{7.1}$$

where the set $E(a)$ contains the edges of node a, and C is a constant between 0 and 1. C gives the rate of decay, since $C < 1$, as similarity flows across edges and $C = 0.8$ [JW02]. Later, it was shown that SimRank scores are not intuitively correct for complete bipartite graphs[1] [AGMC07]. In this thesis application, a complete energy bipartite graph could be obtained, and the graph has edge weights.

To work well with complete bipartite graphs, SimRank++ was introduced, a so-called evidence-based SimRank, which additionally uses edge weights and the so-called spread to achieve similarity scores consistent with the graph's weights [AGMC07]. In particular, SimRank++ introduced the notion of evidence of similarity between nodes a and b

[1]Note that, a complete bipartite graph is a bipartite graph, where every vertex of the first node-set connects to every vertex of the second node-set.

$$evidence(a,b) := e_{a,b} := \sum_{i=1}^{|E(a)\cap E(b)|} \frac{1}{2^i} \tag{7.2}$$

as an increasing function in the number of common neighbours. Further, using normalisation and scaling according to the local variance, one obtains weights W

$$W_{a,i} = \underbrace{e^{-variance(i)}}_{spread(i)} \underbrace{\frac{w(a,i)}{\sum_{j\in E(a)} w(a,j)}}_{normalised\_weight(a,i)}, \tag{7.3}$$

where $variance(i)$ is the variance of the edge weights w of node i. All together, SimRank++ utilises the edge weights to compute similarity scores iteratively by

$$s^{++}(a,b) = e_{a,b} \cdot C \sum_{i\in E(a)} \sum_{j\in E(b)} W_{a,i} W_{b,j} s^{++}(i,j).$$

SimRank++ normalises the edges that have a common source node. A pair of nodes $(v,w) \in V \times V$ is associated with every edge $e \in E$; v is called the source of e and w is called the target of e, where V is a list of nodes and E is a list of edges in a graph. In this work, it is proposed to normalise the weights with a common target node, which is introduced as SimRankTarget++ (s^{++}_{trgt}). Depending on the physical meaning of the source and target, it matters how the weights are normalised. This modification enables the iterative method to calculate the distribution of one target for all the sources instead of all targets' distribution in one source. This is further discussed in Section 7.5.2. Consequently, Q is introduced, which normalises the edges with respect to the destination nodes

$$Q_{a,i} = \underbrace{e^{-variance(i)}}_{spread(i)} \underbrace{\frac{w(a,i)}{\sum_{j \in E(i)} w(i,j)}}_{normalised\_weight(a,i)}, \tag{7.4}$$

and using Q the s^{++}_{trgt} is defined iteratively

$$s^{++}_{trgt}(a,b) = e_{a,b} \cdot C \sum_{i \in E(a)} \sum_{j \in E(b)} Q_{a,i} Q_{b,j} s^{++}_{trgt}(i,j).$$

7.3 Component detection

FE-modelling techniques require the vehicle components to be divided into several parts, e.g. due to material and thickness differences[2]. Consequently, FE-solvers output result sets per defined part. This separation makes the post-processing of the results per part challenging. Since the connectivities of the parts are not available as a component, it is essential to develop a method for component detection to facilitate post-processing.

Components are a group of parts that are highly interdependent in their structural analysis from a CAE analysis perspective. For example, the stiffness of the side-member component (F) in Figure 7.2 depends on three reinforcement plates as well as the inner and outer side-members. One application of the use of components is in the selection of the most important parts for Machine Learning (ML) pipelines. In Chapter 6, it was shown that the use of the maximum of the internal energy is able to filter out the essential parts. However, this approach sometimes excludes smaller parts that are of interest from a crashworthiness point of view, e.g. the reinforcement plates, (l), (m), (n) in Figure 7.2. In this section, a method for component detection is proposed, its result is

[2]. For further background on crashworthiness, see, for example, [DB+04].

verified for the illustrative example, and its scalability is discussed. Finally, options for evaluating energy characteristics for components are discussed.

7.3.1 Component detection method

There are several possibilities for component detection. One option is to preserve this information from Computer-Aided Design (CAD) to CAE by introducing a corresponding workflow within the company. However, this option is, for now, not feasible due to the involved process dependencies that are time-consuming to establish in big OEMs. A second approach is to develop an interpreter for the specific FE solver that transfers each connection type to a generic connection for component buildup. Chapter 5 partially employed this method to identify connection changes in the model, which is further investigated in [Tha20]. The limitations of this approach are:

- The development is time-consuming due to the dependencies on the specific solver.
- The need to use several modelling representations for different types of connections.

Additionally, recent computational power allowed FE-models to include more details. As a result, connectivities, e.g. bolts and clips, are modelled with generic FE entities, e.g. shell or solid elements, instead of a solver-specific abstraction for connections. Therefore, developing the interpreter would be even more complicated. Moreover, both outlined approaches deliver several connections per pair of parts due to assembly requirements, e.g. several bolting or welding. The high number of connections requires additional filtering to distinguish a component's assembly from a component-to-component connection. Thus, a more automatic method is beneficial.

This thesis develops a geometrical search method that detects components. Each part in the vehicle is considered a box and then grouped

into highly overlapping boxes. The geometrical features of the parts define the box, including length, width, and height, along with the coordinate system of the FE-model (L-x, W-y, H-z). The grouping of these boxes involves the following procedural decisions

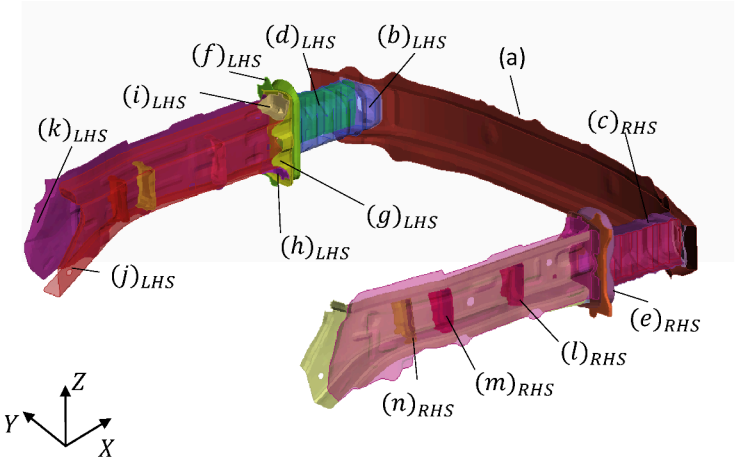
- Include specific entities from the FE-model[3].
- Define FE-modelling guidelines to differentiate parts from connections[4].
- Decide on a box merge in a pairwise comparison.
- Define batches for pairwise comparison via two-dimensional (2D) k-means clustering[5] to reduce computational time.
- Consider two stages in merging: complete and partial overlap. Complete overlap is the scenario where a smaller box (child) is located entirely in a bigger box (parent). By contrast, partial overlap refers to situations where boxes are not completely overlapped.
- Skip merges in the direction of the impact/loading for partial overlapping to capture the load path.

The investigation begins with the FE submodel presented as an illustrative example in Section 4.1. This model includes 28 parts, and 27 parts match the prescribed entity selection, Figure 7.2. From the crash analysis engineering view, this model contains eleven components: (A) bumper beam, (B) crash-boxes on Right Hand Side (RHS) and Left Hand Side (LHS), (C)(D)(E) connector plates on RHS and LHS, and (F) side-members on RHS and LHS. Thus, the intended outcome is eleven components. Connector plates between crash-boxes and side-members could be one component. However, as mentioned above, the

[3]Only shell elements since beam and solid elements usually represent the connections and have a single PID for all the same type of connection in the model.

[4]Require null shell elements for components modelled with solid elements. Null shell is a recommended method for better contact modelling, MAT_NULL in LS-DYNA.

[5]Using the implementation from sklearn.cluster.KMeans python package.



(A) bumper beam	
	(a) bumper beam
	(b) frame front cap
(B) crash-box	
	(c) crash-box inner
	(d) crash-box outer
(C) connector plate 1	(e) connector plate 1
(D) connector plate 2	(f) connector plate 2
(E) connector plate 3	
	(g) connector plate 3
	(h) reinforcement 1
	(i) reinforcement 2
(F) side-member	
	(j) side-member inner
	(k) side-member outer
	(l) side-member reinforcement 1
	(m) side-member reinforcement 2
	(n) side-member reinforcement 3

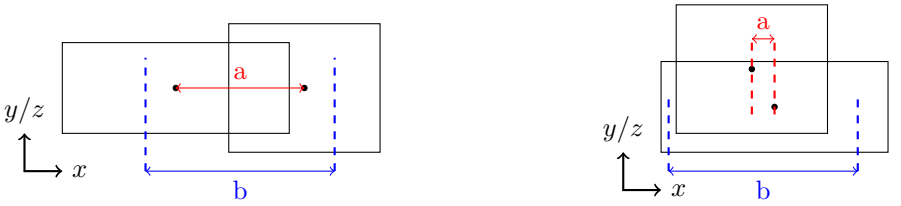
FIGURE 7.2: Grouped component for the FE sub-model, 27 parts and eleven corresponding components. In the table, uppercase letters refer to the components and lowercase letters to the parts. In the figure, only one of the symmetric parts is marked and correspondingly subscripted with either LHS or RHS.

grouping procedure prevents boxes with overlaps in the direction of impact/loading from being grouped together.

Boxes are compared in pairs for grouping boxes. Pairwise comparison of all parts for the whole FE model is computationally expensive. Therefore, k-means clustering[5] is used on the centroid of the boxes. Pairwise comparison takes 346 seconds for the complete Yaris model with 728 initial parts, and k-means clustering reduces the computation time to 23 seconds. In this way, the clustering of the boxes creates batches to reduce the number of pairwise comparisons. The distances of the boxes' centroids are taken into account in the pairwise comparison.

Here, 2D clustering is used as the three-dimensional (3D) distance of the parts clusters the parts locally and will skip some desired merges. For example, the bumper beam component in Figure 7.2 (A) will encounter this problem because centroids of the bumper beam, part (a), and the RHS and LHS frame front cap, part (b), have a significant distance in 3D space. In this example, the 3D distance will cluster the frame front cap with the crash-box, not the bumper beam. Therefore, the 2D space is considered to cluster the boxes and evaluate each one separately, top view (xy), front view (zy) and side view (xz). In this example, the bumper beam and frame front cap have an apparent distance in the top-view and front-view clusters; these boxes are not compared. However, the boxes' centroids are close in the side view and will be in the same cluster to be evaluated. As a result, there is an additional iteration loop to ensure that all the required pairs are evaluated; each loop has a switch between 2D views to change the clustering. After several iterations, all remaining parts are compared pairwise to ensure that all boxes have been compared.

The pairwise comparison investigates two scenarios: complete and partial overlap. Complete overlap is when a box is entirely inside a larger box. Here, the merged box takes over the dimensions of the larger box.



(a) Partial overlap in impact direction, x, not approved for merge.

(b) Complete overlap in impact direction, x, approved for merge.

FIGURE 7.3: Two examples of classifying the overlaps in the direction of the impact with big and small distances of $\alpha := a/b$, subfigure (a) and (b) respectively.

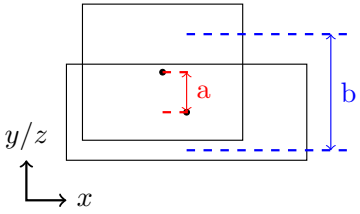


FIGURE 7.4: Decision making of box-merge in impact plane, yz.

Partial overlap uses the comparison quantity

$$\frac{\overbrace{|C_{b_1} - C_{b_2}|}^{\text{a}}}{\underbrace{(L_{C_{b_1}} + L_{C_{b_2}})/2}_{\text{b}}} \leq T, \tag{7.5}$$

i.e. the fraction of the centroids' distance and the average of the boxes' side lengths in the chosen direction. C_{b_1} and C_{b_2} are the centroids' coordinates component for the compared boxes, boxes one and two, respectively. The coordinate components are in the global axis direction: x, y, and z. The $L_{C_{b_1}}$ and $L_{C_{b_2}}$ are the box dimensions in the corresponding

axis for boxes one and two, respectively. The dimension L_c is aligned with global axis: L_x, L_y, and L_z.

The threshold T in equation 7.5 is used for two scenarios, classifying the overlaps in impact direction and deciding on overlap merging. Therefore, two different thresholds are applied for each scenario: α and β, respectively. For impact direction classification, threshold α is set to a low value, e.g. 0.01, which prevents box merging for a/b bigger than this value, Figure 7.3a. Afterwards, if the overlap is not classified in the impact direction, its percentage of overlap is evaluated to decide on boxes merging. For overlap check, the impact plane directions are assessed with the threshold β set close to one, e.g. 0.97, Figure 7.3b. Here, the boxes will merge for a large ratio of a/b, Figure 7.4. In full-frontal impact in the x plane, considered overlaps are in the y and z plane.

Algorithm 1 Box merge for impact direction x.

```
if a_x / b_x ≤ α and a_x > 0 then
    if a_y / b_y ≤ β and a_z / b_z ≤ β then
        merge boxes
    end if
end if
```

7.3.2 Component verification

The result of this method for the illustrative example corresponds to the table in Figure 7.2. Initially, there are 27 boxes representing the 27 parts, and after merging, eleven new boxes are generated containing the parts as in Figure 7.2. Here, the merging is calculated for the frontal impact, which skips the merging in the x-direction; see the coordinate system in Figure 7.2. As a result, even with the existing overlap in the x-direction, the connection plates are not merged, components C, D and E.

Afterwards, this method is implemented on the full Yaris model to assess the scalability of this method. One obstacle in the full model application is the dominancy of the exterior parts that act as a wrapper for a

big proportion of the parts, e.g. the front facia and outer layer of the vehicle body. A solution for this is to exclude the exterior parts defined by the user. An additional option is to combine part filtering with grouping as:

- Filter the most energetic parts.
- Apply the component detection with the limited search for filtered energetic parts and their neighbours.
- Update the filtering with the most energetic components.

In this way, the focus will remain on the structural parts, and exterior parts will not cause an issue.

7.3.3 Component features

Chapter 6 introduced energy features for the Internal Energy (IE) of each part with initial initial absorption time (t_i), max of IE (IE_{max}), and final absorption time (t_n). Since these features are part-based, enabling the post-processing of the results at the component level requires an additional step to combine the features. Two approaches exist to generate component-based features. First, combining the features of the grouped parts belonging to a component. Here, combining features refers to determining the minimum, maximum, sum or average of a group of features. The other alternative is summing up the parts' curves before feature extraction.

For IE curves, the outcome of these two approaches affects the time features more noticeably. For IE_{max}, the effect is minor since IE saturates after the maximum. Consequently, the sum of IE_{max} for parts and the maximum of IE summation curve differs only slightly. However, the t_i and t_n features deviate more between the two approaches. In curve summation, the early ramp-up or late saturation of the IE vanishes for the parts with the smaller energy due to the dominance of the more energetic parts. As a result, the part combined features are used for the time

features, t_i and t_n minimum and maximum, respectively, while IE_{max} is used from the summation curve.

7.4 Energy diagram

Here, an energy diagram is introduced to illustrate simulation behaviours in a crash simulation. For simplification, there is a 2D view using IE_{max} and t_n, where t_n contains the t_i feature and relates to absorption time, $\Delta t = t_n - t_i$. An additional benefit of t_n is that it is easier to understand visually than Δt for processing the sequence of behaviours, i.e. the part's relative behaviour in Chapter 6. The five most energetic parts are selected for each simulation to create the energy diagram, and the average of the energy features is added to the diagram and connected to each part. Note that considering the 28 parts included in each simulation of the illustrative example will make the visualisation challenging. The five parts turn out to be the same for all simulations, including the four plates of the crash-box and the bumper beam.

7.4.1 Diagram examples

Figure 7.5 shows the energy diagram for the base simulation. Left and right directed arrows indicate the LHS and RHS parts of the crash-box, respectively, where a square represents the bumper beam. The final energy diagram is obtained by connecting each part to a point reflecting the average of the energy features of the five parts.

Figure 7.6a displays the energy diagrams for simulations 30 and 31. These simulations have the same thickness value change but on opposite sides. As a result, the corresponding energy diagrams are essentially mirrored. Their structures look identical except for the switch between RHS/LHS, reflecting the change in producing negative or positive yaw. Figure 7.6b compares one of these mirrored simulations to the base model. It shows that the IE_{max} has decreased for RHS crash-box plates, which is due to the stiffness reduction. However, in comparison,

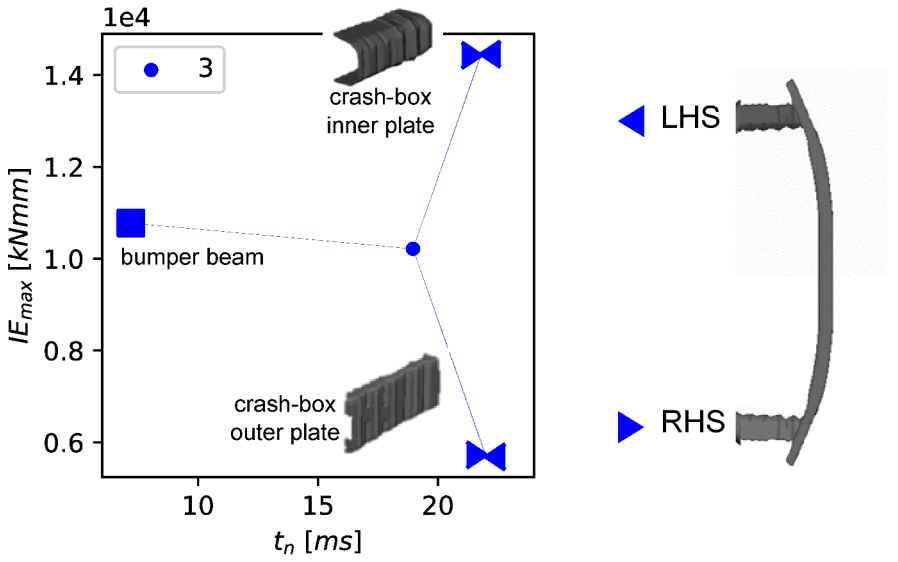
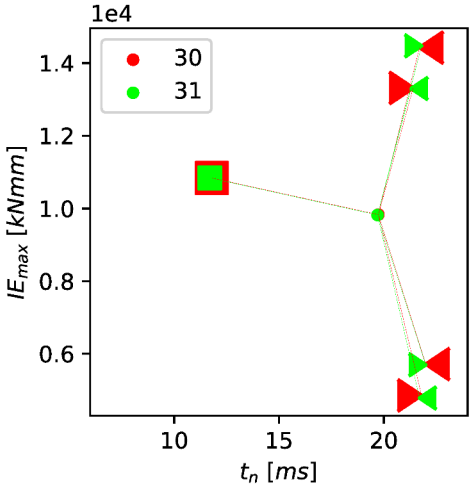


FIGURE 7.5: Example of energy diagram for the base simulation energy diagram, considering five parts, base simulation in Section 4.1.

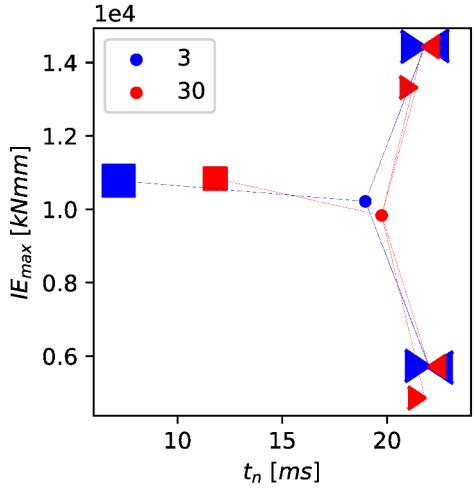
t_n has not changed. The reduction of IE_{max} while t_n is unchanged indicates a so-called stack-up state[6]. However, the average of the energy features shows lower IE_{max}. Therefore, the side-members are absorbing the remaining energy since the total IE should remain the same over all the parts in the simulation. Another noticeable observation is that the bumper beam absorption energy is independent of the crash-box; however, the t_n is dependent.

