Escaping the Tsunami: Evacuation Strategies for Large Urban Areas

Concepts and Implementation of a Multi-Agent Based Approach

D I S S E R T A T I O N

vorgelegt von
Diplom-Informatiker Gregor Lämmel
aus Erlabrunn

An der Fakultät V – Verkehrs- und Maschinensysteme
der Technischen Universität Berlin
zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr.-Ing. Gerd Holbach
Berichter: Prof. Dr. rer. nat. Kai Nagel
Berichterin: Assoc. Prof. Dr. habil. rer. nat. Franziska Klügl

Tag der wissenschaftlichen Aussprache: 08.09.2011
Berlin 2011

D83

Abstract: The evacuation of whole cities or even regions represents a complex problem for transport planning. The urgency and actuality of this complex problem was demonstrated recently by such events as the "Queensland flooding" in Australia, where parts of Brisbane had to be evacuated or the the evacuation of large areas in northeast Japan after th 9.0 megathrust earthquake followed by a devastating tsunami and a subsequent breakdown at the Fukushima Daiichi nuclear power plant. Congruent with the importance of the topic, there is a large body of research regarding emergency evacuations. Many dynamic aspects of an evacuation such as congestion can be handled adequately only if the evacuation process is modeled on a microscopic level. This true for the large-scale scenarios as well. However, most existing models are either not microscopic, or not capable to deal with large scenarios.

This thesis discusses an approach that deals with large-scale evacuations on the microscopic level. Whereas existing models tend to neglect essential aspects like the time-dependent expansion of a hazard, a comprehensive evacuation simulation framework has been developed in order to consider such aspects. During the development of the simulation framework several subproblems of the main evacuation problem have been identified:

- Evacuation routing – The routing problem is to find an appropriate evacuation route for every single evacuee. A straightforward solution is the shortest path solution, where every evacuee takes the shortest path to the safe area. In addition evacuation routes can also be assigned to reduce individual travel times (Nash equilibrium approach) or to reduce the system travel time (marginal social cost based approach). The marginal social costs are the travel costs (travel time) that one additional evacuee imposes on the system. It is the sum of the travel time experienced by the additional evacuee and the additional travel times experienced by all others because of the additional evacuee.

- Time-dependent aspects of the danger – The flooding will not cover the entire hazard zone at once, meaning that while some districts of the city are already inundated, other areas may still be passable. This aspect has to be considered when developing evacuation strategies.

- Risk reduction – The objective of a risk reducing evacuation strategy is to find routes that avoid unnecessary risk.

- Shelter assignment – Shelters are safe places with limited space capacity inside the evacuation area. Problems that arise when it comes to shelters are: where to place them (allocation), which size they must have (capacity problem) and who should be allowed to use them (assignment problem).

These sub-problems are investigated both in theory and experiments. In the theoretical part of this thesis, the evacuation problem is treated as a dynamic network flow problem with time-dependent link travel times. The network constitutes an abstraction of a road network. The evacuees are modeled as agents traveling from their respective starting location to the safe area. The objective is to find a set of individual evacuation routes subjected to the different sub-problems. Approaches to solve the evacuation sub-problems are based on iterative learning algorithms. The approaches proposed include but are not limited to: A Nash equilibrium routing approach, a marginal social cost based routing approach and an approach that solves the shelter assignment problem.

The proposed approaches are implemented as a set of extensions to the MATSim framework, which is discussed in the practical part of this work. MATSim stands for Multi-Agent Transport Simulation and provides a toolbox to implement large-scale agent based transport simulations. The performance of the iterative learning framework to solve the evacuation problem is tested on a real-world scenario for the city of Padang.

Padang is located on the Mentawai segment at the West Coast of Sumatra, Indonesia. Padang is a low-lying (less than 10 m above sea level) city with approximately 850 000 inhabitants and is characterized by its net of urban waterways. The West Coast of Sumatra is a region of high tectonic activity and has been hit by tsunamis in the past. The city has been indicated as one of the most plausible locations for a tsunami of disastrous proportions in near future. An evacuation of the city in case of a tsunami is particularly complicated not only because of the hundreds of thousands evacuees but also because of the dense net of urban waterways which are barriers hindering the evacuees from reaching the safe hinterland.

Simulation runs have been performed based on the Padang scenario for each of the sub-problems. Detailed analyses of the results are given for each simulation run.

There are several important findings that are obtained from the simulation results:

- The shortest path solution, being a straightforward one, is not suitable for the evacuation planning. The reason for this is that the shortest path solution does not consider congestion effects and therefore tends to underestimate the travel times.

- Other routing strategies like the Nash equilibrium approach or the marginal social cost based approach are considering congestion effects and therefore leading to better evacuation results. However, as long as time-dependent aspects of the hazard are not explicitly modeled, those solutions are also unsuitable. The experimental results for the Padang scenario show that without considering the approaching tsunami some agents tend to chose evacuation routes that are parallel to the coastline. The reason is that for those agents the nearest safe location is not somewhere in the hinterland but instead it is next to the coast. In reality, however, it would be recommended to evacuate away from the coast. For evacuation modeling this means the time-dependent aspects of the hazard have to be considered explicitly.

- Usually there are a lot of uncertain factors when it comes to evacuations. One uncertain factor is the advance warning time. The risk that not all evacuees manage to escape increases with the uncertainty in the advanced warning time. Risk should be explicitly modeled, which calls for a risk reducing evacuation strategy.

- Even if the time-dependent aspects and risks are explicitly considered by the model, situations are still possible, when the available time would not suffice for the evacuation of all persons to the safe hinterland. In those situations safe places (so-called shelters) can be build inside the evacuation area. However, the locations for the shelters have to be considered carefully because a shelter at the wrong location could also worsen the situation.

The approaches introduced in this thesis are tested with MATSim specifically on the Padang scenario. However, the learning approaches are developed based on abstract algorithms and, therefore, they should be applicable to other simulation frameworks with moderate efforts. Furthermore, MATSim as a flexible open source simulation framework gives the opportunity to apply to other scenarios as well.