Figure 7.6c presents the energy diagrams for simulations 31 and 61, wherein both the RHS crash-box is stiffer than the LHS, resulting in the negative yaw crash mode. There is an offset in the diagrams, while the structures are similar since they reflect the similarity of the crash mode. The angle differences in each energy diagram correspond to the yaw angle. Note the offset is due to more energy absorption in simulation 61,

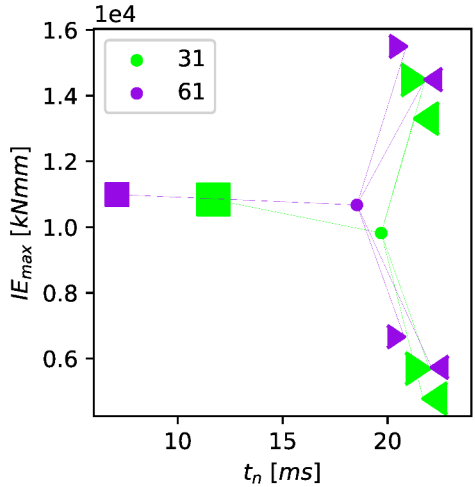
[6]Maximum possible deformation in a component.



(a) Simulations 30 and 31.



(b) Simulations 3 and 30.



(c) Simulations 31 and 61.

FIGURE 7.6: Energy graph for reference simulations, in Section 4.1. Comparing simulations 30-31, 3-30, and 31-61 in (a), (b), (c), respectively.

reflecting its higher thickness values.

7.4.2 Diagram similarities

Representing simulations as a diagram with energy features enables the comparison of simulations. The illustrative example highlights two scenarios: a change in the diagram's structure and an offset of the whole diagram. From an engineer's perspective, these two aspects can be considered as crash mode and absorption factor. First, a crash mode reflects the parts' absorption relative to each other, represented by the diagram's structure. On the other hand, one looks at how much energy is absorbed with the absorption factor. Thus, the absorption factor operates as an offset factor in the energy diagrams in this data representation. Consequently, the same crash mode but different absorption factors exist if the relative stiffness of the components is similar.

With these visual definitions of similarity between simulations using energy diagrams, the research question is: how to implement these in graph analytic methods to estimate simulation similarities? Working with unsupervised learning methods on these energy diagrams would involve treating these as separate data objects and individual weighted graphs, an ongoing open research question [Wan+20]. Instead, the energy features are examined as weights in a graph to be able to use established methods for connection prediction. In this way, the edge weights are used to detect small differences between simulations, as shown in the examples discussed in section 7.4.1.

7.5 Similarity results

This section introduces and compares different SimRank methods for predicting the similarity of the simulation. First, a brief description of the bipartite graph, both part-based and component-based, and its connection to the graph database is given in Chapter 5. Then, the SimRank

methods are evaluated and compared with the Root Mean Square Error (RMSE) of displacements and internal energy as a similarity baseline. Two approaches are used to evaluate the similarity predictions:

- Comparison of a labelled ranking, where the five reference simulations are ranked from an engineering perspective, with a ranking based on the computed similarities.
- Search for similar simulations, where the five reference simulations are searched for the most similar among the remaining 61 simulations.

Finally, the results of an industrial application of this method are presented.

7.5.1 Bipartite graph

Let us provide a brief overview of the graph modelling for both part- and component-based bipartite graphs. Figure 7.7 shows the data schema used here, which is part of the overall graph modelling introduced in Chapter 5. In this schema, a simulation (Sim) node reflects a FE-simulation outcome, where its properties stem from global entities of the simulation, e.g. total mass or impact energy. An FE-model contains many elements, where a group of elements with the same properties is identified as one part. Consequently, one simulation includes several parts, and it is modelled with a (Part) node as the main entity representing the simulation. The edge ○— NRG_PART —○ relates (Part) nodes to (Sim) nodes and includes certain energy features of the parts as weights. Design (Des) nodes bundle parts that have similar FE-model features. A group (Grp) node is a group of different (Part) nodes in the database that reflect the outcome of the component detection method.

For simulation similarity prediction, the goal is to predict the existence of an edge ○— SIM_SIM —○. Both bipartite graphs, part-based and component-based, have two node types with (Sim) and (Des); see

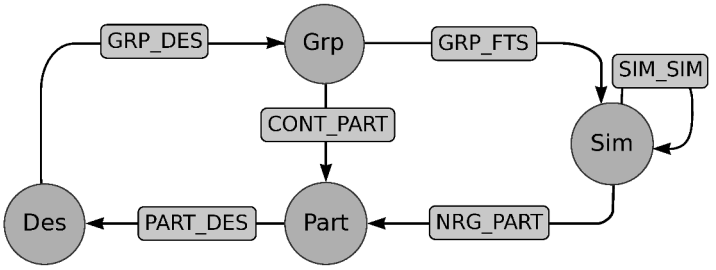


FIGURE 7.7: Graph data schema used for simulations similarity prediction, full schema available in Chapter 5.

Figure 7.8 for the part-based graph. Therefore, the similarity of connections between (Sim)-(Des) is used to predict a (Sim)-(Sim) connection. There are two scenarios for structuring the bipartite graph with (Sim)— SIM_DES —(Des). First, for part-based similarity, (Sim)— NRG_PART —(Part)— PART_DES —(Des) is used. Here, the edge weights are obtained from the energy features of the parts, and the similarity of the simulations is predicted based on the absorption similarity of the parts. Secondly, for the component-based similarity, the path (Sim)— GRP_FTS —(Grp)— GRP_DES —(Des) is taken, and the grouped features are used as weights. The similarities of the simulations are predicted based on the absorption similarities of the components.

7.5.2 Part-based similarity

For the Yaris simulations from Section 4.1, the results of different SimRank formulations with a defined labelled ranking are compared to study part-based similarity. Table 7.1 summarises the results. In this table, the columns are ordered according to the desired ranking based on engineering judgment as follows:

- Simulations (30)-(31) are the most similar due to the symmetric changes.

TABLE 7.1: Similarity prediction from different methods when considering the five most energetic parts in the illustrative example, Figure 7.8. The order of columns is the expected order as described in Section 7.5.2, ($C = 0.8$, *edge weight* $= P_e$).

Method	30-31	3-30	3-31	61-31	61-30	3-61
s	0.4444	0.4444	0.4444	0.4444	0.4444	0.4444
s_w	0.4749	0.4795	0.4798	0.4789	0.4783	0.4892
	6	*3*	*2*	*4*	*5*	*1*
$s_{w,evd}$	0.4379	0.4427	0.4430	0.4420	0.4414	0.4527
	6	*3*	*2*	*4*	*5*	*1*
s^{++}	0.2084	0.2104	0.2102	0.2267	0.2260	0.2295
	6	*4*	*5*	*2*	*3*	*1*
s^{++}_{trgt}	0.3119	0.2719	0.2717	0.2695	0.2695	0.2399
	1	*2*	*3*	*4*	*5*	*6*

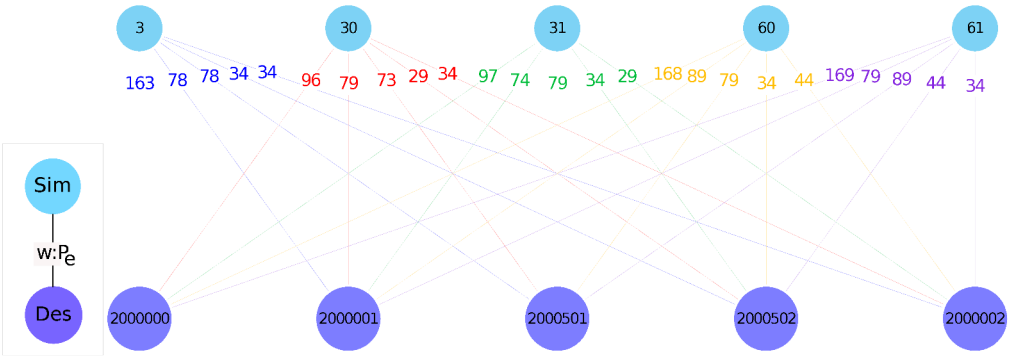


FIGURE 7.8: Bipartite graph for illustrative example with P_e as edge weight, shown values are in $[10MNm/s]$. The edge colour follows reference simulations in Section 4.1.

- Simulations (3)-(61) are the least similar since there is the most significant stiffness change among all simulations.
- The pairwise similarity of (30) or (31) to simulations (3) and (61) should be equal. Equality comes from symmetrical behaviour that acts as a mirrored weight on nodes.
- Simulations (30) and (31) are more similar to (3) than (61) since the stiffness difference is less in (3)-(31)/(30) compared to (61)-(31)/(30). As a result, simulations (3)-(30) and (3)-(31) have the second-order ranking with equal values, and simulations (61)-(31) and (30)-(31) have the third-order ranking.

The used methods include SimRank (s), weighted SimRank (s_w), SimRank with evidence ($s_{w,evd}$), weighted SimRank with evidence and spread (s^{++}), and s^{++}_{trgt}. The Power of energy absorption (P_e), $P_e = IE_{max}/\Delta t$, of the parts is used as the weight for the ○— SIM_DES —○ edges. Another alternative for the edge weight is the IE_{max}; however, P_e gives better results. In the study, the five most energetic parts are selected, which are the bumper beam and two plates of the crash-box on

TABLE 7.2: Part names for PID in Figure 7.8.

Property ID (PID)		Part Name
2000000		bumper beam
LHS	RHS	
2000001	2000501	crash-box inner plate
2000002	2000502	crash-box outer plate

LHS and RHS, Table 7.2.

Table 7.1 presents the (Sim)-(Sim) similarity predictions[7] for the illustrative example. Figure 7.8 shows the five most energetic parts for the five reference simulations. This results in a fully bipartite graph, and disregarding weights means that two simulations are similar if the energetic parts are similar. Therefore, as expected from Section 7.2, SimRank predicts that all simulations are similar, which shows that the method is insufficient to evaluate the similarity between simulations with similar energetic parts but different absorption distributions.

With the P_e weight, the predicted similarities still differ from the expectations for s_w, $s_{w,evd}$ and s^{++}. However, the s^{++}_{trgt} method provides a result that reflects the labelling. Note that to observe an effect using the weight factors, they need to be scaled to be smaller than 2 (P_e scaled with $10\,e8$ based on this model unit system, energy $[N-mm]$ and time $[s]$). If the spread is more expansive than two, then all similarities become zero, and if it is smaller than one, the result is similar to the weighted graph without spread.

It is interesting to further discuss the difference of s^{++}_{trgt} and s^{++} in this use case. Considering s^{++} and normalising the edges regarding the

[7]The SimRank similarity calculation in the NetworkX Python package is modified to evidence-based SimRank with spread consideration.

sum of the edges in the source nodes refers to normalising each part absorption with the total IE in a model, which is more or less constant for all the simulations when considering one load-case. However, normalising for edges in the target nodes, The edge weight is normalised with the total absorbed energy for that specific part in all simulations, which highlights the absorption efficiency of that part for each simulation. Consequently, the second approach is more relevant for comparing simulations as the parts are weighted relative to each other, rather than the first approach, which considers the relativity of the parts in a simulation.

Rankings based on similarity or distance measures are compared for simulations to further investigate the performance of the proposed algorithm. A common approach to assess differences in the simulations is by looking at mesh-based function differences, e.g. [ITG19; Gar+22]. Therefore, the differences in displacement between the simulations are used, and the RMSE is evaluated as a measure of the distances between the simulations. Table 7.3 summarises the RMSE of the displacements and the corresponding ranking for the illustrative example in three scenarios: all parts at the last time step (t_{max}), all parts in all the time steps (t_{all}), and the five most energetic parts in the t_{max}. Table 7.3 shows that none of the three approaches can capture the expected crash behaviour in the order given above. The top three and the last three similarities are invariant. However, there is a different order within each cluster. Differences in the ranking show that this method is time-step dependent. Additionally, parts meshes should be the same to be able to evaluate the RMSE. Further, looking at all time steps is computationally expensive, and the time sequence of events can lead to high differences while the final crash mode is similar, e.g. stack-up situations. While looking at only the last step would solve this issue, the sequences of events will still be missing in the similarity calculation.

Another approach is to evaluate the RMSE for the internal energy of the parts with more global features than displacement, the last row in Table 7.3. Here, for simulation pairs, P_e are compared for the five most

TABLE 7.3: RMSE for the difference of displacement ($Disp$ $[mm]$) and energy power absorption (P_e $[MNm/s]$) in each pair of simulations, considering a different number of parts and time steps in the illustrative example. The even rows show the ranking of the distances. The order of columns is the expected order described in Section 7.5.2.

Scalar			30-31	3-30	3-31	61-31	61-30	3-61
	t_{max}	All*	5.55	7.6	7.8	21.2	22.0	15.3
			1	*2*	*3*	*5*	*6*	*4*
Displ	t_{all}	All	7.97	7.0	7.1	17.4	20.5	13.8
			3	*1*	*2*	*5*	*6*	*4*
	t_{max}	5**	5.9	4.7	4.7	9.1	9.8	6.0
			3	*2*	*1*	*5*	*6*	*4*
P_e	t_n	All	4.5	299.1	295.1	327.5	331.3	69.5
			1	*4*	*3*	*5*	*6*	*2*

* All: All the parts.

* 5: Five most energetic parts.

energetic parts. The main difference is the ranking of (3)-(61) as the second.

So far, we have investigated different configurations of the SimRank++ method for similarity prediction between simulations. An additional hyperparameter to evaluate is the number of employed parts from each simulation that are used in the bipartite graph. Table 7.4 summarises s^{++}_{trgt} prediction for 2, 5, 15, and 28 (all) parts being considered. The order of the predicted similarity between simulations has the expected pattern for the labelled data upwards from including five parts. Noteworthy, the similarity score spread declines when including more parts.

7.5.3 Component-based similarity

Initially, a bipartite graph is constructed based on the FE parts to predict the similarity of simulations. We now consider similarity predictions based on the components, using the result of the component detection presented in Section 7.3. Table 7.5 summarises the prediction result of s^{++}_{trgt} for the component-based bipartite graph while increasing the number of most energetic components included. This evaluation requires at least three components to fulfil the previous section's pre-defined ranking. Parts included in these three components are similar to the five parts in similarity prediction of Table 7.4. However, using components, the predicted similarities have higher values.

Additionally, for the component-based similarity, the maximum range of the spread is achieved by including five components, 15 parts, whereas for part-based similarity, it is with five parts. Note that the 15 parts involved in the component-based similarity are not the same as the 15 parts involved in the part-based similarity. Six of the 15 parts are reinforcement plates of the side-members when filtering with components and are not selected as the most energetic parts when using parts. Consequently, component-based similarity performs adequately with a more stable result; however, part-based similarity is more sensitive.

TABLE 7.4: Part-based s^{++}_{trgt} similarity prediction deviation regarding changing the number of energetic parts included in the illustrative example, Figure 7.8, ($C = 0.8$, *edge weight* $= P_e$).

No. Parts	30-31	3-30	3-31	61-31	61-30	3-61	Range
2	0.424	0.424	0.424	0.424	0.423	0.423	0.007
5	0.312	0.272	0.272	0.270	0.270	0.240	0.072
15	0.276	0.257	0.257	0.252	0.252	0.239	0.037
28	0.309	0.298	0.297	0.295	0.295	0.290	0.018

TABLE 7.5: Component-based s^{++}_{trgt} similarity prediction deviation regarding changing the number of energetic components included in the illustrative example ($C = 0.8$, *edge weight* $= P_e$).

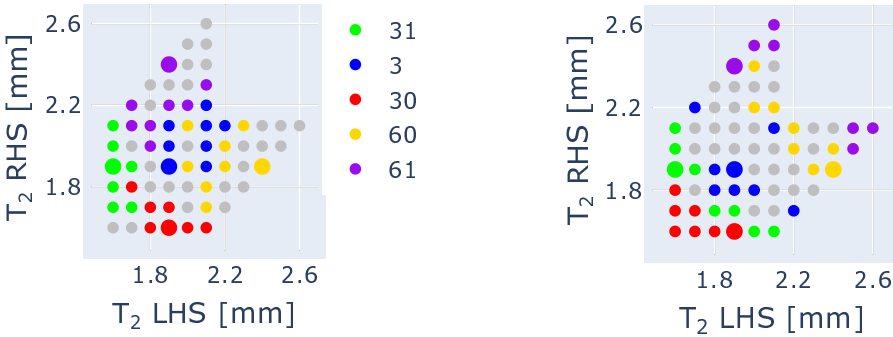
No. Parts	30-31	3-30	3-31	61-31	61-30	3-61	Range
2	0.412	0.421	0.421	0.418	0.417	0.418	0.005
3	0.413	0.407	0.407	0.401	0.400	0.397	0.016
5	0.244	0.226	0.226	0.200	0.199	0.186	0.058
11	0.252	0.239	0.240	0.227	0.226	0.217	0.035

Moreover, the similarity range drops less with increasing the number of components than with the number of parts. Comparing the full model prediction, 28 parts for the part-based compared to eleven components, the component-based has a broader range than the part-based. Overall, component-based similarity shows better results in this use case.

7.5.4 Searching simulations

In this section, the similarity prediction methods are used to search for simulations that are similar to the five reference simulations. First, the capabilities of the s^{++}_{trgt} and RMSE of P_e methods are compared as a search tool. Figure 7.9 visualises the corresponding top seven similar simulations for each of the reference simulations from Section 4.1. The diagonal points, $x = y$, are expected to be similar to those in simulation three with the zero modes. This is because the points have the same thickness ratio between the crash-box plates. Likewise, the simulations under $x = y$ should be similar to simulations 30 or 60 with mode $+v_z$ based on their stiffness range, while the ones above should have mode $-v_z$. The ones with mode $-v_z$ should be similar to simulations 31 or 61. Figure 7.9a shows that the s^{++}_{trgt} method achieves the expected clustering, e.g. the modes stay on one side of the diagonal. On the other hand, Figure 7.9b shows that the RMSE of P_e method fails to recognise the crash modes as the coloured points for simulations 30, 31, 60, and 61 are on both sides of the diagonal. This result shows that this method primarily works by averaging the parts' energy absorption.

The next step is to compare part-based and component-based results. Figure 7.10 visualises the comparison of the two methods while increasing the number of target nodes in the bipartite graph, (Part) and (Grp) for part-based and component-based methods, respectively. This comparison also highlights the deviation between the two methods by increasing the number of parts. Additionally, the colour coding for the zero mode deformation, blue scatter points, is captured the best for the minimum included target nodes, Figure 7.10a, 7.10e. This observation



(a) s^{++}_{trgt}.

(b) RMSE of P_e.

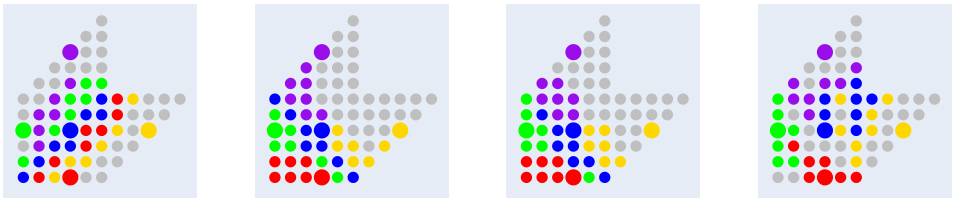
FIGURE 7.9: Most similar simulations to the reference simulations according to two similarity estimations. The used colour code of simulations is from Section 4.1.

emphasises that finding a similar simulation differs from ordering the similarities. For finding the most similar simulation, including the least number of target nodes or part-based assessment, perform satisfactorily. However, from the robustness aspect and the ranking of the prediction, the component-based is more promising.

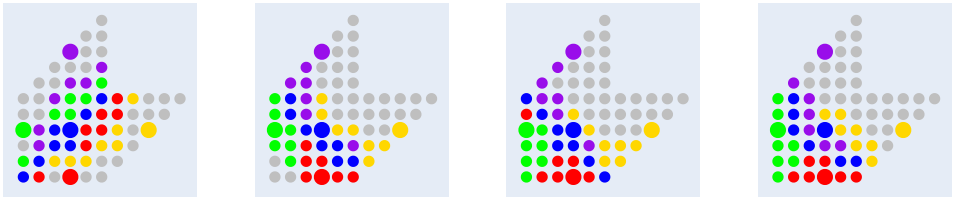
7.5.5 Industrial application

After examining the SimRank++ method for the illustrative example, the approach is applied to data from CEVT using the best-performing configuration so far, i.e. SimRankTarget++. This data includes a total of 611 simulations of three different load-cases from frontal impact (ffo: full front overload, foU: front oblique overlap, and foI: front small overlap) in four successive development stages (primary, early, middle, and late), the detailed data description in Section 4.3 in Chapter 6.

For this data, the bipartite graph is part-based with energy power as the weight factor, $P_e = IE_{max}/\Delta t$. The similarity prediction considers each load-case for each development stage separately for a specified number of parts, i.e. 20. The load-case separation of simulations is due to the use



(a) 2 Parts. (b) 5 Parts. (c) 9 Parts. (d) 28 Parts.



(e) 2 Grps. (f) 3 Grps. (g) 5 Grps. (h) 11 Grps.

FIGURE 7.10: Comparing part-based labelling (a)(b)(c)(d) with component-based (e)(f)(g)(h) while varying the number of Part and Grp nodes respectively. The x-axis, y-axis, and colouring are the same as in Figure 7.9.

case that similarity between different load-cases is usually out of interest. Furthermore, the grouping of development stages is due to Property ID (PID) changes between development stages to avoid connecting two irrelevant parts. The necessary parts mainly vary between load-cases and slightly among developmental stages.

An overview of these results and a deep dive into the results for a single load-case in a single stage of development are presented here. The industrial data is unlabelled, so it is challenging to assess the result of the similarity predictions. In order to tackle this problem, two approaches are presented. First, the result of the similarity prediction is visualised using a histogram and a Kernel Density Estimation (KDE)[8]. Second, specific simulation pairs are selected in each batch to further analyse the similarity prediction, H-LL simulations. These approaches are presented for the foI load-case in the primary development stage.

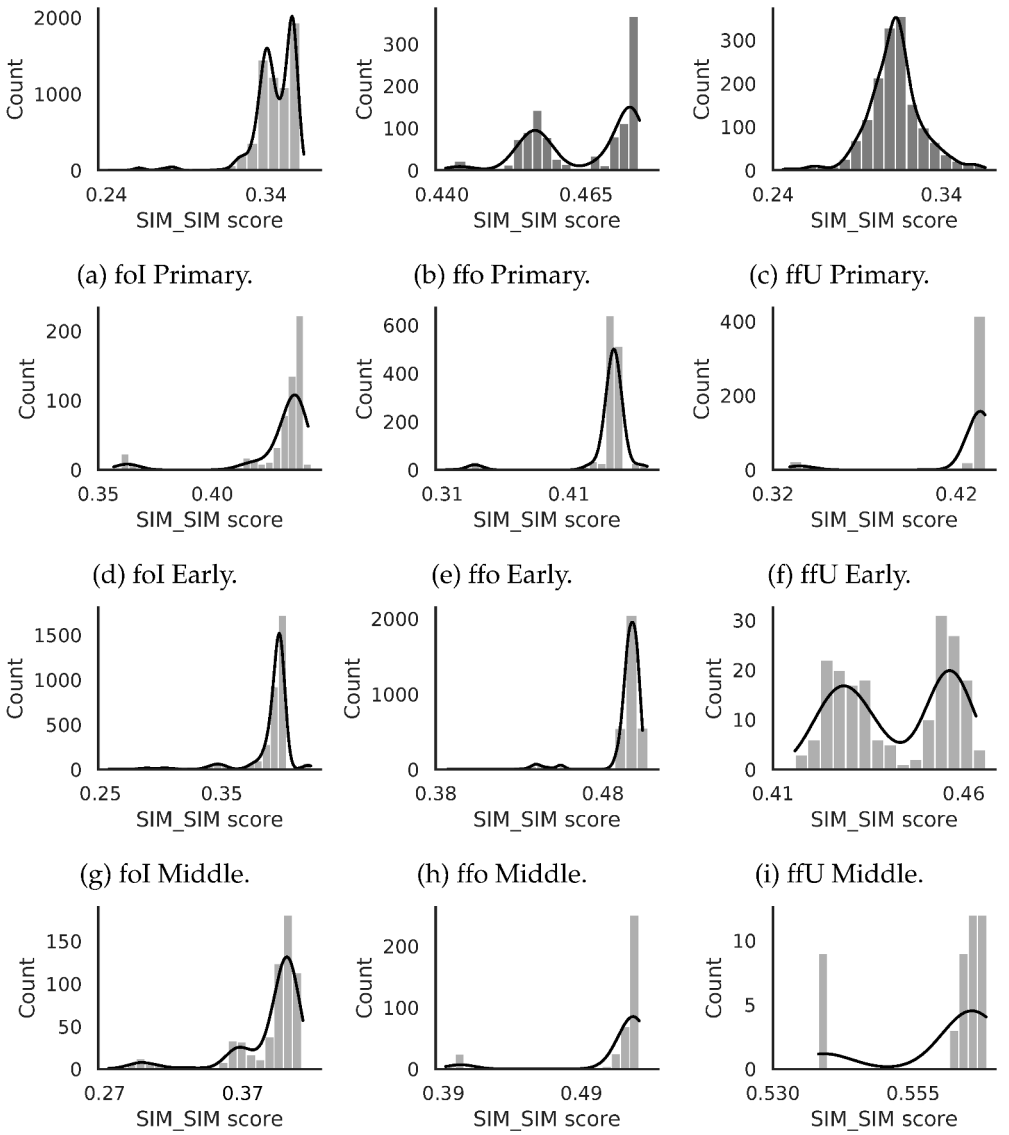
[8]The seaborn.distplot Python package with KDE=True is used.

Similarity density

The similarity prediction score depends on the selection of simulations in the batch and the number of included designs. Figure 7.11 presents the similarity distribution for different load-cases in different development stages. Here, each load-case is the subject of analysis in separate stages of development. Each calculation excludes simulations with a similarity score of less than 0.2. The KDE plot visualises the probability of data being in a given range in the area under the density curve. The KDE plot's range on the horizontal axis refers to the predicted similarity score, whereas the vertical axis reflects the number of simulation pairs for each value. These KDE plots highlight the differences between each batch, e.g. the score range and the number of peaks. Here, 0.2 seems low for some batches to disregard the outliers, e.g. Figure 7.11(a) and 7.11(g). Nonetheless, SimRankTargt++ low computational cost, less than a second, allows users to run the computation interactively with different filtering.

In particular cases, the scores spread is small between simulations, figures 7.11(b) and 7.11(l). The narrow range can highlight limited exploration of the designs. Moreover, the singular high density of similarity prediction is related to the number of parts included in the similarity calculation, figures 7.11(g) and 7.11(k). Like in the illustrative example, a fully connected bipartite graph has tighter prediction scores, and the effect of the weights is not as strong as the structure. For example, in a group of FE-simulations, a fully bipartite graph means all 20 most energetic parts are the same for all the FE-simulations; however, there is a difference between them due to the difference in energy distribution. One approach to extend the deviation of the link prediction score is to include fewer parts to avoid having a fully bipartite graph.

Further, the focus is on similarity predictions for the foI load-case in the primary development stage when considering the 20 most energetic parts, Figure 7.12. The total number of simulations is 115; consequently, the number of similarities pairs is 6555. A noticeable outcome is that the density of similarity prediction shapes clusters of simulations. In this



(a) foI Primary. (b) ffo Primary. (c) ffU Primary.

(d) foI Early. (e) ffo Early. (f) ffU Early.

(g) foI Middle. (h) ffo Middle. (i) ffU Middle.

(j) foI Late. (k) ffo Late. (l) ffU Late.

FIGURE 7.11: KDE plots for s^{++}_{trgt} prediction with 20 most energetic parts in three different load-cases: full front overload (ffo), front oblique overlap (foI), and front small overlap (ffU) in four development stages: primary, early, middle and late that reflects the sequence of the stage, in Section 4.3 of Chapter 6.

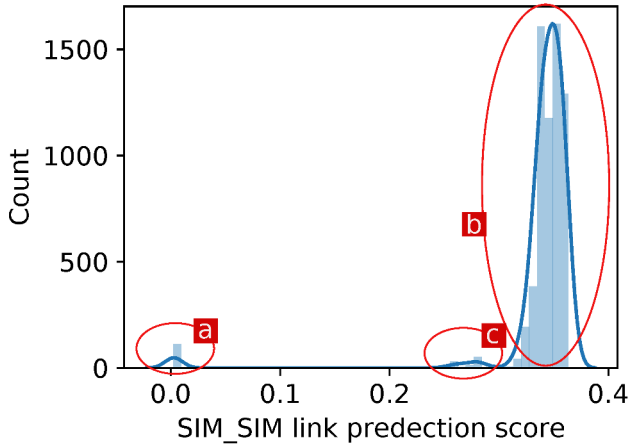


FIGURE 7.12: The s^{++}_{trgt} Link prediction histogram and estimated density for the foI load-case in the primary development stage, CEVT data. The (a), (b) and (c) peaks highlight the three clusters of similarity distributions.

way, density clustering detects the groups of simulations with similar scores of similarities. Figure 7.12 has three clusters at (a), (b), and (c), where (a) includes mainly the outliers.

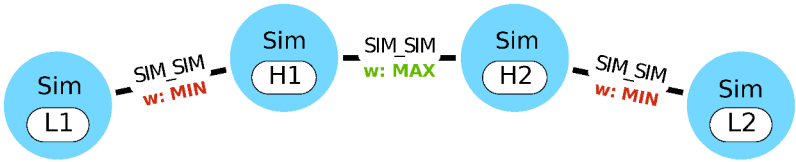


FIGURE 7.13: H-LL simulation schema selected for each load-case in a development stage based on similarity prediction score.

Furthermore, the cluster with the lowest prediction values includes simulations that reflect outliers, which can be used for anomaly detection. Simulation analyses with early termination due to errors or unrealistically high internal energy are manually identified for the considered foI

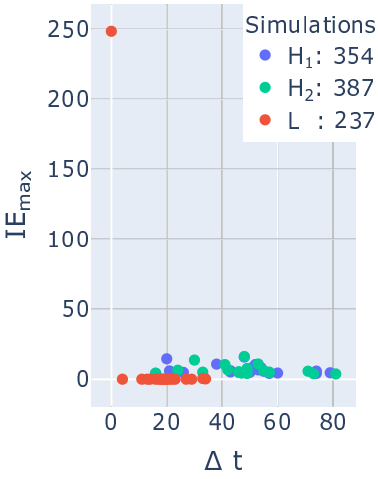
load-case during the primary stage. The similarity scores with and without these simulations are then evaluated. The results show that the low-range similarity cluster, i.e. zone (a) in Figure 7.12, is removed from the density plot when excluding the outlier simulations. The corresponding similarity distribution is plotted in Figure 7.11(a).

H-LL simulations

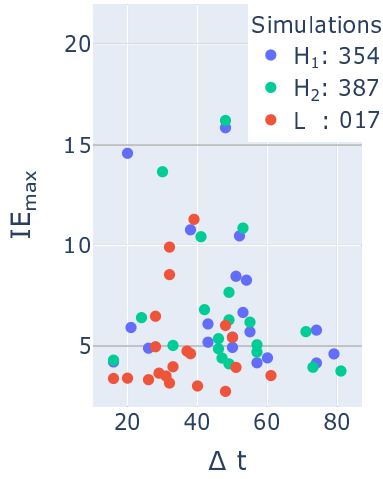
One way to check the similarity distances in detail is to compare the energy features of several simulations. For that, H-LL edges are selected, where H is the edge with the highest similarity; that is, the nodes H_1 and H_2, connected by edge H, are the most similar pair of simulations. The edge is according to the maximum predicted score of ○— SIM_SIM —○. In a second step, the corresponding least similar simulations are selected, that is, both H_1-L_1 and H_2-L_2, Figure 7.13. In some cases, L_1 and L_2 may be the same. These are called H-LL simulations, and studying the results of these simulations in detail helps to assess the reliability of the predicted similarity.

Table 7.6 shows a summary of link prediction for the foI load-case during the primary stage. Table 7.6a takes into account all simulations. Each table row shows the predicted similarity of H-LL simulations while increasing the number of parts. For this data set, if fewer than six of the most energetic parts are considered, the graph will be a disconnected graph. An additional observation is the HL similarity drop after including at least 15 parts in the analysis. Accordingly, the 15 most energetic parts will have the highest similarity range, and afterwards, the H-LL pairs are stable.

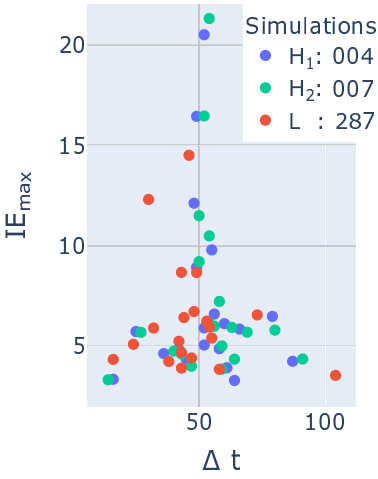
In a further step, the outlier simulations are removed from the cluster (a) in Figure 7.12. The H-LL simulations are identified while increasing the number of parts included in the bipartite graph, Table 7.6b. Increasing the number of parts keeps the similarities range shrinking constantly.



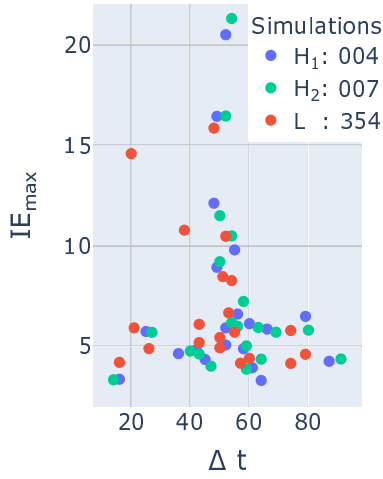
(a) H-LL$_1$



(b) H-LL$_2$



(c) H-LL$_3$



(d) H-LL$_4$

FIGURE 7.14: Analysis of s^{++}_{trgt} prediction, energy features for 20 most energetic parts in H-LL simulations for primary development stage and FoI load-case. In all the plots, blue-green markers refer to high similarity pairs of simulations, and red is the low similarity. (Δt $[ms]$, IE_{max} $[kNmm]$).

Generally, parts selection for the similarity assessment is a hyperparameter for the user; however, the H-LL similarities support revealing the differences.

In addition, the energy features of a number of H-LL simulations are compared in a 2D plot. With the so-called energy scatter plot in Chapter 6, one compares the energy features of the most energetic parts of the simulations, which enables visually assessing the similarity of the simulations' parts in energy absorption. Let us start with the H-LL$_1$, (354)-(387)-(237), from table 7.6a, the s^{++}_{trgt} that predicted simulation (237) as least similar. Figure 7.14a visualises the energy features of 20 parts for each of these three simulations. The plot indicates that simulation (237) has enormous energy in one part, and the simulation ended earlier, causing it to differ noticeably in comparison with the other two simulations. The least similar simulation is expected to be an outlier, cluster (a) in Figure 7.12, and this comparison confirms it.

Next, Table 7.6b summarises the H-LL simulations without the outlier simulations. Here, clusters (b) and (c) are expected in the KDE plot, Figure 7.12. An initial remark is that the minimum number of required parts in the dataset that avoids having a disconnected graph decreases from six to two, comparing Table 7.6a and 7.6b respectively. This drop is an additional cross-check for verifying the filtering of the outliers. Besides, it emphasises that filtered simulations have the two most energetic parts in common, but there is a flip in their order. The flip does not make it possible to reduce it to a single part and still have a connected graph. Consequently, there is a bifurcation in the simulations' behaviour.

In Table 7.6b, increasing the number of parts does not settle in one H-LL simulations group, whereas in Table 7.6a occurs after 15 parts. However, in Table 7.6b, a trend is observed in H-LL simulations. Thses groups are marked in Table 7.6b as H-LL$_2$, H-LL$_3$, H-LL$_4$ that includes simulations

354-387-17, 4-7-287, and 4-7-354 respectively. The first and second sets, H-LL$_2$ and H-LL$_3$, do not overlap in simulations; however, H-LL$_4$ contains HH simulations of H-LL$_2$ and H-LL$_3$. This observation underlines that two behaviour trends are dominant and have a detectable distance from each other.

We now look at the energy function scatter plots of these simulations to verify the assumptions. Figures 7.14b, 7.14c, and 7.14d plot energy scatterplats of these simulations for H-LL$_2$, H-LL$_3$, and H-LL$_4$, respectively. In Figures 7.14b and 7.14c, we see that parts of HH simulations are nearby (blue and green markers), whereas L simulation points are further away (red marker). Furthermore, Figure 7.14d proves the distance between HH of H-LL$_2$ and H-LL$_3$. Here, we can see that the difference is not as noticeable as H-LL$_2$ and H-LL$_3$; parts with lower energy become dominant in similarity prediction as the number of parts increases.

The KDE and H-LL were used with varying numbers of parts as representations to verify the similarity scores in the industrial application. For the practical CAE process, a discovery platform is recommended where the user can actively select the input simulations from the KDE distribution and validate this cluster using the H-LL simulations. For now, the novelty of predicting the similarity of simulations and the lack of labelled data requires these types of visualisation to assess the similarity results. These visualisations can facilitate the integration of the method into the CAE workflow, which enables the collection of feedback from engineers. The integration of these methods into CAE processes is a critical requirement to enable future improvements using graph analytics.

7.6 Summary

Today, the searchability of the web is an obvious benefit for everyone. However, enhanced searchability still needs to be realised in the CAE domain, where its advantages need to be demonstrated to the engineers. For example, it enables multi-disciplinary collaboration by easing the

TABLE 7.6: The H-LL simulations similarities, s^{++}_{trgt} $C = 0.95$, score with varying number of parts included. For all the foI load-case simulations in the primary stage, CEVT data.

No. Parts	H_1	H_2	L_1	L_2	H_1H_2	HL_1	HL_2
6	008	018	386	386	0.486	0.295	0.291
10	004	007	090	090	0.438	0.287	0.294
14	353	354	090	090	0.409	0.266	0.253
16	354	387	237	237	0.388	0.002	0.002
***20**	354	387	237	237	0.364	0.003	0.003

* H-LL_1, Figure 7.14a

(a) All simulations, cluster (a), (b), and (c) in Figure 7.12.

No. Parts	H_1	H_2	L_1	L_2	H_1H_2	HL_1	HL_2
6	135	190	287	287	0.492	0.304	0.307
10	354	357	028	017	0.453	0.340	0.358
***14**	004	007	354	354	0.427	0.375	0.376
16	354	387	017	021	0.405	0.344	0.350
18	004	007	287	287	0.389	0.349	0.349
*20	354	387	017	017	0.377	0.321	0.331

* H-LL_2, Figure 7.14b *** H-LL_4, Figure 7.14d

** H-LL_3, Figure 7.14c

(b) Excluding outliers, clusters (b) and (c) in Figure 7.12.

finding of data within the team and across the company, which increases efficient problem-solving. Moreover, it allows different interconnecting solutions for the same problem and highlights unexplored solutions. In the context of the searchability of crash simulations, the focus was on predicting the similarity of simulations and ranking them accordingly. Regardless, enhanced searchability will also bring benefit to other CAE domains and other semantics of a domain, e.g. design features, cause-and-effect analyses, modelling techniques, discipline requirements, and project decisions.

The graph modelling results in a heterogeneous graph, and the problem is considered unsupervised learning. The only method found suitable for this scenario, a bipartite weighted graph, was extracted in order to be able to use SimRank-style methods. Additionally, an alternative weight normalisation for SimRank++ was introduced. The proposed normalisation is based on target nodes instead of source nodes. The results showed that this works better for the addressed application of predicting the similarity of crash simulations and the corresponding ranking of simulations, shown on a constructed illustrative example with labels.