Zusammenfassung: Die Evakuierung von Städten oder sogar ganzen Regionen ist eine große Herausforderung. Das hat sich auch kürzlich gezeigt bei der Evakuierung von Stadtteilen der Stadt Brisbane in Australien im Bundestaat Queensland oder auch bei der Evakuierung großer Gebiete im Nord-Osten Japans nach dem Erdbeben der Stärke 9 und dem darauf folgenden Tsunami, welcher wiederum einen nuklearen Störfall im Fukushima Daiichi Atomkraftwerk auslöste.

Nicht zuletzt aufgrund der Bedeutung dieser Thematik gibt es bereits sehr viele Forschungsergebnisse auf dem Gebiet der Notfallevakuierung. Viele dynamischen Aspekte einer Evakuierung, wie z.B. entstehender Stau, kann nur adäquat erfasst werden wenn der Evakuierungsprozess auf einer mikroskopischen Ebene modelliert wird.

Die meisten existierenden Modelle sind entweder nicht mikroskopisch oder nicht in der Lage mit großen Szenarien umzugehen. Diese Arbeit präsentiert einen Ansatz, der sowohl mikroskopisch ist als auch mit großen Szenarien umgehen kann. Des weiteren werden in den meisten existierenden Modellen wichtige Aspekte, wie z.B. die zeitabhängige Ausbreitung der Bedrohung, nicht abgebildet. Aus diesem Grund wird ein umfassendes Simulations-Framework erforscht.

Das Evakuierungsproblem wird zunächst in mehrere Teilprobleme zerlegt.

- Evakuierungsstreckenführung: Das Streckenführungsproblem besteht darin einen passenden Evakuierungspfad für alle Personen im Evakuierungsgebiet zu finden. Eine einfache Lösung ist die kürzeste Wege Lösung, wo jede Person den kürzesten Weg nimmt. Jedoch können Evakuierungspfade auch ausgesucht werden, um die individuellen Reisezeiten zu minimieren (Nash-Gleichgewicht Ansatz) oder um die Systemreisezeit zu minimieren (marginaler sozialer Kosten Ansatz). Die marginalen sozialen Kosten sind die Kosten, die eine zusätzliche Person verursachen würde. Es ist die Summe der Reisezeit, erfahren durch die zusätzliche Person und die zusätzlichen Reisezeiten, welche von allen anderen erfahren werden, wegen der zusätzlichen Person.

- Zeitabhängige Ausbreitung des Gefährdungsgebiets: Eine Überflutung wird ein Gefahrengebiet nicht sofort komplett bedecken, d.h. während manche Gebiete bereits überflutet sind können andere noch durchquert werden. Dieser Aspekt muss bei der Entwicklung von Evakuierungsstrategien berücksichtigt werden.

- Risiko Reduzierung: Das Ziel einer risikoreduzierenden Evakuierungsstrategie ist es, Evakuierungswege zu finden, die unnötiges Risiko vermeiden.

- Fluchtburgen: Fluchtburgen sind sichere Orte mit begrenzter Kapazität innerhalb des Evakuierungsgebiets. Durch die Hinzunahme von Fluchtburgen treten folgende Fragen auf: Wo sollen die Fluchtburgen errichtet werden (Verteilungsproblem), welche Größe müssen sie haben (Kapazitätsproblem) und wer darf in die Fluchtburgen fliehen (Zuordnungsproblem).

Diese Unterprobleme werden in der Arbeit theoretisch und praktisch untersucht. Im theoretischen Teil wird das Evakuierungsproblem als ein dynamisches Netzwerkflussproblem mit zeitabhängigen Kantenreisezeiten betrachtet. Dabei stellt das Netzwerk eine Abstraktion eines Straßennetzes dar. Die Flüchtlinge sind als Agenten modelliert, welche von ihrem jeweiligen Startort in sichere Gebiet fliehen. Das Ziel ist es, eine Menge von individuellen Evakuierungspfaden zu finden unter Berücksichtigung der unterschiedlichen Teilprobleme. Ansätze, um die Teilproblem zu lösen, basieren auf iterativen Lernverfahren. Die vorgeschlagen Ansätze beinhalten unter anderem: Ein Nash-Gleichgewicht Ansatz, ein marginaler sozialer Kosten basierender Ansatz und ein Verfahren um das Schutzburgenzuordnungsproblem zu lösen.

Im praktischen Teil dieser Arbeit werden die vorgeschlagen Ansätze als einen Menge von Erweiterung für das MATSim Framework erforscht. Der Name MATSim steht für Multi-Agenten Transport Simulation. MATSim ist ein Werkzeug, um großskalige agentenbasierte Verkehrssimulationen zu erstellen. Die auf dem iterativen Lernverfahren basierenden Lösungsansätze werden in der Arbeit an einem realistischen Szenario für die Stadt Padang untersucht.

Padang befindet sich auf dem Mentawai Segment an der Westküste von Sumatra in Indonesien. Die Stadt hat etwa 850 000 Einwohner und ist von einem Netz von Kanälen und Flüssen durchzogen. Padang liegt in einer Region hoher tektonischer Aktivität und wurde bereits in der Vergangenheit von Tsunamis überflutet. Es wird erwartet, dass die Stadt in absehbarer Zukunft von einem gewaltigen Tsunami betroffen sein wird. Eine Evakuierung der Stadt erscheint nicht nur schwierig wegen den hunderttausenden von Menschen, die betroffen sein werden, sondern auch aufgrund der vielen Flüsse und Kanäle, die im Fall einer Evakuierung Barrieren darstellen, welche die Wege zum sicheren Hinterland versperren.

Für jedes der genannten Teilprobleme wurden Simulationsläufen basierend auf dem Padang Szenario durchgeführt. Für jeden der Simulationsläufe sind Detaillierte Analysen gegeben.

Einige wichtige Erkenntnisse der Simulationsläufe sind:

- Die kürzeste Wege Lösung ist für die Evakuierungsplanung ungeeignet. Das liegt daran, dass die kürzeste Wege Lösung mögliche Staus nicht berücksichtigt und deshalb ist es wahrscheinlich, dass bei dieser Lösung die Reisezeiten unterschätzt werden.