Furthermore, a comparison was made between the similarity predictions of part-based and component-based approaches. Here, an automated approach was introduced that groups the parts and combines the features. While the overall outcome for part-based and component-based similarity is close, the component-based similarity provides a more stable prediction, whereas the part-based similarity is a more sensitive technique. Consequently, component-based similarity is recommended in partial comparison of simulations and part-based for complete model comparison. On the basis of industrial data, the investigation proves the ability of the presented methods to be scaled up to real data scenarios. Overall, this work improves simulation similarity prediction, while the data representations show promise for outlier detection and clustering of simulations based on similarity score distribution.

8 Graph extraction for assisting crash simulation data analysis

8.1 Challenges in extracting graphs

Towards vehicle Knowledge Graph (KG), the aim is to capture knowledge about vehicle development designs by automatically extracting graphs from a Finite Element (FE)-model representing a vehicle. We live in an interconnected world, and graph theory provides powerful tools for modelling and analysing this interconnectedness. In graph theory, graphs are usually given in advance or easily abstracted from problems. However, for many real-world scenarios, the individual data instantiations of modelled graphs need to be determined from the data before further analysis. Therefore, the construction of high-quality graphs has become an increasingly desirable research problem, resulting in many graph construction methods in recent years [Qia+18]. The graph underlying the abstract structure, which effectively facilitates domain conceptualisation and data management, is the reason for the growing interest in this technology.

The simplest scenario for identifying the connectivity of a graph is when it is associated with a physical problem related to the graph. Such graphs include electrical circuits, power grids, linear heat transfer, social and computer networks, and spring-mass systems [Sta+20]. In this work, crashworthiness studies in vehicle design are of interest, where the transformation of crash simulation data into a graph is a challenging and unexplored area of research. The resulting representation is intended to

provide an abstraction of the problem that allows the use of graph theory methods for further automated analysis of the simulations.

Computer-Aided Engineering (CAE) analysis, mostly with the Finite Element Method (FEM), enables car manufacturers to analyse many design scenarios, nowadays between 10,000 to 30,000 simulations per week [Sch22]. In crashworthiness analysis, CAE engineers optimise the distribution of impact energy in the vehicle structure to reduce injuries to occupants or vulnerable road users. How to characterise the sequence of absorbed energy, known as the load-path, is a fundamental question in this analysis. The results of crash simulations include several outputs, such as deformations, accelerations and internal energy. However, the load-path is not explicitly calculated in a crash simulation. Therefore, a CAE engineer must visualise the sequence to reveal the load-path. This chapter proposes and investigates graph representations for automated identification of the load-path from simulation data.

Parts of the FE-model entities are considered as vertices of the structural graph following the schema in Chapter 5. This chapter wants to detect the graph edges that resemble the structural connectivity of the vehicle. This thesis proposes three approaches to determine this structural graph: Component-Based Graph (CBG), single Part-Based Graph (sPBG) and multi Part-Based Graph (mPBG). The CBG follows two steps: finding the connection of the components (a group of parts) and then identifying the connection of the parts in each component. The sPBG and mPBG graphs have additional steps to convert the component connections to part connections, which requires the detection of the parts that are entangled in the connection that supports the flow of energy.

Defining the vehicle structure as a graph is the first step in load-path detection. Secondly, it is calculated as the longest path in weighted directed graphs, where the edge weights between the parts are intended to represent the energy flow during the crash. This thesis studies different edge weighting functions for three graph extraction scenarios and analyse the determined load-paths from an engineering perspective. In

this work, the investigation is carried out on the frontal structure of a complete vehicle with a multi-scenario load-path in a full frontal load-case. However, the approach is applicable to different impact directions and load-case scenarios.

In summary, the main contributions of this chapter are:

- The conversion of a vehicle structure to a weighted directed graph.
- The extraction of features representing the energy flow.
- A further graph segmentation that captures the time sequence of events.
- An automated detection of the load-path.
- The clustering of simulations based on their load-paths.

8.2 Graph extraction

It is a challenging task to generate a graph representing the structure of a vehicle from CAE data. Finding the connectivity of the parts is complex due to the number of connections, the variety of FE-modelling techniques and the variety of physical types of connections. The best way to obtain this information would be to use the Computer-Aided Design (CAD) database, which is more standardised than CAE. However, this data depends on the company's workflow to maintain the link between the CAE and CAD models, which has yet to be well established. In addition, these databases lack information on the dependencies of the part connections, i.e. all parts are connected without any hierarchy. This hierarchy is essential for defining the direction of the edges and for identifying the vertices of the graph as either dead ends or capable of allowing energy to flow through the structure. As a result, the focus is on finding a method to perform this intelligently using the FE-model based on the location and proximity of parts within it.

The FE-model contains mesh faces and volumes with different entities representing the connections. The mesh is defined by nodes and elements, where the element size defines the resolution of the discretisation. The nodes can represent the vertices, and the elements define the edges for a graph defined as $G(V, E)$ with vertices and edges. Consequently, a FE-model mesh itself represents a graph. However, this graph has drawbacks. A small element size, three to five millimetres, for a complete vehicle, will result in a large number of vertices, up to 20 million. This graph size is computationally expensive for Graph Machine Learning (GML), and the lack of semantics makes it difficult to analyse engineering concepts. Coarsening the crash FE mesh is an alternative, which is a topic in FE-modelling [BX96; CMS04; Mon+17]. However, rather than focusing on post-processing aspects, these studies have mainly focused on reducing the compute time of the FE-simulation. Nevertheless, the result will still be a disconnected graph because a FE-model contains multiple meshes whose connectivity is not element-based. However, the focus is on linking FE entities to extract the structure of a vehicle as a connected graph.

The graph extraction problem is split into two steps in order to determine the connectivity. First, component-level connectivity and then connectivity of parts within a component. Thereby, the hierarchy information is kept in the graph structure. Previously, a grouping method for identifying components was introduced in Chapter 7. Here, this method is extended to search for connections between components. In addition, edges are added to the graph, connecting parts that belong to the same component. To include absorption timing in the graph, the addition of timing segmentation based on the timing of outgoing edges is investigated; see section 8.3.2.

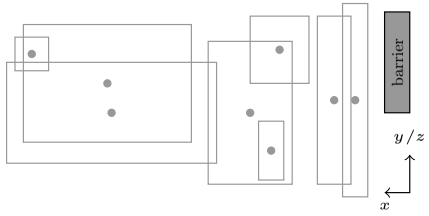
The parts of the FE entities are considered as vertices of the structural graph of the vehicle, which follows the schema in Chapter 5. The objective is to detect the edges that resemble the structural connectivity of the vehicle, for which three scenarios are proposed: CBG, sPBG and mPBG. It is necessary to extract information from the structure of the

vehicle to obtain the connectivity between parts. This is done by creating three-dimensional (3D) axis-aligned boxes for each part that contain the volume of the part geometry, Figure 8.1a. Then, based on the overlap of the boxes, defined rules group them as components, Figure 8.1b, and later form the structure of the graph from the overlap of the boxes. The following three subsections will discuss the detailed differences between these methods and, for now, only describe the general idea.

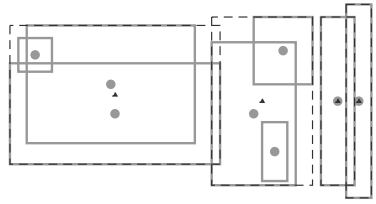
The CBG follows two steps: finding the connections between components, Figure 8.1c, and then determining the part connectivity in each component, Figure 8.1d. sPBG and mPBG have additional steps to convert component connections to part connections, which requires identifying the parts involved in the connectivity that supports the energy flow. Two scenarios for this are explored as single and multi-part-based graphs, figures 8.1e and 8.1f, respectively. All these methods consider a directed graph whose directions are set to have a positive inner product with the impact axis, direction x in Figure 8.1a.

8.2.1 Component-based graph

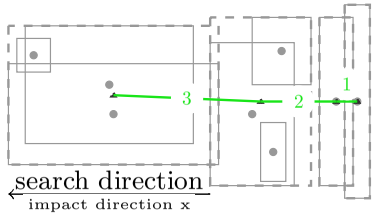
The construction of CBG requires first the detection of the components and then the detection of the connections between components. The component detection considers each part as a box, then groups them together as a component, and finally evaluates the component box. For CBG, in addition to the part vertices, component vertices are also introduced into the graph. The location of these vertices is at the centre of the components, and the component parts are connected to them. For example, in Figure 8.1a with eight parts, four components are detected, and corresponding component boxes are generated in Figure 8.1b. Then, using a threshold value (TLV), the algorithm searches for immediately adjacent components. The thresholding allows having several neighbours. The search algorithm sorts components by impact direction, starting from the impactor/barrier position and moving into the vehicle along the impact direction, e.g. x in Figure 8.1c. Finally, all the parts in each component are connected to the component box.



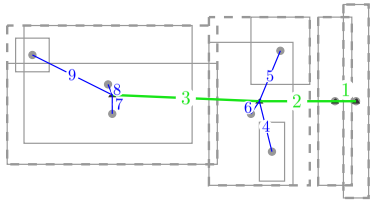
(a) Each part as box with its centre of geometry.



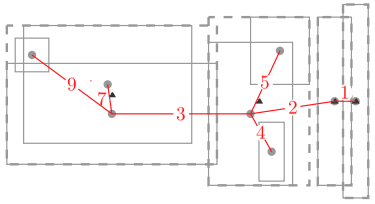
(b) Group parts as a component.



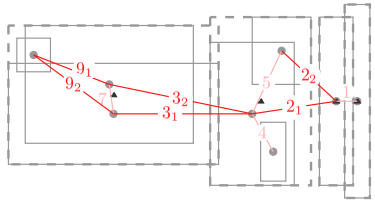
(c) Edges connecting components.



(d) CBG



(e) sPBG



(f) mPBG

FIGURE 8.1: Abstract visualisation of the graph extraction steps. While the method works in 3D, the visualisation is in 2D. Solid square: parts, dashed square: component, circle: centroid of box, triangle: centroid of component, green edges: component to component, blue edges: component to part, red edges: part to part.

The result at this stage, Figure 8.1d, is a connected graph, which is a heterogeneous graph of parts and components. Evaluating the longest path for a heterogeneous graph requires additional evaluation of edge features between vertices of different types. Therefore, the goal is to modify this graph into a homogeneous graph. First, only the components are considered as vertices, the vertices of the parts are deleted, and the features of the component vertices are evaluated based on the parts, as introduced earlier in Chapter 7. This graph is CBG and doesn't contain the detailed features of all the parts. Another approach is to use the heterogeneous graph as an input to find further connectivities of the parts. Sections 8.2.2 and 8.2.3 present this approach.

8.2.2 Single part-based graph

The sPBG is a basic approach to convert the heterogeneous part-component graph into a part graph by transferring the component vertex and its corresponding edges to a part vertex. Because of the single part selection, it is called sPBG and is considered multiple alternative scenarios in Section 8.2.3. There are several ways to determine the corresponding part for each component. First, the largest part, the geometric aspect of the component, is selected as the corresponding vertex for the component connection as a simple scenario. For example, in Figure 8.1e with this consideration, the ○— 1 —○ remains in the same position as the component-part graph because the connecting components contain a single part. The edges ○— 2, 4, 5 —○ move from the component box to the largest part, so ○— 6 —○ is removed. Finally, the edge ○— 7 —○ disappears in the last components and edges ○— 8, 9 —○ move to the other end of ○— 7 —○.

The sPBG graph is characterised by having a main connection from the beginning to the end of vehicles with several dead ends for each master part. It is expected that the identification of the energy flow of the simulation will be limited by the existence of many dead ends. Furthermore, for sPBG, a single part is the representative of a component and

therefore, only a single part interacts with the other parts, which in some cases is not appropriate. For example, the side-member, which is a thin-walled structure, has two U-sections welded and several reinforcement plates. In this example, information about the interactions of the other U-profiles and reinforcement plates will be missed if only one part is considered to represent the component. Next, multiple connections between the components are discussed with mPBG. Multiple connections reinforce the lack of internal connections compared to sPBG.

8.2.3 Multi part-based graph

The mPBG is an alternative to sPBG by allowing multiple representatives for components. This approach allows for part interactions in the components and between components. Here, the component vertices are transferred and distributed using the information from the component discovery process, rather than selecting the largest box. As described in Chapter 7, the component detection algorithm has two scenarios for identifying the components: full and partial overlap merge. Full overlap means a box is completely within the parent box, whereas partial overlap addresses partially overlapping scenarios. These two scenarios are treated differently for mPBG extraction. In the case of a full merge, the part is connected to its parent box, similar to sPBG. However, in partial overlap scenarios, both boxes will represent the component. In this case, a component vertex is transferred to all partially overlapped boxes. Nevertheless, each part will retain its connections to the child based on full merges. Figure 8.1f visualises these two scenarios. The edge ◯— $2, 3$ —◯ branches to two edges ◯— $2_1, 2_2$ —◯ and ◯— $3_1, 3_2$ —◯ respectively compared to the sPBG due to a partial merge. Furthermore, the edge ◯— 9 —◯ branches to ◯— $9_1, 9_2$ —◯ since it is added after the partial merge and belongs to both parent boxes.

8.3 Load-path detection

Understanding how an external load is transferred to a given structure helps to evaluate the performance of different components, improve structural strength and reduce structural weight in structural design and optimisation. The so-called load-path of a component is a concept for tracking the transferred load within a structure, starting from the load points and ending at the support points, which has been studied in structural design for several years [MV09]. Reviews of different approaches to load-path detection are proposing a new metric to find detailed load-paths at mesh size for better component design. However, the load-path is of interest in the context of crash analysis, which involves the interaction of several components. Load-paths are typically defined as vehicle parts capable of generating resisting forces during a crash event [LHB03]. Nine load-paths are first defined and classified in order to identify load-paths during a crash [LHB03]. These can be easily examined for signs of loading after a crash. On the other hand, this work mainly introduces new measures for evaluating real crashes.

The aim is to find the path to compare simulations by highlighting the importance of different paths during the crash. The longest path calculation[1] is used to find the load-paths involved in absorbing the crash energy. This calculation aims to look at the internal energy absorption of the parts as manufacturers optimise the energy absorption capabilities of the load-paths [LHB03]. This uses the Internal Energy (IE) features introduced in Chapter 6. Initially, one has an unweighted graph with IE features for vertices. An essential step is to convert vertex features into edge weights. In this way, the edge weights hold the absorption characteristics, and instead of the longest unweighted path, the potential load-path is computed.

The following subsections first introduce the edge weights as a single feature of the internal energy flow, f_{IE}, and the time segmentation, s_t.

[1]The longest path in a directed acyclic graph, `dag_longest_path()`, from NetworkX.

The f_{IE} is computed from the max of IE (IE_{max}) using internal energy flow calculation, see Section 8.3.1. For s_t, the graph is updated with time segmentation to have absorption time features on the edges; see Section 8.3.2. Finally, Section 8.3.3 will present several ways to combine edge features.

8.3.1 Internal energy flow

The flow equation is applied for the propagation of the IE_{max} feature from the vertices to the edges, f_{IE}. The graph is a directed weighted graph $G(V, E)$ with vertices V, edges E and w assigning each edge a weight $w(e)$. The assumption is that the energy flow from vertex i to j, $w_{i,j}$, is represented by an edge weight between vertices i and j. The energy flow equation relates the absorbed internal energy IE_j of a vertex v_j to the balance of the input and output IE from that vertex to its neighbours:

$$IE_j = \sum_{n \in I(j)} w_{n,j} - \sum_{n \in O(j)} w_{j,n}. \tag{8.1}$$

For a vertex v in a graph, let $I(v)$ and $O(v)$ denote the set of in-neighbours and out-neighbours of v, respectively. The computation starts by computing edge weights with vertices that only have incoming edges, called dead ends. The flow is calculated from the dead ends backwards along their edge directions to find the inflow at the dead end vertices. The active vertices for the next step calculation are the source vertices to the dead ends. Consequently, the inflow energy to the active vertices is found when all their outflow energy is available. Until all its outflows are known, a vertex is withheld from being an active vertex. In addition, there is a different treatment for the dead ends at vertices that have an inflow degree of zero. These source-only vertices reflect where the impact is initiated and where, accordingly, the kinetic energy input takes place. Therefore, these vertices are not considered when they are marked as active vertices. Instead, the edge weights of these source-only vertices are calculated when their outgoing neighbours are

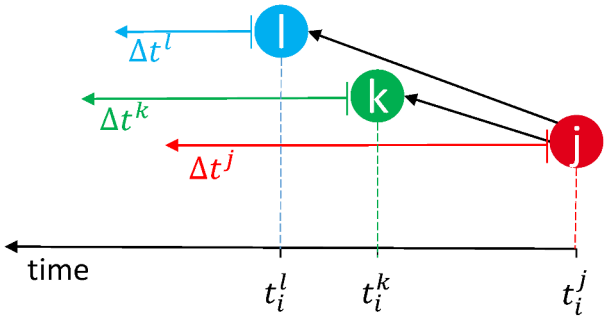
the active vertices. In some cases, the weights of all their outgoing edges have already been evaluated, but the active vertices may have more than one incoming edge. In this case, the energy flow is partitioned to the in-degree, $I(v)$. An unequal stiffness of the structure does not allow an equal distribution. Therefore, equal partitioning can lead to errors in the flow calculation discussed in 8.4.1.

8.3.2 Time segmentation

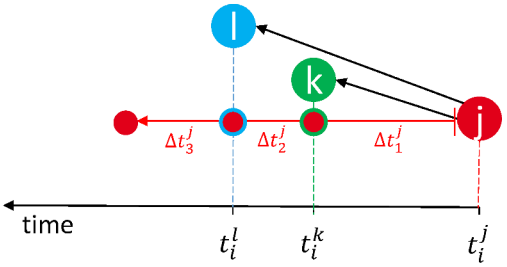
Converting the vertex absorption times into edge weights is more complex than the handling of IE_{max}. This is because the graph connectivity of the vertices differs from the time sequence of the parts that absorb energy. Moreover, the time information of each vertex is an absorption interval, initial absorption time (t_i) to final absorption time (t_n), which can overlap with one of its neighbours, Figure 8.2a. The time interval of absorption is segmented for each vertex to overcome this. The segmentation is based on the t_i value of the successors of the vertex. Accordingly, vertices are added to the graph for each segmented time, connecting each successor vertex to the vertex added for the time segmentation, Figure 8.2b. Only one vertex is added if some successors have the same t_i. Then, the t_i of the successor vertices is sorted to find the connection between the new vertices, Figure 8.2c. In addition, the initial time t_i^k is added as a vertex feature for the kth segment so that all nodes have a t_i. Finally, the edge weight s_t for time segmentation is for a directed edge from m to n defined by $s_t := t_i^m - t_i^n$.

8.3.3 Feature combination

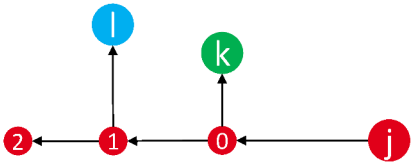
Two approaches are investigated to combine IE_{max} and the timings of part absorption. In the first approach, the vertex feature IE_{max} is modified according to the absorption time before the flow calculation from Section 8.3.1. The integration of the IE curve over time, $IE\Delta t$, is used. The start and end of the integration are set to the minimum $t_{i_{min}}$ and maximum $t_{n_{max}}$ of the absorption times, t_i and t_n, respectively, over all



(a) The initial state of the graph without time segmentation vertices, with visualisation of t_i for each vertex on the time axis. The absorption time is indicated by an arrow in front of each vertex.



(b) Adding two vertices according to the t_i of the successors, k and l, and an additional vertex for representing the remaining absorption time.



(c) The final graph after adding time segmentation vertices and connecting the nodes.

FIGURE 8.2: An example of a time segmentation process for vertex j with two outgoing edges to the successors' vertices of k and l.

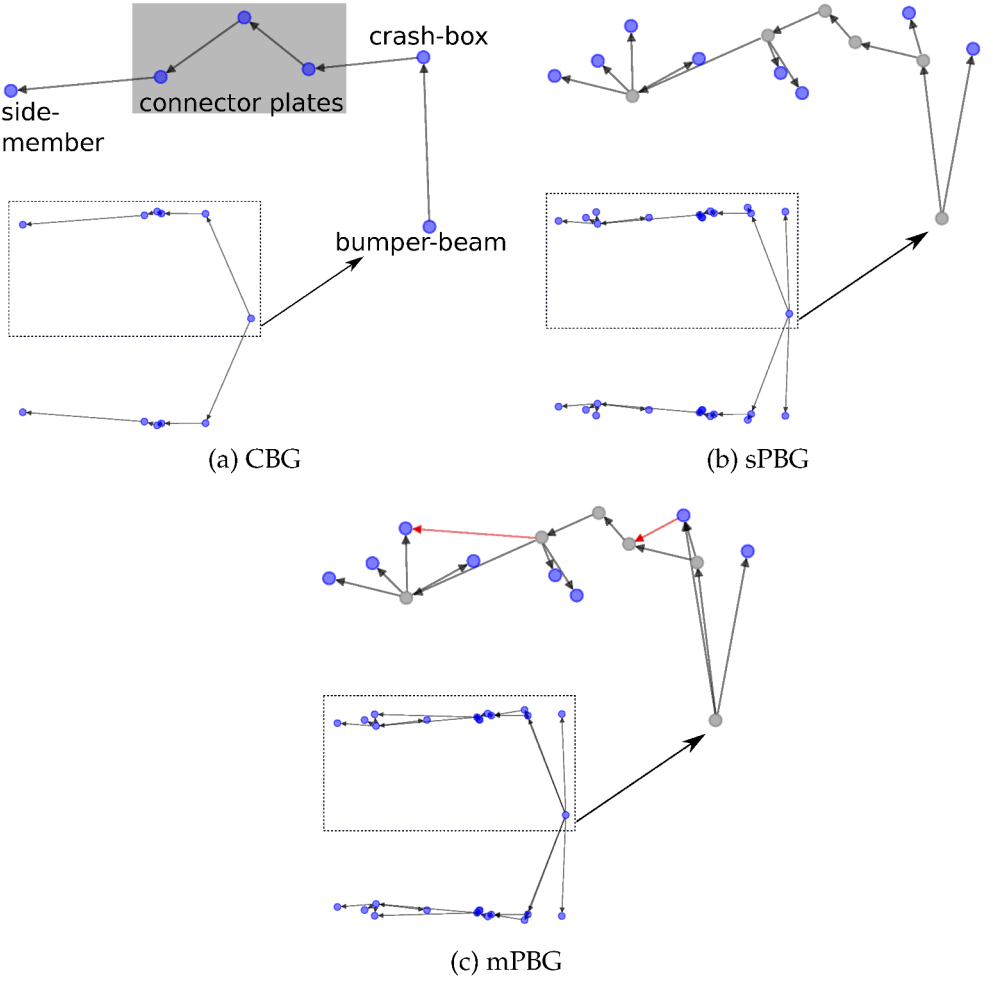
parts. To simplify the calculation, the area under the curve IE is divided into three zones. For each zone, the area under the curve, A, is calculated:

- $(t_{i_{min}},\ t_i)$ unload period, $A_1 = 0$
- $(t_i,\ t_n)$ absorption period, $A_2 = IE_{max}(t_n - t_i)/2$
- $(t_n,\ t_{n_{max}})$ saturated period, $A_3 = IE_{max}(t_{n_{max}} - t_n)$

The sum of these areas is the new node feature, and the combined edge weight is computed, as in Section 8.3.1, with the flow of $IE\Delta t$, $f_{IE\Delta t}$. The second approach is the time segmentation graph. For this graph, The Power of energy absorption (P_e), $P_e = IE/\Delta t$, is calculated where $\Delta t = s_t$, see Section 8.3.2, and $IE = f_{IE}$, see Section 8.3.1.

8.4 Result

The illustrative example presented in Chapter 4 is used to evaluate the method. This study contains 66 simulations; each model contains 27 parts and 11 components. The model structure is the same; therefore, the graph structure remains the same for all simulations. Figure 8.3 shows the extracted graph for CBG, sPBG and mPBG. Here, in the graph visualisation, the vertices are positioned in the centre of its part or component box. In Figure 8.3a for CBG, the vertices of the graph are labelled by these components. For sPBG and mPBG, each vertex refers to a part in figures 8.3b and 8.3c, where the parts corresponding to the vertex of a component are coloured grey. The mPBG has additional edges compared to sPBG that are marked in red, Figure 8.3c. While the CBG, sPBG and mPBG graphs are the same for 66 simulations, adding the time segmentation to the graphs can change the structure for each simulation due to different time sequences. Figure 8.4 shows the differences in two simulations generated by time segmentation for mPBG. The following sections evaluate the computation of the IE flow and the detection of the load-path.



(a) CBG

(b) sPBG

(c) mPBG

FIGURE 8.3: Extracted graphs for the illustrative example, in Chapter 4. A zoomed view of the upper half is shown for each graph[2]. The additional edges for mPBG compared to sPBG are marked red in (c).

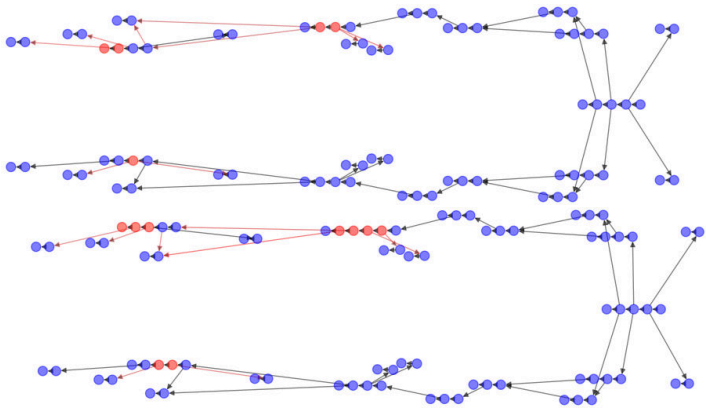


FIGURE 8.4: The mPBG segmentation differences for simulations (0) and (27) due to different times of absorption.

8.4.1 Graph flow

The Root Mean Square Error (RMSE) of the inflow and outflow is used to evaluate the flow calculation as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N} IE_j - \left(\sum_{n\in I(j)} w_{n,j} - \sum_{n\in O(j)} w_{j,n}\right)}. \tag{8.2}$$

The flow calculation has a small error in the order of 2 to $3e-16$ for three graph extraction methods. The comparatively high spread of the RMSE for CBG indicates that for some simulations, the connectivity of the CBG graph is limited, which increases the RMSE for these simulations.

8.4.2 Load-path detection

Here, we first discuss the result of the load-path detection for five reference models, as in Chapter 4, and show how the load-path detection

[2]The zoomed views use `networkx.kamada_kawai_layout()` with vertex distances and positions to improve the visualisation.

characterises the simulations. The best method is then used to classify all 66 simulations. In the reference simulations – 3, 30, 31, 60, 61 — the crash-box thicknesses differ as follows. Simulation 3 has the same thickness on both Left Hand Side (LHS) and Right Hand Side (RHS). Compared to 3, simulations 30 and 31 are less stiff on RHS and LHS, respectively. Whereas simulations 60 and 61 are stiffer on LHS and RHS, respectively, compared to 3.

Figure 8.5 summarises the load-path detection with four edge weights as described in Section 8.3. Columns (a) and (c) are the single feature results for f_{IE} and s_t. The other two columns are weighted with combined features $f_{IE\Delta t}$ and s_{P_e}, columns (b) and (d) respectively. The results of three different graph extraction methods for each scenario are shown, with the detected paths marked in red. Based on structural stiffness, the expected energy load-path for simulations 30 and 60 is at the RHS (bottom) and for simulations 31 and 61 at the LHS (top).

It is expected that whether you get a top or bottom load-path for graphs with f_{IE} edge weight will be opposite for graphs with s_t weight. This is due to the physics of the problem, i.e. stiffer parts take more time for absorption and deform less, which means lower IE. The only exception we observe is in the result with CBG and s_t weighting. Here, the detected path for these simulations does not continue to the side-member, and a different side of the structure is detected compared to sPBG or mPBG. This example shows the limitation of CBG in time feature extraction, i.e. the component level is less sensitive than the part level.

Next, for the combined features, $f_{IE\Delta t}$ and s_{P_e}, for most scenarios, the detected load-path remains in the expected direction of the structure. The only exception is the CBG graph for simulation 3. Simulation 3 is a symmetric model and lacks a dominant load-path due to its symmetry. Again, the CBG method lacks the detail to realise the effect of time in detecting the load-path. The additional obvious observation is that with s_{P_e} weight, the detected path is shorter. This detection describes well that the crash-box influence is much greater than those of the remaining

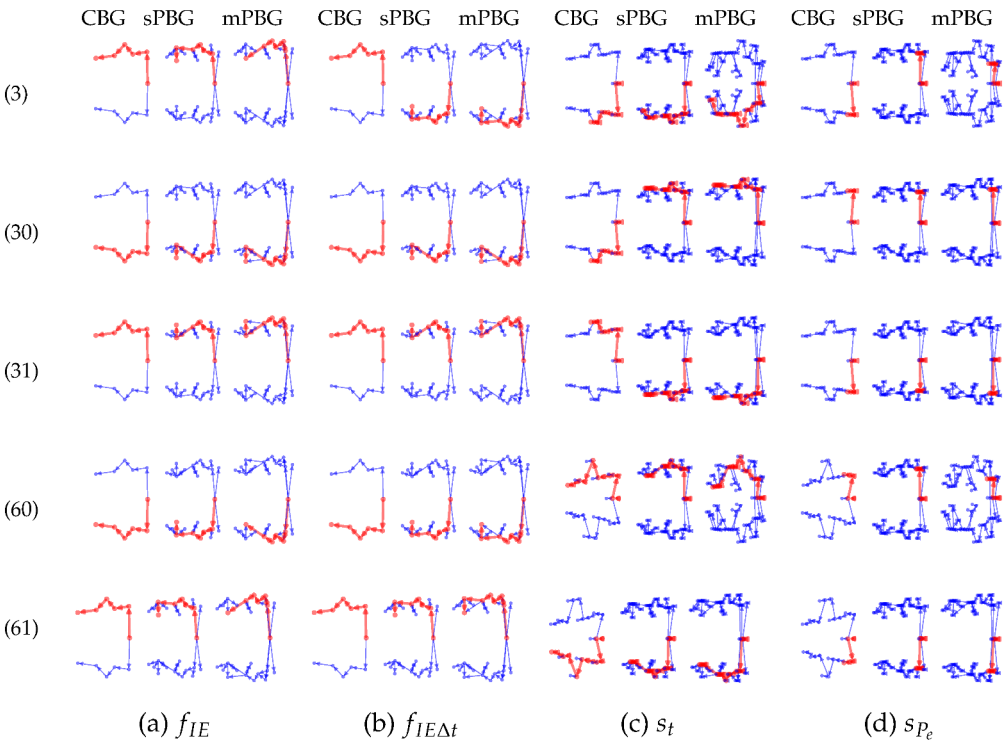


FIGURE 8.5: Load-path detection, marked in red, for the five reference simulations from Chapter 4. LHS and RHS are at the top and bottom, respectively. Each simulation and edge weight setup includes results of CBG, sPBG and mPBG.

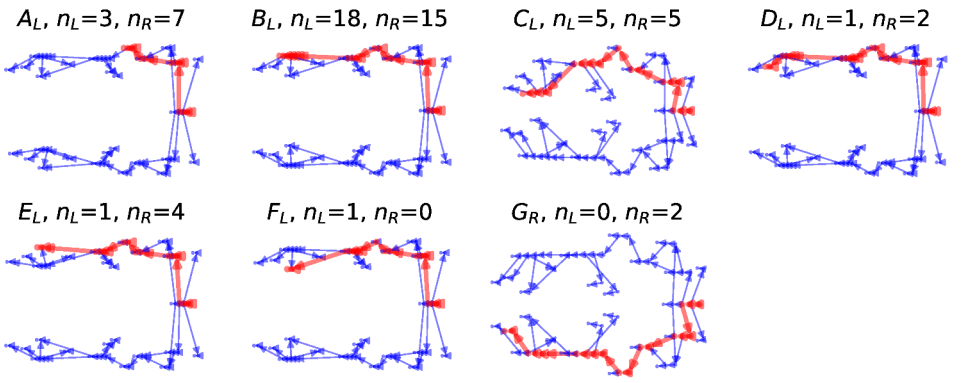


FIGURE 8.6: Identical load-paths, marked in red, are identified for the simulation dataset from Chapter 4. Only one is shown if there is a symmetric pair. n_L and n_R are the number of occurrences of a load-path in the dataset, respectively.

parts. Therefore, this path captures the efficient path of the load rather than the full path along the structure.

Among these approaches, the mPBG with s_t detects the most detailed load-paths, which is better for simulation comparison. As a result, it is used to visually categorise all 66 simulations. This method categorises the data into 12 identical load-paths, where 10 are symmetric pairs, i.e. an LHS path corresponds to an RHS path. Figure 8.6 summarises the clusters. Most of the simulations, 33, are grouped in cluster B. The biggest difference between the clusters is between cluster A and the rest, where the path ends with a crash-box absorption. The remaining clusters have similar absorption for the crash-box and differ in vertex selection for the side-member at the end of the path.
To summarise, it is best to use different graph extraction approaches and edge weights (w) for different applications, as follows:

a) CBG, $w = f_{IE}$: crash mode analysis, advantage: simple and stable.

b) mPBG, $w = f_{IE\Delta t}$: IE flow path analysis, advantage: more details.

c) mPBG, $w = s_t$: simulation clustering using load-path, advantage: sensitivity.

d) mPBG $w = s_{P_e}$: analyse part or component efficiency.

8.5 Summary

Load-path detection in crash analysis was the focus of this chapter using graph approaches. Due to the lack of graphs in the CAE data, graph extraction methods were introduced to convert the CAE analysis of crashes into graphs. To characterise the absorption path of the vehicle structure, not only was the vehicle structure abstracted into the graph, but also edge directions and edge weights were defined. By computing the longest weighted path in a graph, automated detection of load-paths now becomes feasible. Vehicles with the same structural design have an almost similar graph structure, while edge weighting and time segmentation detect differences in load-paths. The method showed promising results, analysing an illustrative example with 66 simulations.

In addition, posture detection methods can be used to further process the data during the crash [Ma+22]. With these methods, part features should remain at the vertex level for active part detection. However, as far as it is known, there is limited research on directed graphs to find the load-path. Furthermore, converting a whole vehicle into a graph requires additional considerations. For a complete vehicle, graph extraction can often lead to several unconnected graphs due to the existence of larger parts. The graph extraction works for sub-models, but further heuristics are needed to extend its application, which is beyond the scope of this work.

This method, as well as being useful for the CAE engineers, the load-path clusters can also be used as labels, which opens up new possibilities for using supervised Machine Learning (ML) for CAE. The next step is to implement graph embedding methods to automatically classify the

results. Finally, a static graph was extracted from the undeformed geometry. As the deformed structure may lead to additional contacts between parts that do not exist in the undeformed structure, it may be useful in the future to consider the deformed structures as well.

9 GAE-vehicle-safety: an ontology for Graph Assisted Engineering in vehicle safety

Modelling Computer-Aided Engineering (CAE) data is challenging because the data is complex and multiple disciplines with different requirements — CAE engineers, Computer-Aided Design (CAD) engineers and attribute managers — interact with the CAE data. However, the flexibility of graph-based modelling allows uncertainties to be considered and the model to evolve. Earlier, in Chapter 5, an introductory attempt was made to define a semantic representation that stores information about the different crash scenarios, the vehicle design deviations during the development process, and the quantities of interest that measure the outcome in Chapter 5. Semantic selections were proposed that follow the development concepts, Finite Element (FE)-modelling terminology, crashworthiness evaluation quantities and other relevant entities. Here, the focus is on aligning the data graph modelling with Semantic Web Technologies (SWT) standards.

A wealth of knowledge about efficient data management for sharing, discovering, integrating and reusing data has been generated in the first 20 years of the SWT field. The field can be characterised as studying and applying ontologies, linked data, and Knowledge Graph (KG)s. The driving idea is that ontologies should be reusable by others, and in the context of the SWT, ontologies are an essential means of data integration, sharing and discovery [Hit21]. In addition to ontologies, the SWT is not just about putting data on the Web. It is about creating links to

allow a person or a machine to explore how data is linked. Accordingly, this chapter introduces the ontology of graph modelling from Chapter 5. The result of this chapter publishes the data using the principles of linked data introduced by Tim Berners-Lee [BL10], and also the container for the workflow to convert CAE data into Resource Description Framework (RDF)s.

9.1 From graph modelling to ontology

If machines are to be expected to perform useful reasoning tasks on documents, then the language needs to go beyond the basic semantics of a schema [MVH+04]. The first level on top of a graph schema that is required for the SWT is an ontology language. This is a formal description of the meaning of the terminology used in Web documents [MVH+04]. Consequently, interoperability and inference are missing from a graph database compared to ontologies. In terms of interoperability, the focus of ontologies is on vocabulary definition. If there is a common vocabulary and data is shared according to a defined vocabulary, other people will be able to use and understand it because it's formally defined. People will be able to define or build applications that use that data. Interoperability is at the heart of the whole concept of the SWT. Next, the inference is the ability to infer new knowledge from existing knowledge. An ontology contains fragments of knowledge that are used to make inferences. The combination of a graph database and ontologies allows new facts to be inferred. Consequently, there is a great advantage to introducing an ontology on top of an existing database.

In the graph data modelling, the Neo4j database was used, which is a property graph [Mil13]. The reason for this selection is that property graphs are the most commonly used in popular graph databases. Property graphs offer a more flexible model compared to other directed edge labelled graphs [Hog+21]. This flexibility is a great advantage for data modelling and was the driving force behind the choice of investigation. On the other hand, ontology concepts are developed hand in

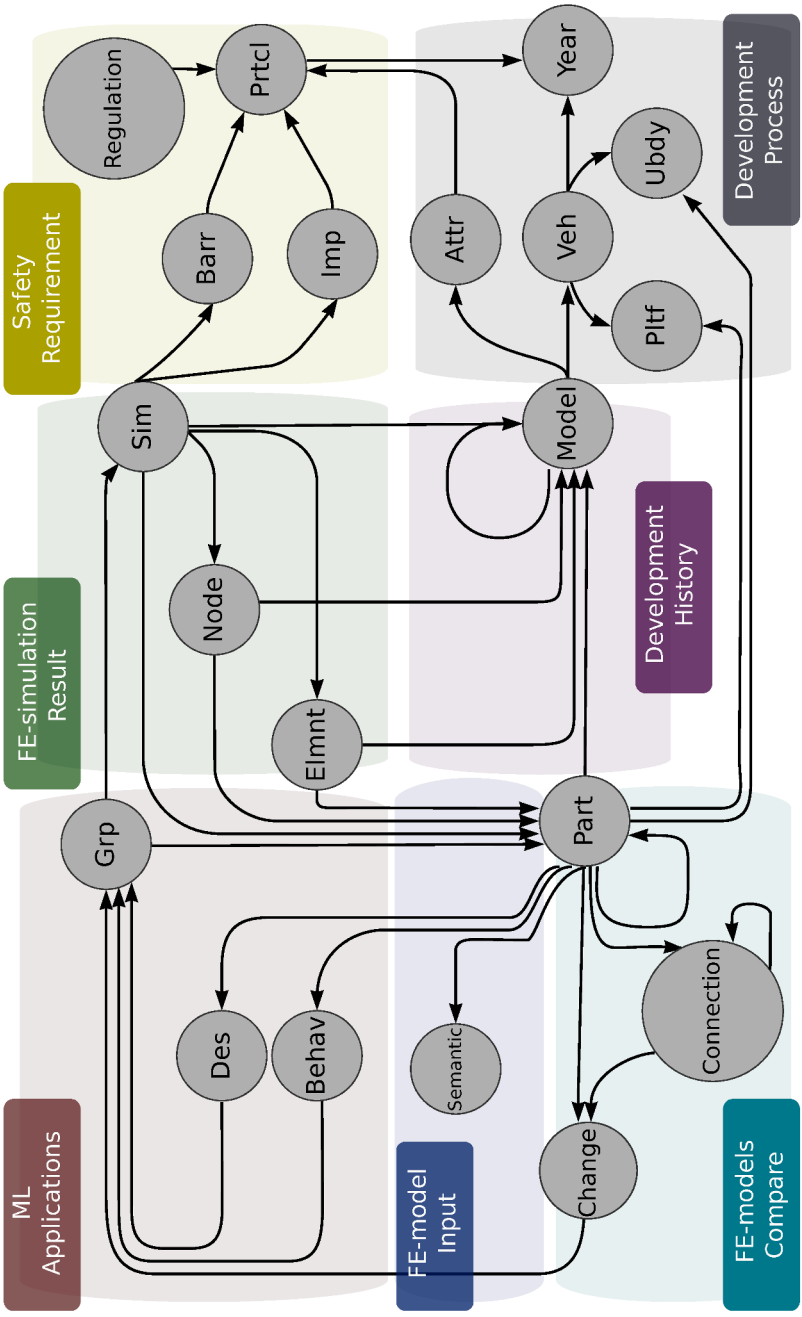


FIGURE 9.1: CAE data graph schema, the lower parts reflect the overall development process, and the upper part is the analysis discipline, more detail in Chapter 5.

hand with RDF and RDF schema (RDFs). RDF is a standardised data model based on directed edge-labelled graphs, and the standard for defining a semantic schema for RDF graphs is the RDFs [Hog+21]. An ontology language is the first level above RDF that is required for the SWT [MVH+04]. The result is a new database in Neo4j that represents the ontology as a graph in Neo4j according to the standards with the neosemantics plugin (n10s)[1]. This allows the import and export of the ontology graph.