- Besser optimierende Strategien wie der Nash-Gleichgewicht Ansatz und der marginale soziale Kosten basierende Ansatz sind so lange die zeitabhängigen Gegebenheiten der Gefahr nicht berücksichtigt werden auch ungeeignet. Die Experimente für das Padang Szenario zeigen, dass so lange die Überflutung nicht explizit modelliert ist, Flüchtlinge zu Evakuierungspfaden parallel zur Küste tendieren. Es ist jedoch empfehlenswerter sich möglichst von der Küste wegzubewegen.

- In der Regel gibt es eine ganze Reihe von Unsicherheiten im Fall einer Evakuierung. Eine unsichere Gegebenheit ist die zur Verfügung stehende Vorwarnzeit. Es besteht ein Zusammenhang zwischen unsicherer Vorwarnzeit und dem Risiko, dass es nicht alle Flüchtlinge entkommen. Das Risiko sollte explizit modelliert werden, d.h. eine risikoreduzierende Evakuierungsstrategie wird benötigt.

- Auch wenn die zeitabhängigen Aspekte und das Risiko explizit modelliert werden, kann es Situationen geben wo die Zeit für eine Evakuierung ins sichere Hinterland nicht ausreicht. In solchen Situation können sichere Orte (Fluchtburgen) innerhalb des Evakuierungsgebiets errichtet werden. Die Plätze an denen Fluchtburgen errichtet werden, müssen jedoch sorgfälltig ausgewählt werden, da eine Fluchtburg an der falschen Stelle die Situation auch verschlechtern kann.

Die in dieser Dissertation eingeführten Lösungsansätze wurden mit MATSim basierend auf dem Padang Szenario untersucht. Jedoch sind die Lösungsansätze basierend auf abstrakt betrachteten Algorithmen entwickelt wurden und können aus diesem Grund auch für andere Simulations-Frameworks mit mäßigem Aufwand nutzbar gemacht werden. MATSim als ein anpassungsfähiges Open Source Simulations-Framework bietet die Möglichkeit die vorgestellten Lösungsansätze auch auf andere Szenarien anzuwenden.

Acknowledgments

After this long time, there are many people who contributed in some way to this thesis, for which I would like to express my thanks. I am grateful to my supervisor, Kai Nagel, whose encouragement, supervision and support from the preliminary to the concluding level enabled me to finish this work in the first place.
I would also like to thank Franziska Klügl for kindly agreeing to be the second advisor. Her comments and discussions on earlier versions of this work pointed me in the right direction.
There are many other people that I would like to thank for outstanding scientific collaboration, fruitful discussions, great inspirations, software engineering, patience, friendship, trust, and love.

Co-authors in related publications: Daniel Dressler, Gunnar Flötteröd and Hubert Klüpfel
&
The Last-Mile Team: Nils Goseberg, Sven Hebben, Florian Moder, Torsten Schlurmann, Neysa Setiadi and Hannes Taubenböck
&
MATSim developers: Dominik Grether, Marcel Rieser, David Strippgen, Michael Zilske
&
Colleagues: Christoph Dobler, Tyler Frazier, Johannes Illenberger, Benjamin Kickhöfer, Andreas Neumann
&
Administrator of the Computeserver: Norbert Paschedag
&
Other people from TU Berlin: Andrea Stillarius and Margit Ulbrich
&
For proof-reading: Lars Lamprecht
&
Christian, Jakob, Jörn, Matthias, Patricia, René, and Uli
&
my parents Harald and Karla
&
Tanja

This thesis would not have been possible without you.
This research was funded in part by the German Ministry for Education and Research (BMBF) under grants 03G0666E ("Last Mile") and 03NAPI4 ("Advest").

Contents

Chapter 1

Motivation of this work

The evacuation of whole cities or even regions represents a complex problem for traffic planning. The urgency and actuality of this complex problem was demonstrated recently by such events as the "Queensland flooding" in Australia, where parts of Brisbane had to be evacuated or the evacuation of large areas in northeast Japan after the 9.0 megathrust earthquake followed by a devastating tsunami and a subsequent breakdown at the Fukushima Daiichi nuclear power plant. The success of an evacuation depends on many factors. One of these factors is the amount of advance warning time. The sooner a warning can be triggered for the affected area, the more time is available for the evacuation.

In recent years, a lot of work has been done to establish early warning systems for various hazard scenarios. Depending on the hazard, a warning can be triggered hours or even days before the event (hurricanes, inundations of rivers), some hours or minutes before the event (tsunamis) or only a few seconds before the event (earthquakes). Today a hurricane can be predicted with high probability some days before. There are well established web-services (e.g. from the *National Hurricane Center* for the US (http://www.nhc.noaa.gov) or from *Tropical Storm Risk* worldwide (http://www.tropicalstormrisk.com)) where information about approaching hurricanes can be found, meaning normally there is enough time to make a well prepared and organized evacuation. Tsunamis are triggered by earthquakes, volcanic eruptions or massive landslides and therefore not very well predictable. However, if the source of a tsunami is detected soon enough, an advance warning is still possible, but the time would be at most as long as the tsunami's travel time[1]. For that reason an early detection is crucial for the evacuation of the affected areas. When it comes to earthquakes an evacuation of the affected is not possible beforehand. But even for earthquakes a warning is possible (see, e.g., http://www.elarms.org). However, the advance warning time will only be in the range of seconds. Still, this time can be used to switch industrial systems into safe mode and advise persons to take protective measures. An early warning can also help to reduce the probability of derailment

[1] In deep water tsunami waves travel with more than 800 kilometers per hour, meaning it takes them only a few hours to cross entire oceans. For more information see http://en.wikipedia.org/wiki/Tsunami (accessed September 2010).