Figure 9.1 visualise the schema of the CAE data and the regulations. See Chapter 6 for the detailed concept of this graph modelling. In the following, first, the ontology is defined. Then, the ontology is imported into Neo4j, and the mapping is implemented.

9.1.1 Implementation

Web Ontology Language (OWL) is used among several possible languages to represent an ontology [MVH+04]. The OWL is designed to work with applications that need to process the content of information rather than simply presenting information to humans. OWL enables greater machine interpretability of Web content than that supported by XML, RDF, and RDFs by providing additional vocabulary along with formal semantics. OWL was developed to extend the vocabulary provided by the RDF specification [MVH+04]. The ontology is developed using Protégé[2]. Protégé is an ontology editor tool which, among other open-source tools such as SWOOP and Apollo, is the only one that uses databases to store ontologies. It is a more graphical ontology tool [Ala13]. In addition, the ontology can be imported into or exported from a variety of ontology representation languages, including RDF and OWL.

[1] `https://neo4j.com/labs/neosemantics/`
[2] `https://protege.stanford.edu/`

(a) Imported ontology from Neo4j export.

(b) Restructured ontology in Protégé.

FIGURE 9.2: Tree views of ontology classes and object properties in Protégé.

Since the database is initially available in Neo4j[3], the neosemantics plugin allows RDF and its associated vocabularies, such as OWL, RDFs, to be used in Neo4j. The first step in this process is to upgrade the database of neo4j-community from `4.2.4` to `4.4.12` in order to install the neosemantics plugin. Before loading the `.dump` file, it is essential to perform the configuration setup[4]. In the configuration file of Neo4j, the following is added to `neo4j.conf`.

```
dbms.unmanaged_extension_classes=
    n10s.endpoint=/rdf
dbms.allow_upgrade=true
```

The first line is to be able to have `:POST` and `:GET` commands. The second line is to be able to update the database on loading. Then, the Neo4j server is started, and the neosemantics initialisation is set.

```
CALL n10s.graphconfig.init();
CREATE CONSTRAINT n10s_
unique_uri ON (r:Resource)
ASSERT r.uri IS UNIQUE
```

Then, the server is stopped, and the load is run for the `.dump` file[5].

```
bin/neo4j-admin load --from=file.dump
--database=neo4j --force
```

Once the data has been loaded, the server is restarted with the dump file to be loaded and updated.

Neo4j has support for the export of the database schema as an ontology. This is done with the neosemantics cypher command of the form,

```
GET /rdf/neo4j/onto?format=N-Triples
```

[3] https://github.com/Fraunhofer-SCAI/GAE-vehicle-safety

[4] https://neo4j.com/labs/neosemantics/tutorial/

[5] file available at, https://github.com/Fraunhofer-SCAI/GAE-vehicle-safety

TABLE 9.1: Ontology imported summary to Neo4j.

termination Status	triples Loaded	triples Parsed	name spaces	extra Info	call Params
"OK"	325	543	null	""	{}

As mentioned above, this output can then easily be imported into Protégé. As you can see in Figure 9.2a, this output isn't hierarchically structured. After the definition of the ontology classes, the arrangement of the classes in a taxonomic, sub-class-super-class hierarchy is one of the main conditions for the development of an ontology [NM+01]. Moreover, the class names follow the abbreviations that are good for querying data quickly but not for ontology classes. It is usually a good idea to avoid abbreviations in concept names that the name itself would be indicative of what the concept is. What is essential in naming is to define a naming convention for classes and adhere to it [NM+01]. This led to the development of a new ontology using Protégé with an appropriate name and considering the hierarchy of the data, Figure 9.2b. The final step is to import this ontology into Neo4j by using the following command

```
CALL n10s.onto.import.fetch(
    "file://path-to-file.rdf",
    "RDF/XML"
);
```

Note that the language you import should match the language Protégé will output. Here, the format is RDF. From the loading of the ontology, nodes are generated as defined classes and relationships. Mapping the imported ontology into the database is the final step. An essential step in this is to set the prefixes for the namespaces. The URIs for different ontologies are defined using prefixes. The following prefixes have been defined,

```
CALL n10s.nsprefixes.add("sch",
"http://schema.org/");
CALL n10s.nsprefixes.add("gae",
    "http://www.semanticweb.org/GAE/#");
CALL n10s.nsprefixes.add("akt",
    "http://www.aktors.org/ontology/
    support#");
```

The next step is the mapping of the database to the imported ontology,

```
:param uri: "http://schema.org/";
CALL n10s.mapping.add( $uri + "Vehicle",
"Veh");

:param uri: "http://www.aktors.org/ontology/
support#";
CALL n10s.mapping.add($uri + "Year","Year");
:param uri: "http://www.semanticweb.org/
GAE/#";
CALL n10s.mapping.add($uri + "Platform",
"Pltf");
CALL n10s.mapping.add($uri + "UpperBody",
"Ubdy");
CALL n10s.mapping.add($uri + "Attribute",
"Atr");
CALL n10s.mapping.add($uri + "Barrier",
"Barr");
CALL n10s.mapping.add($uri + "Impactor",
"Imp");
CALL n10s.mapping.add($uri + "Protocol",
"Prtcl");
CALL n10s.mapping.add($uri +
"AdultOccupantProtection", "Aop");
CALL n10s.mapping.add($uri +
```

```
"ChildOccupantProtection", "Cop");
CALL n10s.mapping.add($uri + "SafetyAssist",
"Sas");
CALL n10s.mapping.add($uri +
"VulnerableRoadUser", "Vru");
CALL n10s.mapping.add($uri + "ResultPage",
"Resultpage");
CALL n10s.mapping.add($uri + "Class",
"Category");
CALL n10s.mapping.add($uri + "Change",
"Change");
CALL n10s.mapping.add($uri + "CommonNode",
"Common_Nodes");
CALL n10s.mapping.add($uri + "Behavior",
"Behav");
CALL n10s.mapping.add($uri + "Design",
"Des");
CALL n10s.mapping.add($uri + "Group",
"Grp");
CALL n10s.mapping.add($uri + "Model",
"Model");
CALL n10s.mapping.add($uri + "Element",
"Elmnt");
CALL n10s.mapping.add($uri + "Node",
"Node");
CALL n10s.mapping.add($uri + "Part",
"Part");
CALL n10s.mapping.add($uri + "Connection",
"Connection");
CALL n10s.mapping.add($uri + "CommoneNode",
"Common_Nodes");
CALL n10s.mapping.add($uri +
 "RigidBodyElement", "RBE");
```

```
CALL n10s.mapping.add($uri + "Weld",
"Weld");
CALL n10s.mapping.add($uri + "Semantic",
"Semantic_Type");
CALL n10s.mapping.add($uri + "Simulation",
"Sim");
```

9.1.2 Lesson learned

Previously, we looked at the process of creating an ontology for existing graph modelling. Here is a summary of the issues faced in developing the ontology for this domain.

- From a technical point of view, it wasn't obvious how to add an ontology for an existing graph modelling domain. Most of the existing work in this area uses existing ontologies and maps them to their data model.
- In the absence of a central domain ontology, the starting point was to review existing ontology classes using the Linked Open Vocabularies (LOV)[6], which includes most open data ontologies. However, none of the existing vehicle ontologies are available from this platform, which means that additional effort is required to find the relevant classes.
- During the project, there were many updates for neo4j, each time adding overhead for technical installation. This is an indication that the product still has a lot of room for growth. In this case, where the data was rather small, the upgrade is possible within a few days. This may be more problematic for larger databases.

[6] https://lov.linkeddata.es/dataset/lov/

9.2 Query CAE data

So far, the Graph-Aided Engineering (GAE) ontology was introduced, and its benefits demonstrated within the CAE domain and vehicle development process. Some examples of cypher queries are presented to answer the questions specified in Section 5.2. In order to evaluate the coverage of the GAE ontology, queries were written for all the competence questions described in Section 5.2. Here are simple examples of such queries:

- What is the most common design change in a development tree?

```
MATCH (d:Des)-[]-(p:Part)-[]-(s:Sim)-[]-(m:Model)
RETURN DISTINCT apoc.node.degree(d) AS dCount,
m.model_name, d.des_pid
ORDER BY dCount DESC LIMIT 10
```

- What are the top three strategic parts for a particular load-case?

```
MATCH (d:Des)
WITH max(apoc.node.degree(d)) AS maxD

MATCH (d:Des)-[]-(p:Part)-[]-(s:Sim)-[]-(m:Model)
WITH d,p,s,m,maxD,
    apoc.node.degree(d) AS dCount
RETURN DISTINCT
    CASE
        WHEN maxD*.85 < dCount
           < maxD*.95 THEN d.des_pid
        ELSE 0
     END AS imprt_des,dCount
     ORDER BY imprt_des DESC LIMIT 3
```

- What are the top 10 essential analyses of the development?

For this,first, the computational graph is defined.

```
CALL gds.graph.project.cypher(
   'devTree',
   'MATCH (m:Model) RETURN id(m) AS id',
   'MATCH (m:Model)-[r:MODEL_REF]->(n:Model)
       RETURN id(n) AS source, id(m) AS target'
)
YIELD
  graphName AS graph, nodeQuery,
  nodeCount AS nodes,
  relationshipQuery, relationshipCount AS rels
```

Then, the PageRank is used to find the top essential analysis.

```
CALL gds.pageRank.stream('devTree', {
   maxIterations: 20,
   dampingFactor: 0.85,
```

```
   tolerance: 0.1
})
YIELD nodeId, score
RETURN
gds.util.asNode(nodeId).model_name AS name,
score
ORDER BY score DESC, name ASC
```

In order to demonstrate the usability of GAE for the retrieval and further extraction of information from CAE data, some query examples were presented. There is still a CAE domain dependency for processing this data, although an ontology is defined for modelling the data, and the data can be queried with the defined ontology. As the VSSo has the structure of the vehicle components defined in the ontology, there is a great advantage in linking the GAE ontology to the VSSo [Klo+18b]. In the GAE, however, this is limited to the platform and the upper-body of the vehicle. Linking these two ontologies will require some additional work. Currently, CAE parts lack the metadata to identify which component they belong to. However, implementing this workflow and connecting the ontology will support cross-functional development in the company. For example, instead of depending on CAE entities such as a part or node element, the output can be a component signal output, e.g. engine acceleration signal.

9.3 Summary

The development of an ontology is an iterative process. Rather than being developed by a small team, it should be developed within a community. With this in mind, an ontology is added to the existing work on graph data modelling. Many new classes have been defined for this ontology. This is due to the lack of related work, according to the best known. Defining new classes is not preferred following the linked data

standards and "Ontology Development 101" [NM+01]. The development of ontologies in this area is still in its primary phase. The novelty explains the need to define new classes. The aim was to support the growth of work in this area by defining the ontology and making it accessible.

Maintaining a consistent class hierarchy may become a challenge as domains evolve [NM+01], since,

- There is no one right way to model a domain. There are always viable alternatives. The best one will almost always depend on how you intend to use it and how you intend to extend it.
- The development of an ontology is necessarily an iterative process.
- Ontology concepts should be close to objects (physical or logical).

Given these challenges in maintaining and extending an ontology, it is recommended that all future contributors follow the conventional guidelines described in "Ontology Development 101" [NM+01], as this thesis has attempted to do. The workflows were tested on four different data sources: Volkswagen, Porsche AG, China Euro Vehicle Technology AB and the University of Wuppertal. For two of these datasets, Porsche AG and the University of Wuppertal, an optimisation study was carried out. The other two included analyses of the engineering made in the development processes. The aim was to reduce the effort required for companies to implement the workflows in order to make them easier to use. It is interesting to note that even though there are differences in the companies' data, the overall workflows and metadata format are common. For example, all three companies capture the development process step in XML format data. The research data, however, clearly lacks this structure and the presence of metadata. The difference shows that the companies have already implemented workflows for traceability and reproducibility due to business requirements but that this has not been the case in the research sector.

Considering the use case presented in Section 1.3.1, a major obstacle to achieving the vision of supporting Autonomous Driving (AD) with passive safety is the availability of vehicle CAE data in the Internet of Vehicles concept. Manufacturers are currently reluctant to open up their data due to safety regulations in legal cases. Therefore, an external driver is needed to achieve this. There are two possible scenarios that could support activities in this direction. The first is a change in safety regulations. There are still many crash scenarios that are not covered by safety regulations. Projects such as VIRTUAL [Eurb] and OSCAR [Eura] are already investigating how to fill this gap in order to allow for a diversity of analyses. There is a clear overhead for companies to perform all these tests with a physical test. Therefore, It is highly likely that the release of the CAE model will be required for vehicles on the market. In the case of the pedestrian analysis, this is already partly the case, as there are many points of impact on the vehicle that need to be evaluated. Currently, car manufacturers are asked to submit their CAE results, and only a few selected points are analysed, as many impact points should be evaluated to assess the performance of a vehicle.

The second support for the release of CAE data could come from the company's challenge to meet the conflict-free vision in AD. It has been a long time since the technology was ready for full Autonomous Vehicles (AVs) to be available on the market. However, the demand for conflict-free vehicles has been hampered by legal issues and market fears. To overcome this, companies have tried to improve their methods and increase the variety of data used to train the models. However, it is likely that even experienced drivers will find themselves in a situation on the road that is difficult to control and could lead to an accident. Vision Zero (VZ) is a public programme aimed at eliminating road deaths and serious injuries. The idea was first introduced to the Swedish Parliament in 1997, and the programme has helped to reduce the number of fatalities, but the number of accidents is still far from zero. There is no such thing as a real zero in nature, and an unattainable goal could be the problem for the AVs. Therefore, it may be possible to change the target

for AVs to safe accidents instead of zero accidents when car manufacturers enable multi-scenario crash analysis. In this case, carmakers may choose a more collaborative solution than single-vehicle safety analysis and make the CAE data available.

10 Method programming

This chapter is an overview of the code developed in this thesis. In order to take advantage of web-based reporting for advanced visualisation of the data, this work was oriented towards web-based reporting. Figure 10.1 is an illustration of the data processing of this project. The focus here is mainly on the development of the backend using the Django framework (python programming language) and the development of machine learning functions. Accordingly, some data visualisation examples are developed using plotly-dash and Django templates to demonstrate usability. However, for scalability and better performance, it is recommended to transfer this work to a front-end platform. This is partly done in CAEWebVis. The following sections first present the benefits of web-based reporting for Computer-Aided Engineering (CAE). This is followed by a summary of the functionality developed in this package. For more information about the code and how to use it, please visit the GitHub[1] project of the thesis.

10.1 Why web-based development

The lack of semantics in CAE data makes the data disconnected. Disconnected data within an OEM, and even more so between OEMs, is one of the barriers to realising the car-graph vision. The complexity of raw simulation data with the lack of semantics in the current vehicle development workflow means that design engineers and attribute managers rely on reporting from CAE engineers. This static reporting limits

[1] https://github.com/Fraunhofer-SCAI/GAE-vehicle-safety

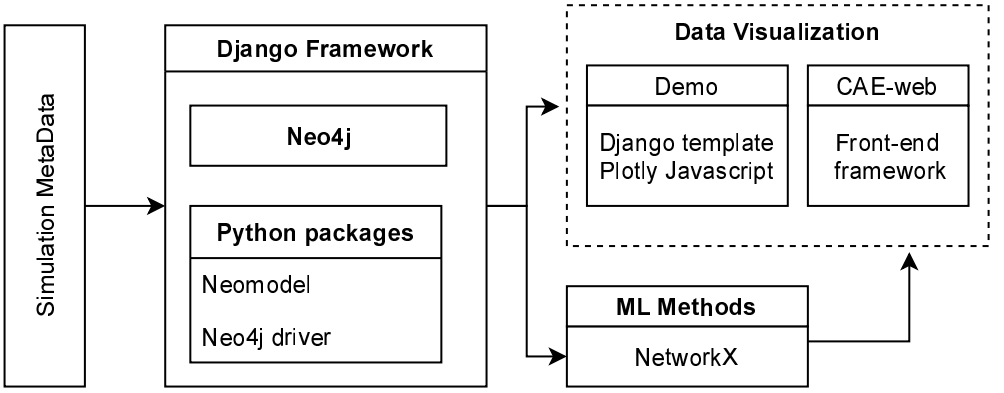


FIGURE 10.1: Data processing workflow.

independent exploration of the data. This limitation results from the company's dependency on available software and the skills required to use CAE, Product Data Management (PDM) and/or site data management (SDM) tools. The lack of multidisciplinary collaboration reduces the efficiency of problem-solving. In addition, the presence of semantics enables more efficient exploration of the data. Currently, the size of the CAE model limits the number of simulations that a CAE engineer can simultaneously load into post-processing software - four to six simulations.

Therefore, the development of a car-graph is supported with a web-based platform called CAEWebVis to enable semantic reporting for CAE. Web-based platforms for CAE data were introduced as early as the CAE bench [Häg+10], which was later commercialised by MSC Software. This idea was followed by a number of other companies, such as SCALE and d3View. Later, an open-source CAE platform was introduced that replaced the pre- and post-processing tools and the solver [Lee+20]. However, most OEMs are still poorly integrated with web technologies after two decades. The barriers are high implementation overhead, lack of software independence from the enterprise data structure, and the use of relational databases that are not as agile as graph

databases.

What is new about this web-based platform is the use of a graph database, which adds semantics to the data and is an essential step in building a Knowledge Graph (KG). Furthermore, these databases' schema flexibility, real-time updates and powerful recursive path query make them superior to relational databases. Among the available options for graph databases, this work uses a property graph (Neo4j). Neo4j has the flexibility to store properties for both nodes and edges. The project is structured in Django, a Python web framework, to allow reuse and speed up web visualisation. The target audiences for CAEWebVis are CAE attribute managers, CAE engineers, design engineers and data analysts in automotive R&D. CAEWebVis uses a web-based user interface that allows project members from different teams to access CAE results, understand design performance limits, compare simulations and apply algorithms to the vehicle graph. The vision is to have a low overhead tool to support the developments needed to build safer cars.

10.2 Functions

The code is structured in the following[2]

- `ld_data`: Loading the data into the database, each of the remaining functions can be enabled or disabled during the loading process. The default is to load the simulations with their energetic parts Chapter 6.
- `nrg_fts`: Extracting the energy properties from the simulation output[3], in Chapter 6.
- `sim_rnk`: Similarity prediction of the simulation part of the bipartite graph with different simRank methods, in Chapter 7.

[2]Current implementation is only for LSDYNA.
[3]ls-dyna output, binout.

- `comp_detn`: Recognizing components and grouping parts, in Chapter 7.
- `physc_grph`: Adding connections between the parts to convert the vehicle into a directed connected graph, in Chapter 8.
- `ld_pth_detn`: modifying the physical graph edge features and structure to detect the crash load-path, in Chapter 8.

Each of these functions is developed as a Django app with a template for visualising the data. You can download the project and further documentation[4] and the GitHub repository[5]. You can download each application separately and follow the instructions for input and output formats. However, if you are using the complete setup, the best scenario is to call the functions within the load data. To adapt the code to different file structures, you need to define the configuration file to set the path to the data.

10.3 Limitations

Introducing the car-graph problem to the KGs community promotes a new type of problem for graph analytics. Note that the difference in problem typically results in a shortage of available analysis packages. NetworkX [HSSC08] and DeepSNAP [Rex20] are examples of Python libraries that assist efficient analytics and deep learning on graphs. The usage of NetworkX is hampered since the effects of edge weights differ in the CAE domain in contrast to, say, analysing customers. Furthermore, DeepSNAP methods are available for a heterogeneous graph, and its methods look primarily at having multiple nodes for labelling. However, in engineering, having multiple edges implies having different functions in the message passing, for which modified or new methods or needed. The hope is that the automotive industry's needs will

[4] `https://fraunhofer-scai.github.io/GAE-vehicle-safety`
[5] `https://github.com/Fraunhofer-SCAI/GAE-vehicle-safety`

increase the interest in research on different types of Graph Machine Learning (GML) methods in the future.

11 Final discussion

The lack of reusable data in the field of Computer-Aided Engineering (CAE) was the basis for the idea and structure of this study. The focus was on crash simulation, which is one of the most computationally intensive analyses in the automotive industry. The primary goal was to translate the engineering problem into graphs, taking into account graph structure and properties, in order to extend the use of the data. What this thesis achieved overall is summarised as follows,

- New representations of the data to reveal patterns and discover new knowledge, in Chapter 6,
- Similarity prediction of simulations to enhance the interconnectedness of data, in Chapter 7,
- Development of methods for automated component detection to support component-level post-processing analysis, in Chapter 7,
- Vehicle structure graph generation to bridge graph structure analysis algorithms to the CAE domain, in Chapter 8,
- Load-path detection based on impact direction, in Chapter 8,
- A graph modelling and an implemented ontology for CAE, in chapters 5 and 9.

For each point, the summary and specific scope for future work were presented in the corresponding chapter. A more global conclusion and discussion is provided here. In the course of this research, the obstacles encountered will also be highlighted. This is followed by a presentation of possible future work in this area.

11.1 Conclusion and future work

The complexity of CAE raw data and the lack of semantics in the current vehicle development workflow causes design engineers and attribute leaders to rely on CAE engineer reporting. However, this static reporting restricts the independent exploration of the data. The lack of semantics in CAE data makes the data disconnected and hinders multidisciplinary collaboration, which degrades efficient problem-solving. It is envisioned that the introduced car-graph empowers semantic reporting for CAE. It should enable project members from different teams to access the CAE results, understand the design performance limitations, compare simulations, and use Machine Learning (ML) algorithms on the car-graph. In particular, the capability of the method and workflow was demonstrated in knowledge discovery, searchability and clustering of simulations with a focus on crash Finite Element (FE)-simulations. Due to a lack of time and labelled data, the exploration of the method is still at a proof-of-concept level. This fact is the motivation for the release of the method as code-free software. This will allow CAE engineers to further evaluate the capabilities of the method. It is hoped that this first step will encourage the community to continue to grow.

Overall, the focus has been on finding similarities and trends in the data for the benefit of CAE engineers working in the automotive industry. The target was to answer questions with graph analytics and ML rather than just developing an ontology without a clear use case. Graph modelling has been divided into seven segments in Chapter 5. In this way, data modelling will enable the growth of data modelling for other CAE disciplines within the automotive industry as well as other industries. Among these segments, "Safety Requirement" has a specific graph structure for crash simulation that defines the analysis requirements. In addition, the features loaded in the "ML Applications" segment are also specific to FE crash simulation analysis, which is the internal energy absorption of parts.

In order to maintain the relationship between the number of simulations

and the number of features, a relatively small proportion of the data was loaded. However, more data could definitely be loaded into the graph database to enrich it. Regarding the CAE data modelling, there is further information available that is beneficial to include and will further enhance the graph model. For example, more semantics of the configuration of a vehicle, e.g. engine or roof. Material, accelerometers, section forces, deformations and safety requirements could also be included in the simulation output. Moreover, incorporating Computer-Aided Design (CAD) data, such as design guidelines and manufacturing limitations on the design, would bring the CAE engineers to the next level for efficient problem-solving. For requirements, Euro New Car Assessment Program (NCAP) is only one of many safety assessment protocols, and other protocols and requirements are missing. In addition, for benchmarking purposes, many vehicle characteristics, such as vehicle dimensions or weight distribution, exist in other public data sources and can enrich the data.

It has been shown that the internal energy output has the ability to detect differences between simulations. For other CAE disciplines, however, this may not be the case. This is because the Internal Energy (IE) is the most important quantity for the main physics of passive safety, i.e. the conversion of Kinetic Energy (KE) to IE absorbed by the parts. As long as the feature engineering is tuned to the domain of study, e.g. IE or IE flow for crash simulation, the current representations and methods can be used for other CAE domains. An example of this is the use of frequency outputs for Noise Vibration Harshnesss (NVH) analysis. It is important to choose features that allow the underlying analysis to be represented. In this project, due to the lack of labelled data and the reliance on engineering judgment, this was one of the most time-consuming steps.

Implementing graph modelling for other CAE domains is required to enable graph-based searching across the entire CAE domain. The main benefit of extending the applications of graph analysis methods to the CAE domain will be to support cross-domain solutions. Including the

multi-discipline requirements as the search object will support ranking the cross-discipline solutions. However, including more simulation outcomes will also require a feature embedding to combine the features or aim for a SimRank formulation with multi-edge weights. Another step to enhance searchability is to improve the data labelling to leverage result assessment. Enhanced graph models with larger numbers of simulations stored would also allow applying Graph Neural Network (GNN)s in this domain. In this way, design features support further link prediction tasks such as the similarity of the FE-models' inputs or cause-effect relations that interconnect the input designs and output features.

In the context of the code transfer, the code was developed in the Python programming language and focused on the LS-DYNA input and output files. The code was developed in such a way that the reading and other processes are well separated, taking into account these solver dependencies. As a result, any transfer of this code will have to be supplemented with a reading function. It is expected that this transfer of work will take a few hours to complete. In future workflow, using formats such as VMAP[1] will have the advantage of removing solver dependencies.

Furthermore, future vehicle safety aspects consider assisting active safety systems with passive safety characteristics, as well as supporting broader crash scenarios and diversity in crash analysis. Therefore, the car-graph is designed to bridge autonomous driving with product identity during development, using CAD databases to incorporate vehicle configurations. In this way, the car-graph shall allow an extension of the safety evaluation from regulated tests, which are just examples of real crash scenarios, to more diverse crash scenarios. As a result, car-graph will couple active and passive safety to associate autonomous driving with safe crashes in situations where a crash is inevitable.

[1] A new Interface Standard for Integrated Virtual Material Modelling in Manufacturing Industry, `https://www.scai.fraunhofer.de/en/projects/VMAP.html`.

11.2 Final discussion

The Knowledge Graph (KG) is one of the components of the Semantic Web Technologies (SWT), as discussed in Chapter 2.1. The strength of the KG lies in the data engineering that builds the semantics with the modelling of the graph data and the ontologies. The defined semantics and relationships are used for further graph data analysis or reasoning about the data. ChatGPT [Bro+20] and its released model in 2023 is providing Large Language Model (LLM)s access to external sources, a.k.a. tools or plugins, and has become a very active area of research with rapid progress [HDY23]. This effect raises questions about KG and its growing field of application in the feature. Therefore, in addition to discussing future work, it is relevant to have a broader overview of knowledge engineering and to compare Language Model (LM)s and KGs.

Knowledge engineering as a discipline has changed considerably since its initial flowering during the period associated with the development of expert systems in the 1980s. Knowledge engineering is defined as the discipline of building and maintaining processes that produce knowledge, which provides a framework for understanding the history of knowledge engineering, Figure 11.1. Figure 11.1 clearly shows the importance and growth of the LMs and particularly of the LLMs in the present decade. LLMs are neural networks that are trained on a large data source and require considerable computing power. What we see in this figure is that the era of SWTs is currently stabilising, while the use of LLMs is growing. An interesting question for the field of knowledge engineering is what the future of this field will look like. With impressive results from LLMs, but they are still [Vra23]:

- Output with hallucination [Wel+19] and requires ground truth
- Expensive to train and operate
- Difficult to fix and update
- Difficult to audit and explain, which is necessary for domain analysis

- Inconsistent responses
- Struggles with low-resource languages
- In a coverage gap in the long tail of knowledge

None of these problems exists for KGs, to the best known. Therefore, using KGs to generate AI solutions for storing domain-specific knowledge still has great potential and utility. It is hoped that research into augmented LMs [Mia+23] and symbolic AI [FV20] will bridge the domains of KGs and LLMs to exploit the capabilities of both.

11.3 Limitations

In the scope of KG for CAE, the focus was only on crash simulation, but an attempt was made to consider the wider scope of KG for CAE and even for mechanical engineering. This points to the lack of standards and consistent use cases as the main problem in this area. There are researches such as ASSESS [Wal+23], MpCCI ontology [Mey+20], but none of them is published with a persistent URI (PURL, DOI, w3id) and following standards. As the strength of existing KGs lies in the reuse of existing ontologies, the lack of adherence to World Wide Web standards makes the reuse of terminologies, and hence ontologies, time-consuming. It is therefore important to emphasise that for any future work within domain-specific KGs, it is important to follow the SWT groups and guidelines.

Companies are also interested in building KG. However, as with any industrial intellectual property, there is a fear of releasing ontologies. Having open ontologies, especially with the extra focus that exists on the importance of the data, is a time-consuming discussion for many companies. Therefore, it is essential to emphasise the benefits of open ontologies and the difference between opening company data. Like all the existing technical standards that have elevated the domain, the implementation of an ontology is like an extension of the terminology. Companies can have open-source ontologies. But they can still keep

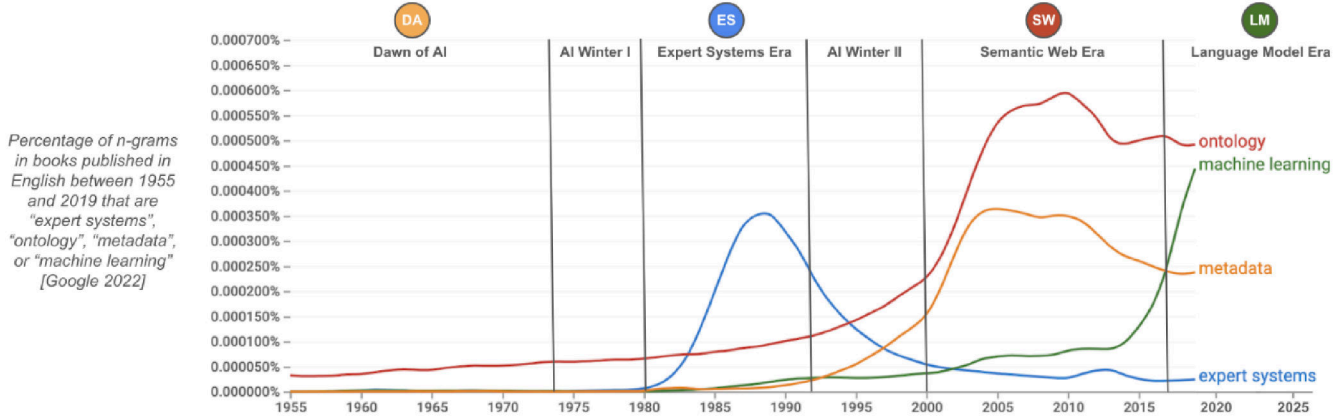


FIGURE 11.1: Seventy years of evolving requirements for knowledge production processes [Gro+23].

their databases. In this way, the elevation of KGs with LLMs will be more feasible. Moreover, in the case of collaboration between companies or outsourcing work to the supplier, the projects will still benefit from common ontology.

The lack of labelled data, which limits unsupervised learning methods, is another challenge in this area. One approach for improving data labelling is the integration of the current method in companies as a dynamic report platform to collect feedback from engineers, e.g. CAEWebVis[2] introduced in Chapter 10. Only with feedback on the predicted similarities from engineers can one envision a transferring of the unlabelled CAE data to labelled data. Such labelling will open up new possibilities to empower further ML solutions, e.g. GNNs. The illustrative example, databases and source code are released with a user tutorial to enable further exploration of graph analysis for the CAE process[3].

Developing a KG and using Graph Machine Learning (GML) to extend discovery to nontextual data is challenging. Textual data is where most of the resources, examples, and related work come from. As with any cross-domain project, there was a noticeable lack of relevant work and communities to support this work. This challenge causes projects of a new domain ontology to forget or skip thinking about what questions they are trying to answer. This is the most important consideration to actively consider along any domain-specific KG project. Nevertheless, finding good questions to make good use of KGs is challenging; no one is alone in this, but it is worth trying. This work has opened up a significant new area and has attracted much interest, and it is hoped to continue working with a larger community. The bottom-up approach to data modelling and answering questions was what made the data modelling stand out. In addition, over the years of this project, more interest and understanding of this work has been clearly seen with international conferences showing the importance of data searchability is now being introduced into the field.

[2] CAEWebVis.scai.fraunhofer.de/

[3] github.com/Fraunhofer-SCAI/GAE-vehicle-safety

Bibliography

[ACG21] Daniel Alvarez-Coello and Jorge Marx Gómez. "Ontology-based integration of vehicle-related data". In: *2021 IEEE 15th international conference on semantic computing (ICSC)*. IEEE. 2021, pp. 437–442.

[Ack+08] Sascha Ackermann, Lothar Gaul, Michael Hanss, and Thomas Hambrecht. "Principal component analysis for detection of globally important input parameters in nonlinear finite element analysis". In: *Weimar Optimization and Stochastic Days 5.0*. 2008.

[AGMC07] Ioannis Antonellis, Hector Garcia-Molina, and Chi-Chao Chang. "Simrank++: Query rewriting through link analysis of the click graph". In: *Proceedings of the 17th international conference on World Wide Web*. 2007, pp. 1177–1178.

[Ala13] Emhimed Salem Alatrish. "Comparison some of ontology". In: *Journal of Management Information Systems* 8.2 (2013), pp. 018–024.

[AS21] Bilal Abu-Salih. "Domain-specific knowledge graphs: A survey". In: *Journal of Network and Computer Applications* 185 (2021), p. 103076.

[Ben+17] Filippo Benvenuti, Claudia Diamantini, Domenico Potena, and Emanuele Storti. "An ontology-based framework to support performance monitoring in public transport systems". In: *Transportation Research Part C: Emerging Technologies* 81 (2017), pp. 188–208.

[Ber05] Pavel Berkhin. "A survey on PageRank computing". In: *Internet Mathematics* 2.1 (2005), pp. 73–120.

[BFG05] C.-S Böttcher, S. Frik, and B. Gosolits. "20 years of crash simulation at Opel - experiences for future challenge". In: *LS-Dyna-Anwenderforum, Bamberg, Germany* (2005).

[BL10] Tim Berners-Lee. *Linked data*. `https://www.w3.org/DesignIssues/LinkedData.html`. 2010. (Visited on 10/01/2023).

[BMM18] Gerrit Bagschik, Till Menzel, and Markus Maurer. "Ontology based scene creation for the development of automated vehicles". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2018, pp. 1813–1820.

[Bou+19] Flavien Boussuge, Christopher M Tierney, Harold Vilmart, Trevor T Robinson, Cecil G Armstrong, Declan C Nolan, Jean-Claude Léon, and Federico Ulliana. "Capturing simulation intent in an ontology: CAD and CAE integration application". In: *Journal of Engineering Design* 30.10-12 (2019), pp. 688–725.

[BRK21] Mohamed Karim Belaid, Maximilian Rabus, and Ralf Krestel. "CrashNet: An encoder–decoder architecture to predict crash test outcomes". In: *Data Mining and Knowledge Discovery* 35.4 (2021), pp. 1688–1709.

[Bro+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[BS22] Akshay G Bharadwaj and Binil Starly. "Knowledge graph construction for product designs from large CAD model repositories". In: *Advanced Engineering Informatics* 53 (2022), p. 101680.

[Buc+21] Georg Buchgeher, David Gabauer, Jorge Martinez-Gil, and Lisa Ehrlinger. "Knowledge graphs in manufacturing and production: A systematic literature review". In: *IEEE Access* 9 (2021), pp. 55537–55554.

[BX96] Randolph E Bank and Jinchao Xu. "An algorithm for coarsening unstructured meshes". In: *Numerische Mathematik* 73.1 (1996), pp. 1–36.

[Cen15] Center for Collision Safety and Analysis. *Toyota Yaris finite element model validation detail mesh*. 2015. URL: `https://www.ccsa.gmu.edu/models/2009-toyota-yaris/` (visited on 03/26/2020).

[Che+17] Yu Fan Chen, Miao Liu, Michael Everett, and Jonathan P How. "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 285–292.

[Che+18] Wei Chen, Fangzhou Guo, Dongming Han, Jacheng Pan, Xiaotao Nie, Jiazhi Xia, and Xiaolong Zhang. "Structure-based suggestive exploration: a new approach for effective exploration of large networks". In: *IEEE transactions on visualization and computer graphics* 25.1 (2018), pp. 555–565.

[Che+19] Yi Chen, Zeli Guan, Rong Zhang, Xiaomin Du, and Yunhai Wang. "A survey on visualization approaches for exploring association relationships in graph data". In: *Journal of Visualization* 22.3 (2019), pp. 625–639.

[CMS04] Anoop Chawla, Sudipto Mukherjee, and Aneesh Sharma. "Mesh generation for folded airbags". In: *Computer-Aided Design and Applications* 1.1-4 (2004), pp. 269–276.

[CS20] Se-Hang Cheong and Yain-Whar Si. "Force-directed algorithms for schematic drawings and placement: A survey". In: *Information Visualization* 19.1 (2020), pp. 65–91.

[DB+04] Paul Du Bois, Clifford C Chou, Bahig B Fileta, Albert I King, and Hikmat F Mahmood. "Vehicle crashworthiness and occupant protection". In: *AISI* (2004), pp. 27–280, 304–330.

[DC20] Taşkın Dirsehan and Ceren Can. "Examination of trust and sustainability concerns in autonomous vehicle adoption". In: *Technology in Society* 63 (2020), p. 101361.

[DZ18] Xianping Du and Feng Zhu. "A new data-driven design methodology for mechanical systems with high dimensional design variables". In: *Advances in Engineering Software* 117 (November 2017 2018), pp. 18–28. DOI: 10.1016/j.advengsoft.2017.12.006.

[DZ19] Xianping Du and Feng Zhu. "A novel principal components analysis (PCA) method for energy absorbing structural design enhanced by data mining". In: *Advances in Engineering Software* 127 (September 2018 2019). using pca to generate simulation, pp. 17–27. DOI: 10.1016/j.advengsoft.2018.10.005.

[EI17] Iredia Davis Erhunmwun and U. B. Ikponmwosa. "Review on finite element method". In: *Journal of Applied Sciences and Environmental Management* 21 (2017), pp. 999–1002.

[Eura] European Union's Horizon research. *OSCAR-2020*. https://www.osccarproject.eu/. (Visited on 10/02/2023).