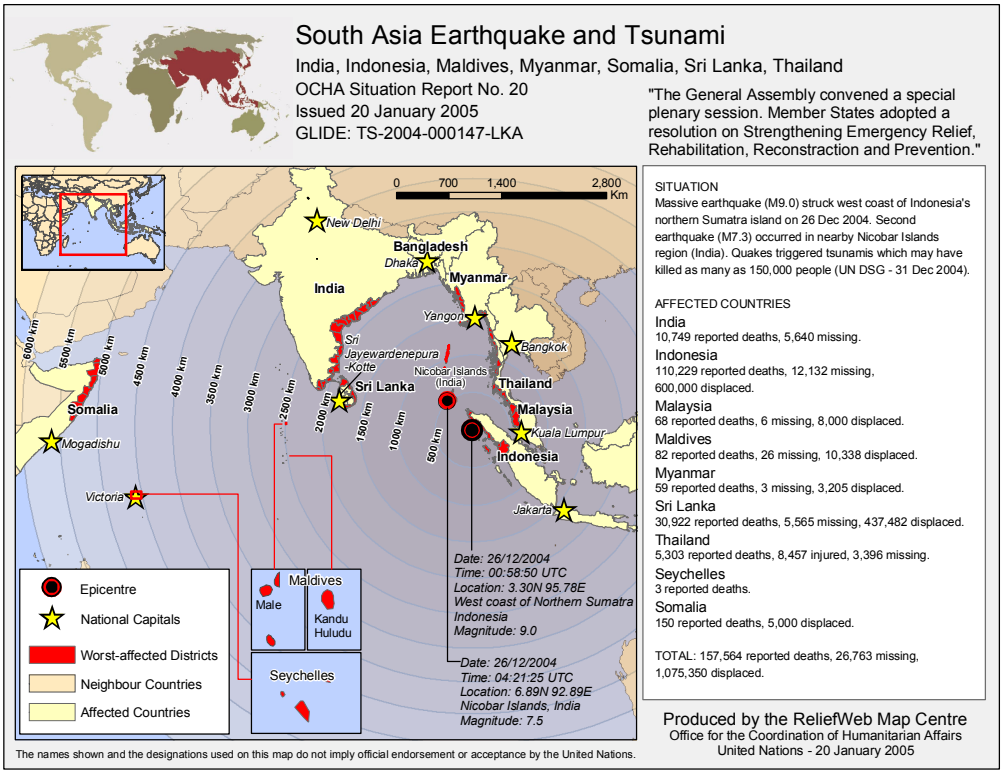


Figure 1.1: Preliminary estimate of the effects of the 2004 Indian Ocean tsunami (source: United Nations Office for the Coordination of Humanitarian Affairs).

of railcars. Even if the time is too short for the rail car to come to a full stop, a slowdown can reduce the risk of derailment drastically. The reason is that there is a quadratic relation between derailment probability and velocity of the rail car (Hohnecker et al., 2010).

The focus of this work is on the second scenario, the evacuation in the case of a tsunami. The most devastating tsunami in recent times was the so called 2004 Indian Ocean tsunami. A map showing the epicenter of the causative earthquake and preliminary estimates of the casualty figures are given in Figure 1.1.

As a reaction to the tsunami threat, the United Nations Organization agreed on a conference held in January 2005 in Kobe, Japan, that an International Early Warning Program was necessary to deal with future comparable natural disasters. The German Government supported the development of such an early warning system resulting in the German Indonesian Tsunami Early Warning System (GITEWS, see http://www.gitews.de for de-

tails). GITEWS is operational since November 2008. However, the question what does the affected communities do with a tsunami warning remains unanswered. To put it in words of Schiermerier (2009): "Five years after the Indian Ocean disaster, the technology is in place, but local preparedness is less advanced". This shortcoming became apparent during the Padang earthquake on September 30, 2009. After this quake an initial tsunami warning had been triggered leading to chaotic scenes on the streets. The warning was later canceled. The quake caused major destruction with more than a 1 000 casualties. A detailed eyewitness report of this disaster is given by Taubenböck et al. (2010). A tsunami can inundate areas that are several kilometers away from the coastline, which has been shown by the 2004 tsunami in Banda Aceh or more recently by the tsunami in northeast Japan. This makes the planning for an evacuation of large areas with possibly hundreds of thousands people within a short period of time necessary. Therefore a detailed evacuation planning is needed beforehand. As a first step the characteristics of an emergency evacuation have to be determined. A general discussion on this matter is given by Barrett et al. (2000). According to the dimensions identified by the authors, basic characteristics of a tsunami evacuation are:

- *Shape and size of the emergency source* – For a tsunami this corresponds to the inundation area. However, the inundation does not cover all the area at once, but it is a highly dynamic component.

- *Shape and size of the evacuation area* – The shape and size of the evacuation area depends on the maximum expansion of the inundation area. However, the actual size may be a lot bigger as the inundation area itself, since if the evacuees stop their evacuation immediately after they left the inundation area they might block others from leaving.

- *Rate of growth of the evacuation area* – The velocity of the water mass decreases with its advance on land, meaning that some area might be flooded seconds after the tsunami wave hits the shoreline while other areas will only be flooded tens of minutes later.

- *Size and makeup of the evacuation population* – An important parameter is the population distribution at the time the tsunami occurs. It is obvious that this depends on the time of day and day of week. Another important parameter is the willingness to evacuate. In situations with a large amount of warning time an evacuation can be enforced externally, e.g. by police. However, in the case of a tsunami the advance warning time can be very short. This would not leave enough time to enforce people to evacuate who are unwilling to do so.

- *Amount of warning time* – The amount of warning time depends strongly on the location of the tsunamigenic earthquake, the capability to detect the quake and its location and the presence of an intact warning system.

- *Level of disruption of the road network* – During the earthquake buildings or bridges can collapse and block evacuation routes. On the other hand, if there is enough time to make an evacuation thinkable, then the tsunamigenic earthquake must be located hundreds of kilometers away, so a total and citywide breakdown of the transportation network is not expected.

- *Level of danger of the emergency* – It seems natural to define the level of danger by the (expected) flooding heights. However, since the mass of water contains a lot of debris also relatively low inundation heights might be fatal. From this point of view the level of danger is the same all over the inundation area. The differences between variant locations, however, are that some parts of the city get flooded earlier than others. As a consequence those parts that get flooded earlier also have to be evacuated earlier. During an evacuation nobody should move towards a location that gets flooded earlier if an alternative escape route exists. In other words nobody should be exposed to avoidable risk. The author recommends introducing the level of risk as an additional characteristic to those introduced by (Barrett et al., 2000).

- *Level of risk* – The level of risk tells how risky it is to take a particular evacuation route. As an example the risk depends on the probability that a bridge that is part of the evacuation route withstands the earthquake. In general, it is hard to quantify risk. However, often it is possible to compare the levels of risk between alternative evacuation strategies. In the bridge example the risk of an evacuation route that avoids the bridge might be less and therefore the bridge-avoiding route is more recommended.

In order to take account of all characteristics a detailed analysis of the so-called last mile in the tsunami evacuation and warning chain needs to be accomplished. This would arguably exceed the work that can be accomplished in a PhD thesis. This calls for a project that deals with the last-mile, where the evacuation modeling is one working package within the project. The Last-Mile research project develops a numerical last-mile tsunami early warning and evacuation information system for the city of Padang (see, e.g., Taubenböck et al. (2009a) for a detailed project description). Padang is located on the Mentawai segment at the West Coast of Sumatra, Indonesia. Padang is a

low-lying (less than $10\,m$ above sea level) city with approximately 850 000 inhabitants and is characterized by its net of urban waterways (see Figure 1.2).

The city is located in a region of high tectonic activity and has been hit by tsunamis in the past. The best documented tsunamis are the ones from 1797 and 1833 (Natawidjaja et al., 2006). The city has been indicated as one of the most plausible locations for a tsunami of disastrous proportions in near future (Borrero et al., 2006). This assumption is still valid even after the 30 September 2009 earthquake with a moment magnitude of $M_w = 7.6$. This recent earthquake did not rupture the (expected) Sunda megathrust and did not significantly relax the 200 years of accumulated the stress on the Mentawai segment (McCloskey et al., 2010). An evacuation of the city in case of a tsunami is particularly complicated not only because of the hundreds of thousands evacuees but also because of the dense net of urban waterways which are barriers hindering the evacuees from reaching the safe hinterland. This makes a detailed evacuation analysis and modeling necessary. Therefore a comprehensive simulation framework to test and optimize different evacuation strategies is needed. When developing evacuation strategies there are three, sometimes conflicting, objectives:

Optimization from the evacuees' perspective: No evacuee will agree to take an obvious detour when heading for a safe place, or to select an obviously faraway safe place instead of a nearby one. This requires identifying evacuation solutions that are fair in that no evacuee can gain by switching to a different solution. This calls for a strategy that results in a Nash equilibrium of all evacuees.

Optimization from a global perspective: It is desirable to evacuate the system as quickly as possible, which is equivalent to minimizing the total evacuation time of the whole population (see 3.1.3). While a Nash strategy has the obvious and important advantage of general acceptance, it may be suboptimal in this regard because some evacuees may do great damage to others by blocking their ways/shelters.

Risk reducing: No evacuee should be exposed to avoidable risky situations during the evacuation. There is a relation between risk and uncertainty. For example it could be risky to recommend a bridge as part of a tsunami evacuation route if it is not certain if the bridge withstand a tsunami. Therefore risky evacuation routes should be avoided as long as non-risky, possibly longer, routes exist.

The aim of this work is to develop a microscopic evacuation simulation and modeling framework that addresses the above discussed objectives. Fur-