[Eurb] European Union's Horizon research. *VIRTUAL-2020*. https://projectvirtual.eu/. (Visited on 10/02/2023).

[Far70] Iraj Farhoomand. *Nonlinear dynamic stress analysis of two-dimensional solids*. University of California, Berkeley, 1970.

[FR91] Thomas M. J Fruchterman and Edward M. Reingold. "Graph drawing by force-directed placement". In: *Force-Directed Placement, in: Software-Practice and Experience* (1991), 21no11pp1129–1164.

[FS20] Naouress Fatfouta and Julie Stal-Le Cardinal. "Towards a framework for integrated and collaborative knowledge management for engineering design – a case study". In: *Proceedings of the Design Society: DESIGN Conference* 1 (May 2020), pp. 559–568. DOI: 10.1017/dsd.2020.136.

[FSR19] Naouress Fatfouta, Julie Stal-Le Cardinal, and Christine Royer. "Empirical study of car crash simulation analysis within the development phase". In: *Proceedings of*

the Design Society: DESIGN Conference 2019-Augus (2019), pp. 2843–2852. DOI: 10.1017/dsi.2019.291.

[FV20] Giuseppe Futia and Antonio Vetrò. "On the integration of knowledge graphs into deep learning models for a more comprehensible AI—Three challenges for future research". In: *Information* 11.2 (2020), p. 122.

[FZL20] Qingqing Feng, Xionghui Zhou, and Junjie Li. "A hybrid and automated approach to adapt geometry model for CAD/CAE integration". In: *Engineering with Computers* 36 (2020), pp. 543–563. DOI: 10.1007/s00366-019-00713-4.

[Gar+22] Jochen Garcke, Rodrigo Iza-Teran, Mandar Pathare, Daniela Steffes-lai, and Pit Schwanitz. "Identifying similarities and exceptions in deformations and mesh functions". In: *SIMVEC, Baden-Baden*. 2022.

[GB07] Peter J. Gawthrop and Graham Bevan. "Bond-graph modeling". In: *IEEE Control Systems Magazine* 27.2 (2007), pp. 24–45.

[GG16] José J Granda and Toby Gloekler. "Bond graph models for reconstruction of vehicle barrier equivalent speeds". In: *Proceedings of the International Conference on Bond Graph Modeling and Simulation*. 2016, pp. 35–47.

[GM96] Kajal K. Gupta and John L. Meek. "A brief history of the beginning of the finite element method". In: *International journal for numerical methods in engineering* 39.22 (1996), pp. 3761–3774.

[GPP17] Jochen Garcke, Mandar Pathare, and Nikhil Prabakaran. "ModelCompare". In: *Scientific Computing and Algorithms in Industrial Simulations*. Springer, 2017, pp. 199–205.

[Gra11] Jose J Granda. "Automating the process for modeling and simulation of mechatronics systems". In: *Bond Graph Modelling of Engineering Systems*. Springer, 2011, pp. 385–430.

[Gro+23] Paul Groth, Elena Simperl, Marieke van Erp, and Denny Vrandečić. "Knowledge graphs and their role in the knowledge engineering of the 21st century". In: *Dagstuhl Reports*. Vol. 12. 9. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2023.

[Häg+10] J Hägele, U Hänle, A Kropp, M Streit, C Kerner, and M Schlenkrich. "The CAE-Bench project–A web-based system for data, documentation and information to improve simulation processes". In: *Technical paper, BMW Group, Silicon Graphics GmbH, Munich, Germany* (2010).

[Hau83] Eberhard Haug. "Static and dynamic finite element analysis of structural crashworthiness in the automotive and aerospace industries". In: *Structural Crashworthiness* (1983).

[HDY23] Alon Halevy and Jane Dwivedi-Yu. "Learnings from data integration for augmented language models". In: *arXiv preprint , id: 2304.04576* (2023).

[Hit21] Pascal Hitzler. "A review of the semantic web field". In: *Communications of the ACM* 64.2 (2021), pp. 76–83.

[Hog+21] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. "Knowledge graphs". In: *ACM Computing Surveys (CSUR)* 54.4 (2021), pp. 1–37.

[HSSC08] Aric Hagberg, Pieter Swart, and Daniel S Chult. *Exploring network structure, dynamics, and function using NetworkX*. Tech. rep. Los Alamos National Lab, 2008.

[Hue+21] Armand Huet, Romain Pinquié, Philippe Véron, Antoine Mallet, and Frédéric Segonds. "CACDA: A knowledge graph for a context-aware cognitive design assistant". In: *Computers in Industry* 125 (2021), p. 103377.

[ITG19] Rodrigo Iza-Teran and Jochen Garcke. "A geometrical method for low-dimensional representations of simulations". In: *SIAM-ASA Journal on Uncertainty Quantification*

7.2 (2019), pp. 472–496. ISSN: 21662525. DOI: 10.1137/17M1154205.

[Jac+14] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. "ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software". In: *PloS one* 9.6 (2014), e98679.

[Ji+21] Shaoxiong Ji, Shirui Pan, Erik Camburlria, Pekka Marttinen, and S Yu Philip. "A survey on knowledge graphs: Representation, acquisition, and applications". In: *IEEE transactions on neural networks and learning systems* 33.2 (2021), pp. 494–514.

[Joh+18] Joel Johansson, Manuel Contero, Pedro Company, and Fredrik Elgh. "Supporting connectivism in knowledge based engineering with graph theory, filtering techniques and model quality assurance". In: *Adv. Eng. Informatics* 38 (2018), pp. 252–263. DOI: 10.1016/j.aei.2018.07.005.

[JW02] Glen Jeh and Jennifer Widom. "SimRank: A measure of structural-context similarity". In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2002, pp. 538–543.

[Kam+21] Sachin S Kamble, Amine Belhadi, Angappa Gunasekaran, L Ganapathy, and Surabhi Verma. "A large multi-group decision-making technique for prioritizing the big data-driven circular economy practices in the automobile component manufacturing industry". In: *Technological Forecasting and Social Change* 165 (2021), p. 120567.

[Kej19] Mayank Kejriwal. *Domain-specific knowledge graph construction*. 2019.

[Ker99] Boris S Kerner. "The physics of traffic". In: *Physics World* 12.8 (1999), p. 25.

[Kes+19] Philipp Kestel, Patricia Kügler, Christoph Zirngibl, Benjamin Schleich, and Sandro Wartzack. "Ontology-based

approach for the provision of simulation knowledge acquired by Data and Text Mining processes". In: *Advanced Engineering Informatics* 39 (2019), pp. 292–305.

[KF18] Megan Katsumi and Mark Fox. "Ontologies for transportation research: A survey". In: *Transportation Research Part C: Emerging Technologies* 89 (2018), pp. 53–82.

[KK89] Tomihisa Kamada and Satoru Kawai. "An algorithm for drawing general undirected graphs Tomihisa Kamada and Satoru Kawai". In: *Information Processing Letters* 31.1 (1989), pp. 7–15.

[Klo+18a] Benjamin Klotz, Raphaël Troncy, Daniel Wilms, and Christian Bonnet. "Generating semantic trajectories using a car signal ontology". In: *Companion Proceedings of the The Web Conference 2018*. 2018, pp. 135–138.

[Klo+18b] Benjamin Klotz, Raphaël Troncy, Daniel Wilms, and Christian Bonnet. "VSSo: A vehicle signal and attribute ontology (short paper)". In: *SSN Workshop at ISWC. CEUR Workshop Proceedings*. 2018.

[Kra+22] David Kracker, Revan Kumar Dhanasekaran, Axel Schumacher, and Jochen Garcke. "Method for automated detection of outliers in crash simulations". In: *International Journal of Crashworthiness* (2022), pp. 1–12.

[KS18] Robert Kirkwood and James A. Sherwood. "Sustained CAD/CAE integration: integrating with successive versions of step or IGES files". In: *Engineering with Computers* 34 (1 Jan. 2018). DOI: 10.1007/s00366-017-0516-z.

[Kum+20] Ajay Kumar, Shashank Sheshar Singh, Kuldeep Singh, and Bhaskar Biswas. "Link prediction techniques, applications, and performance: A survey". In: *Physica A: Statistical Mechanics and its Applications* 553 (2020), p. 124289.

[Lan+19] Volker A Lange, Johannes Fender, Lailong Song, and Fabian Duddeck. "Early phase modeling of frontal impacts for crashworthiness: from lumped mass–spring models to deformation space models". In: *Proceedings of the Institution*

of Mechanical Engineers, Part D: Journal of automobile engineering 233.12 (2019), pp. 3000–3015.

[Lee+20] Woojin Lee, Donghee Kim, Yeongdo Park, and Kang Yul Huh. "Development of a web-based open source cae platform for simulation of IC engines". In: *International Journal of Automotive Technology* 21.1 (2020), pp. 169–179.

[LHB03] Mats Lindquist, Anthony Hall, and Ulf Björnstig. "Real world car crash investigations–A new approach". In: *International Journal of Crashworthiness* 8.4 (2003), pp. 375–384.

[Li+20] Ying Li, Vitalii Zakhozhyi, Daniel Zhu, and Luis J Salazar. "Domain specific knowledge graphs as a service to the public: powering social-impact funding in the US". In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 2793–2801.

[Li+21] Xinyu Li, Mengtao Lyu, Zuoxu Wang, Chun-Hsien Chen, and Pai Zheng. "Exploiting knowledge graphs in industrial products and services: a survey of key aspects, challenges, and future perspectives". In: *Computers in Industry* 129 (2021), p. 103449.

[Lim17] Jae Moon Lim. "Lumped mass-spring model construction for crash analysis using full frontal impact test data". In: *International Journal of Automotive Technology* 18 (2017), pp. 463–472.

[LV07] John A. Lee and Michel Verleysen. *Nonlinear Dimensionality Reduction*. Springer, 2007.

[Ma+22] Nan Ma, Zhixuan Wu, Yiu-ming Cheung, Yuchen Guo, Yue Gao, Jiahong Li, and Beijyan Jiang. "A Survey of human action recognition and posture prediction". In: *Tsinghua Science and Technology* 27.6 (2022), pp. 973–1001.

[Men+18] T Menzel, G Bagschik, L Isensee, A Schomburg, and M Maurer. "Detailing a keyword based scenario description for execution in a simulation environment using the example of scenarios on German highways". In: *Workshop*

Fahrerassistenzsysteme und automatisiertes Fahren. Vol. 12. 2018, pp. 15–26.

[Mey+20] Morten Meyer, Zhuo Yu, Priyanka Gulati, Ahmad Delforouzi, Josef Roggenbuck, and Klaus Wolf. "Ontologies for digital twins in smart manufacturing". In: (2020). URL: http://doi.org/10.13140/RG.2.2.11346.17607 (visited on 10/01/2023).

[Mia+23] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. "Augmented language models: A survey". In: *arXiv preprint id:2302.07842* (2023).

[Mil13] Justin J Miller. "Graph database applications and concepts with Neo4j". In: *Proceedings of the southern association for information systems conference, Atlanta, GA, USA*. Vol. 2324. 36. 2013.

[ML97] Marc H Meyer and Alvin P Lehnerd. *The power of product platforms*. Simon and Schuster, 1997.

[Mon+17] Filippo Montevecchi, Giuseppe Venturini, Niccolò Grossi, Antonio Scippa, and Gianni Campatelli. "Finite element mesh coarsening for effective distortion prediction in wire arc additive manufacturing". In: *Additive Manufacturing* 18 (2017), pp. 145–155.

[MRH92] Stuart G Mentzer, Randa A Radwan, and William T Hollowell. *The SISAME methodology for extraction of optimal lumped parameter structural crash models*. Tech. rep. SAE Technical Paper, 1992.

[MSS+03] Andreas Maier, Hans-Peter Schnurr, York Sure, et al. "Ontology-based information integration in the automotive industry". In: *ISWC*. Vol. 2870. Springer. 2003, pp. 897–912.

[MT08] Liquan Mei and C. A. Thole. "Data analysis for parallel car-crash simulation results and model optimization". In:

Simul. Model. Pract. Theory 16.3 (2008), pp. 329–337. DOI: 10.1016/j.simpat.2007.11.018.

[Muz+22] Abu Jafar Md Muzahid, Syafiq Fauzi Kamarulzaman, Md Arafatur Rahman, and Ali H Alenezi. "Deep reinforcement learning-based driving strategy for avoidance of chain collisions and its safety efficiency analysis in autonomous vehicles". In: *IEEE Access* 10 (2022), pp. 43303–43319.

[MV09] Kun Marhadi and Satchi Venkataraman. "Comparison of quantitative and qualitative information provided by different structural load path definitions". In: *International Journal for Simulation and Multidisciplinary Design Optimization* 3.3 (2009), pp. 384–400.

[MVH+04] Deborah L McGuinness, Frank Van Harmelen, et al. "OWL web ontology language overview". In: *W3C recommendation* 10.10 (2004), p. 2004.

[New05] Mark EJ Newman. "Power laws, Pareto distributions and Zipf's law". In: *Contemporary physics* 46.5 (2005), pp. 323–351.

[New59] Nathan M Newmark. "A method of computation for structural dynamics". In: *Journal of the engineering mechanics division* 85.3 (1959), pp. 67–94.

[Ngu21] Tuong H. Nguyen. *5 impactful emerging technologies for 2022*. 2021.

[NM+01] Natalya F Noy, Deborah L McGuinness, et al. *Ontology development 101: A guide to creating your first ontology*. 2001.

[Ope22] OpenPASS Working Group. *CommonRoad Scenario Designer*. Version 0.9.0. `https://openpass.eclipse.org/`. 2022.

[OS13] Christopher Ortmann and Axel Schumacher. "Graph and heuristic based topology optimization of crash loaded structures". In: *Structural and Multidisciplinary Optimization* 47.6 (2013), pp. 839–854.

[PG22] Anahita Pakiman and Jochen Garcke. "Graph modeling in computer assisted automotive development". In: *2022*

IEEE International Conference on Knowledge Graph (ICKG). 2022, pp. 203–210. DOI: 10.1109/ICKG55886.2022.00033.

[PGS22] Anahita Pakiman, Jochen Garcke, and Axel Schumacher. "SimRank-based prediction of crash simulation similarities". In: *Applied Intelligence* (2022). Under review, INS Preprint No. 2210, Institut für Numerische Simulation, Universität Bonn.

[PGS23a] Anahita Pakiman, Jochen Garcke, and Axel Schumacher. "Graph extraction for assisting crash simulation data analysis". In: *Graph-Based Representation and Reasoning*. Accepted. Springer Nature Switzerland, 2023, pp. 171–185. ISBN: 978-3-031-40960-8.

[PGS23b] Anahita Pakiman, Jochen Garcke, and Axel Schumacher. "Knowledge discovery assistants for crash simulations with graph algorithms and energy absorption features". In: *Applied Intelligence* (2023), pp. 1–20.

[PJ21] Archana Patel and Sarika Jain. "Present and future of semantic web technologies: a research statement". In: *International Journal of Computers and Applications* 43.5 (2021), pp. 413–422.

[Qia+18] Lishan Qiao, Limei Zhang, Songcan Chen, and Dinggang Shen. "Data-driven graph construction and graph learning: A review". In: *Neurocomputing* 312 (2018), pp. 336–351.

[Rex20] Rex Ying, Jiaxuan You, Zecheng Zhang, Xinwei He, Rok Sosic, Jure Leskovec. *DeepSNAP*. 2020.

[Sch+22] P. Schwanitz, L. Greve, F. Moldering, and et.al. "Towards AI Based REecommendations for Design Improvement (AI-B-REDI)– A joint research project of Volkswagen, Audi, Porsche, Fraunhofer SCAI, SCALE GmbH and SIDACT GmbH". In: *SIMVEC, Baden-Baden*. 2022.

[Sch22] Pit Schwanitz. "Towards AI Based Recommendations for Design Improvement (AI-B-REDI)". presentation at SIMVEC, Baden-Baden. 2022.

[Sch73] Edward W Schneider. "Course modularization applied: the interface system and its implications for sequence control and data analysis." In: (1973).

[SD20] Björn Schembera and Juan M Durán. "Dark data as the new challenge for big data science and the introduction of the scientific data officer". In: *Philosophy & Technology* 33.1 (2020), pp. 93–115.

[SlPG21] Daniela Steffes-lai, Mandar Pathare, and Jochen Garcke. "Towards a framework for automatic event detection for car crash simulations". In: *NAFEMS World Congress 2021*. 2021.

[SN13] Naoki Sugiyama and Takashi Nagatani. "Multiple-vehicle collision in traffic flow by a sudden slowdown". In: *Physica A: Statistical Mechanics and its Applications* 392.8 (2013), pp. 1848–1857.

[SS17] Dominik Schneider and Axel Schumacher. "Finding optimized layouts for ribs on surfaces using the graph and heuristic based topology optimization". In: *World Congress of Structural and Multidisciplinary Optimisation*. Springer. 2017, pp. 1615–1628.

[SSS18] K Srinivasa, G Siddesh, and H Srinidhi. "Introduction to data visualization". In: *Network data analytics*. Springer, 2018, pp. 321–331.

[Sta+20] Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li, and Anthony G. Constantinides. "Data analytics on graphs part III: Machine learning on graphs, from graph topology to applications". In: *Foundations and Trends in Machine Learning* 13.4 (2020), pp. 332–530. ISSN: 1935-8237. DOI: 10.1561/2200000078-3.

[Tha20] Shramana Thakur. "Modelling and analysis of automobile simulations using knowledge graphs". MA thesis. Fraunhofer SCAI-Universität Bonn, 2020.

[TT21] Si Ying Tan and Araz Taeihagh. "Adaptive governance of autonomous vehicles: Accelerating the adoption of disruptive technologies in Singapore". In: *Government Information Quarterly* 38.2 (2021).

[Urb+21] Itziar Urbieta, Marcos Nieto, Mikel García, and Oihana Otaegui. "Design and implementation of an ontology for semantic labeling and testing: Automotive Global Ontology (AGO)". In: *Applied Sciences* 11.17 (2021), p. 7782.

[VR+16] Michiel Van Ratingen, Aled Williams, Lie Anders, Andre Seeck, Pierre Castaing, Reinhard Kolke, Guido Adriaenssens, and Andrew Miller. "The European new car assessment programme: a historical review". In: *Chinese journal of traumatology* 19.02 (2016), pp. 63–69.

[Vra23] Denny Vrandečić. "Keynote: The future of knowledge graphs in a world of large language models". The Knowledge Graph Conference. 2023. URL: `https://www.knowledgegraph.tech/speakers/denny-vrandecic-2/,https://www.youtube.com/watch?v=WqYBx2gB6vA&t=1059s` (visited on 10/01/2023).

[Wal+23] Joe Walsh, Scott Shaw, Rod Dreisbach, William Schindel, Laura Michalske, and David Leal. "Unified model characteristics for engineering simulation draft 4.0". In: NAFEMS. 2023.

[Wan+20] Yue Wang, Zhe Wang, Ziyuan Zhao, Zijian Li, Xun Jian, Hao Xin, Lei Chen, Jianchun Song, Zhenhong Chen, and Meng Zhao. "Effective similarity search on heterogeneous networks: A meta-path free approach". In: *IEEE Transactions on Knowledge and Data Engineering* (2020).

[Wan+21] Hanzhi Wang, Zhewei Wei, Yu Liu, Ye Yuan, Xiaoyong Du, and Ji-Rong Wen. "ExactSim: benchmarking single-source SimRank algorithms with high-precision ground truths". In: *The VLDB Journal* 30.6 (2021), pp. 989–1015.

[WC92] Steven C Wheelwright and Kim B Clark. *Revolutionizing product development: quantum leaps in speed, efficiency, and quality*. Simon and Schuster, 1992.

[Wel+19] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. "Neural text generation with unlikelihood training". In: *arXiv preprint , id: 1908.04319* (2019).

[Zie+20] Julian Ziegler, Peter Reimann, Florian Keller, and Bernhard Mitschang. "A graph-based approach to manage cae data in a data lake". In: *Procedia CIRP* 93 (2020), pp. 496–501.