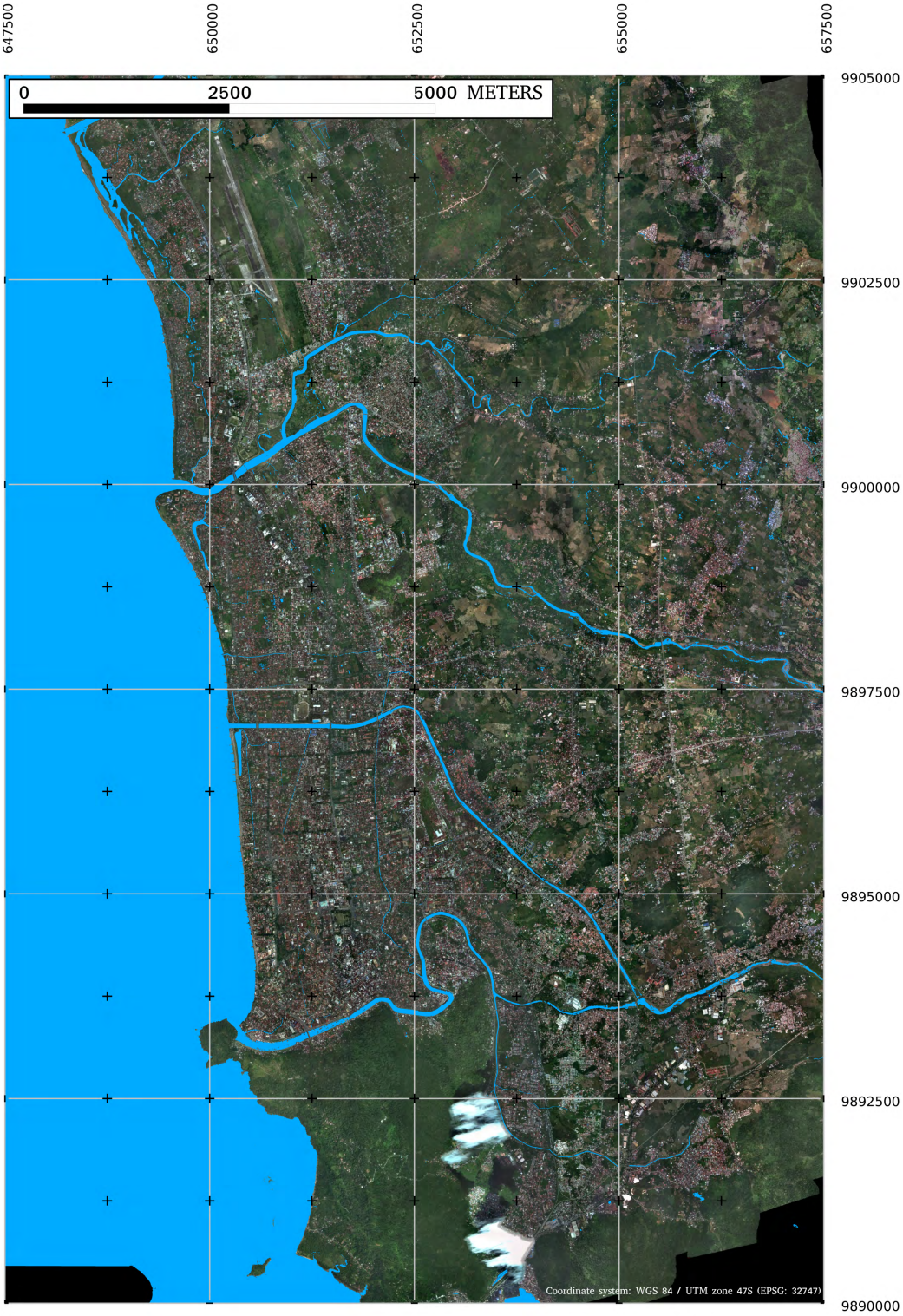


Figure 1.2: Map of Padang.

thermore, different evacuation strategies are discussed within a microscopic mobility simulation including, but not limited to, their applicability to the real world based on the Padang scenario. The simulation framework can serve as a decision support system for authorities, urbanists and disaster agencies by solving the evacuation problem. In reality the evacuation problem consists of a large number of sub-problems. In this work four major problems are addressed:

- *Evacuation routing* – The routing problem is to find an appropriate evacuation route for every evacuee. In general evacuation routes are assigned to reduce individual travel times (Nash routing) or to reduce the system travel time (system optimum).
- *Time-dependent aspects of the danger* – The flooding will not cover all of the danger zone at once, meaning that while some districts of the city are already inundated other districts may still be passable. This aspect has to be considered when developing evacuation strategies.
- *Risk reduction* – The objective of a risk reducing evacuation strategy is to find routes that avoid unnecessary risk.
- *Shelter assignment* – Shelters are safe places with limited space capacity inside the evacuation area (i.e. buildings for vertical evacuation). Problems that arise when it comes to shelters are: where to place them, of which size they must be and who is allowed to use them.

The evacuation simulation framework is implemented as a set of extensions within MATSim (http://www.matsim.org). MATSim stands for **M**ulti-**A**gent **T**ransport **S**imulation and provides a toolbox for implementing large-scale agent-based transport simulations.

The desired solution for the evacuation problem is found by an evolutionary or iterative learning approach, where the evacuees (agents) learn the appropriate behavior by trial and error in the mobility simulation. The works contributes a comprehensive simulation framework that deals with the above discussed problems and is applicable to large scale scenarios.

The remainder of this work is organized as follows. Chapter 2 gives an overview of existing work in the research area of evacuation modeling. Chapter 3 introduces the simulation models that can cope with discussed evacuation sub-problems and discusses some theoretical background as well. A detailed description of MATSim and the necessary extensions that have been made in order to use MATSim for evacuation simulations are discussed in Chapter 4. That chapter also introduces the basic simulation scenario for the city

of Padang and gives simulation results for different routing strategies. Chapter 5 proposes an approach to model the time-dependent aspects of the danger within MATSim. A model for a risk reducing strategy is given in Chapter 6. The performance of the risk reducing strategy is demonstrated by experimental results. Chapter 7 discusses a simulation-based approach to solve the shelter assignment problem. Finally, in Chapter 8, this thesis concludes with a summary of the findings and a discussion of some open issues that will be a matter of future research.

Chapter 2

Related work

2.1 Evolution of evacuation modeling

Congruent with the importance of the topic, there is a large body of research regarding emergency evacuations. As a first classification, one may differentiate between two situations: (i) large-scale citywide or regional evacuations, e.g. because of nuclear power plants failures or because of hurricanes. (ii) evacuation from within buildings, ships, airplanes, etc. Case (i) usually uses vehicular evacuation. Case (ii) usually concerns pedestrian evacuation:

Ongoing research focuses on citywide or regional evacuations, i.e. case (i). The development of these tools was much influenced by the development of tools in the areas of transport planning and traffic management. At the core of many of these methods is a static assignment routine (e.g., (Sheffi, 1985; Ortúzar and Willumsen, 2001)). A typical example for traffic-based evacuation simulation based on static concepts is MASSVAC (Hobeika and Kim, 1998) although later versions contain dynamic aspects.

A shortcoming of static assignment is that it does not possess any consideration of the time-of-day dynamics. In contrast, dynamic traffic assignment (DTA) is defined as a distribution of time-dependent trips on routes. A typical approach to implement DTA is day-to-day re-planning: The traffic flow simulation (also called network loading) is run with pre-specified routes, route costs are extracted, some or all of the routes are modified, the traffic flow simulation is run again, etc., until some stopping criterion is fulfilled. If every trip uses a route which minimizes the expected travel time, then the system is in a (time-dependent) Nash equilibrium. Another way is to select the different route alternatives following a pre-specified distribution function, which leads to a (time-dependent) stochastic user equilibrium (SUE) (Sheffi, 1985; Zhang et al., 2008)).

Many DTA packages have been tested in the evacuation context: MITSIM (Jha et al., 2004), Dynasmart (Kwon and Pitt, 2005; Chiu et al., 2005), PARAMICS (Chen and Zhan, 2004), and VISSIM (Han and Yuan, 2005). Oak Ridge National Laboratory has a package named "OREMS" (see http://www-cta.ornl.gov/cta/One_Pagers/OREMS.pdf – accessed March 2011) explicitly for evacuation traffic. Publications stressing dynamic aspects of traffic-based

evacuation as a novelty can be found with dates as early as 2000 (e.g. Sattayhatewa and Ran, 2000; Barrett et al., 2000). For a review, see Alsnih and Stopher (2004).

Regarding case (ii), during the last two decades the field has changed dramatically. There is a clear tendency towards microscopic simulations. A good overview of pedestrian evacuation modeling and software can be found in the books of the bi-annual conference series "Pedestrian and Evacuation Dynamics" (Schreckenberg and Sharma, 2002; Galea, 2003; Gattermann et al., 2006; Klingsch et al., 2010; Peacock et al., 2011). Pedestrian evacuation simulations can be classified into microscopic and macroscopic ones. Microscopic models represent space, time, and persons on a fine-grained level. Possible microscopic approaches are Cellular Automata (CA) (Klüpfel et al., 2003), discretized differential equations ("molecular dynamics (MD)") (Helbing et al., 2002, 2005), or movement rules based on random utility modeling (Bierlaire et al., 2003). Examples of software packages based on microscopic models are Exodus (Galea, 2002), Myriad (www.crowddynamics.com), Egress (www.aeat-safety-and-risk.com/html/egress.html), and PedGo (Klüpfel, 2006). Macroscopic models use the analogy of flows of pedestrians and liquids. Examples of software packages based on macroscopic models are Aseri (Schneider and Könnecke, 2002) and Simulex (www.iesve.com). See Jafari et al. (2003) and Kuligowski (2004) for surveys. Compared to what is known in terms of field measurements (e.g. Predtetschenski and Milinski, 1978; Weidmann, 1993), most if not all packages lead to similar results (Rogsch, 2005).

A large body of work (e.g. Theodoulou and Wolshon, 2004; Lim and Wolshon, 2005) uses micro-simulation to investigate the issues of contraflow evacuation, i.e. the reversal of inbound lanes of a freeway in order to obtain additional outbound capacity. In contrast, little work seems to exist that specifically deals with evacuation using other modes of transport than walking for the evacuation of confined spaces, or driving for the evacuation of cities or regions. Han (1990) describes mixed evacuation traffic, geared to Taiwanese requirements. Some models for mixed traffic have been developed in recent years. A CA based model which is capable of simulating mixed traffic of motorbikes and cars has been introduced by Lan and Chang (2005). The authors demonstrate the model based on a small scenario with a mix of 150 motorbikes and cars. A model that simulates mixed flow of motorized and non-motorized traffic at a crossing is given by (Wang et al., 2009).

2.2 From reactive models to deliberating agents

Another classification is to distinguish models based on the intelligence of their behavioral entities. There is a wide range of models, from those where the evacuees are modeled as reactive particles without an explicit behavior to multi-agent systems with sophisticated behavioral rules. The choice of the right model generally depends on the purpose of application. Simple models are appropriate for finding feasible solutions to evacuation problems from which explicit recommendations can be derived. Since those models neglect central behavioral aspects like panic or herding behavior, they do not model the real world; instead they give a lower bound for evacuation times and optimized evacuation directions. For more complex geometries, this is no longer a single movement towards one or two exits, but may involve rather complex movements in a building or in a street network. The arguably simplest solution is a grid-based potential function where the "uphill direction" leads to the nearest exit (Nishinari et al., 2004). The same can be done using continuous spatial variables, at the expense of much longer computing times (Hoogendoorn et al., 2002). Alternatively, routing can be done along graphs (Hamacher and Tjandra, 2001; Gloor et al., 2004a), which is a much faster technique when the abstraction to a graph is possible.

Evacuation directions depend on the objective of the evacuation. Objectives can be divided into two classes:

- The user optimal solution of an evacuation problem minimizes the individual travel times. In such a solution no evacuee can gain by unilateral deviation from the evacuation plan. This solution is called Nash equilibrium (named after John Forbes Nash (Nash, 1951)). There are studies investigating Nash equilibria for exit door selection problems in evacuation scenarios (Lo et al., 2006; Ehtamo et al., 2010).

- The system optimal solution (SO) of an evacuation problem minimizes the system travel times or average travel time. To solve the SO problem it is common to abstract the evacuation problem to a (dynamic) network flow problem (see, e.g., Ford and Fulkerson, 1962; Jarvis and Ratliff, 1982; Hamacher and Tjandra, 2001). The SO can also be found by strictly simulation based approaches (see Chapter 3.1.3)

Regardless of whether the solution comes from a simulation or a mathematical model, once an evacuation route or plan has been assigned to an evacuee she has to stick to this plan. In other words, it is not possible to

change the evacuation route during the trip. As a consequence it is not possible to react to unexpected incidents during an evacuation. In transport science this method is called day-to-day re-planning or pre-trip re-planning (Cascetta and Cantarella, 1991), while the capability to change the route while on trip is called within-day re-planning or en-route re-planning. Clearly, en-route re-planning capability is more realistic. The process of route or plan creation is often associated with a so-called mental layer and the execution of the plans in the evacuation environment is called network loading. It is obvious that en-route re-planning is more demanding than pre-trip re-planning. For en-route re-planning the plans' adaptation needs to be called frequently from within the network loading, rather than only having to alternate between the network loading and the mental layers as one does in pre-trip re-planning. En-route re-planning is often implemented by a rolling horizon approach. In a rolling horizon approach the re-planning entities make their decisions based on short-term and medium forecasts of the conditions in the transport environment (see (Chiu et al., 2005; Liu et al., 2007) for applications of the rolling horizon approach in the evacuation context). The same approach can be used to integrate sensor information, for example by modifying evacuees' decisions such that sensor data is replicated in a better way (Vortisch, 2005; Flötteröd, 2008). Since models with within-day/en-route re-planning are computationally more demanding, they are usually slower than models that only use day-to-day/pre-trip re-planning.

In recent developments regarding pedestrian evacuation more behavioral aspects are added. Such behavioral aspects help to model the evacuation dynamics more realistically. For example, people may discard the warning as irrelevant; they may not head for the nearest exit; people have a tendency to follow each other (herding) (Helbing et al., 2000; Lieberman et al., 2005). More complex aspects in terms of locomotion can be modeled with force-based models. The most prominent one is the so-called social force model proposed by Helbing and Molnár (1995). In force-based models attracting and repelling forces determine the pedestrian movement. Walls and other obstacles emitting repelling forces, while the evacuees' destinations emit attracting forces. An overview about different force-based models is given in (Oleson et al., 2008). Locomotion is one important behavioral aspect and can be seen as part of the evacuees' low-level behavior. Low-level behavior corresponds to the non-deliberative part of the humans' decision making, like obstacle avoidance. High-level behavior includes complex decision making, like trip planning. In particular AI (artificial intelligence) or agent-oriented groups focus on high-level behavior aspects when modeling evacuees. In those evacuation simulation frameworks the evacuees are modeled by autonomous agents with complex decision-making architectures. Murakami et al. (2003) propose

an evacuation simulation framework, where a rule-based model generates the high-level behavior. In the simulation the agents are divided into leaders and evacuees that make their high-level decision based on a simple set of rules. The leaders advise the evacuees either to go to an exit or they advise the evacuees to follow them. The evacuees can follow the leaders or make their own decision. The actual behavior of both groups is based on a simple rule set. Pan et al. (2007) developed an evacuation simulation based on autonomous agents with individual physical parameters (like age, gender or weight), limited sensing capabilities (like a bounded visual viewing angle) and a system of decision rules, which is triggered by the perceived situation. In general those models are following the belief-desire-intention (BDI) architecture. BDI is based on Bratman's theory of human practical reasoning (Bratman, 1987). The basic idea behind the BDI principle is to separate plan creation (deliberation about what to do) from plan selection and execution. This approach allows to make reasoned decision even under time pressure. Early work on BDI based agents can be found in (Rao and Georgeff, 1991). The more intelligence a deliberating agent has, the more computing time the agent needs to make decisions. For that reason multi-agent systems with high-level decision-making are mostly applied to domains such as the RoboCup simulation league[1], where only a few agents are involved. However, recent developments in the multi-agent community focus also on larger pedestrian simulations, e.g. (Klügl and Rindsfüser, 2007) developed a multi-agent pedestrian simulation with more than 40 000 agents simulating the commuter traffic in the morning hours at the railway station of Bern. This model has also been applied to a simulation of a TGV-train accident in a tunnel (Klügl et al., 2009). In the hypothetical scenario the train got stuck due to a fire at the train's engine. Roughly 800 autonomous agents, representing the trains' passengers, are part of the simulation. With the advance of computer technology, it is very likely that autonomous agent based evacuation simulations will become even more large-scale. However, for the time being there are no such simulation frameworks that can deal with hundreds of thousands of evacuees. And there seems to be universal agreement that these behavioral aspects increase evacuation times; simulations with reduced behavioral aspects can, in consequence, be used as an optimistic benchmark.

2.3 Decision support systems

A more integrative approach is represented by the decision support systems (DSSs). Rather than concentrating on a single aspect, DSSs attempt to give

[1] http://www.robocup.org/robocup-soccer/simulation (accessed June 2010)

integrated environments to the analyst. In consequence, evacuation simulations are usually parts of evacuation DSSs, but other elements are included as well, such as: analytical tools; decision models for simulated individuals (such as the decision when to evacuate); integration with geographic/spatial data bases; possibility to interact with the system in real time; graphical user interface. Examples are TEDDS (which uses MASSVAC as evacuation traffic module; see (Hobeika et al., 1994)), IMDAS (which uses OREMS as evacuation traffic module). With the advent of mobile devices, such as smart phones, that integrate GPS, WiFi and 3G support, a new kind of DSSs are emerging. With those devices a much better integration of real-time information into DSSs is possible. Such information comprises not only the location of the evacuees, but also the flow conditions (e.g. the flow speed can be calculated from a time series of GPS positions). Furthermore it is possible to display location-dependent evacuation recommendations on the mobile devices. An example is (Ishida et al., 2007), where the locations of the evacuees are in real time integrated into a multi-agent simulation. The simulation is then used to predict how different evacuation strategies would work in the real world. Another ongoing project is REPKA (see http://www.iis.fraunhofer.de/bf/ln/referenzprojekte/repka.jsp – accessed March 2011). The aim of the project is to develop a real-time area based evacuation management system. Real-time data about crowd densities and flow conditions will be collected from the evacuees' mobile devices and then put into simulation and optimization models to find appropriate evacuation strategies.

2.4 Computing time for large-scale scenarios

To the author's knowledge, none of the above approaches is able to simulate large-scale scenarios (with millions of entities) while remaining microscopic: With a CA (cellular automata) model, an area of $40\,km \times 40\,km$ translates into 10^{10} cells. Even if every cell only needs 1 Byte, this still translates into 10 GByte of memory, resulting in large simulation times. For the social-force based approach, the problem is the sub-second time resolution that is typically used (Farkas, accessed 2008). DTA approaches seem the most likely candidates, but to the author's knowledge their implementation of the traffic flow dynamics usually is still too time-consuming for scenarios of that size: Ref. (Sbayti et al., 2007) reports a study using Dynasmart-P consisting of 1347 nodes and 3004 links. 200,000 vehicles were loaded onto the network. The runtime for about 30 iterations of 2 hours of simulation was almost 8 hours. This means running one iteration with this 1347 nodes/ 3004 links scenario takes about 16 minutes. If the runtime scales with the scenario size

it would be very time consuming to run larger scenarios. In Ref. (Wen et al., 2006), the DynaMIT framework was applied to a real-time scenario but on a small network (243 nodes and 606 links). In that study a rolling horizon approach was chosen to have a 5 min estimation and 30 min prediction on that network. Two iterations of estimation and two iterations of prediction took about 1 min. If the runtime scales with the size of the network the performance is comparable to the Dynasmart-P approach and again too slow for large-scale scenarios.

One way to achieve faster computation with a microscopic model is to use a model with deliberately large time steps and to concentrate computationally on those areas (links) where the pedestrian movement actually takes place (Gloor et al., 2004b).

Additional speed-ups could be gained by parallel computing (Cetin et al., 2003). On distributed architectures, however, it is necessary to serialize/deserialize objects that move from one CPU to another. When done automatically, it can be found that this is rather slow and thus only suitable for scenarios where there is little communication (e.g. (Standish, accessed 2005)); when done manually, it causes considerable software engineering overhead. However, in situations where the synchronization overhead is small compared to the computational complexity of the model, parallel computing, in principle, can bring big performance gains. This is for example the case when it comes to complex models for movement and collision avoidance. One example that uses parallelization to for collision avoiding movements in a multi-agent pedestrian context is ClearPath (Guy et al., 2009). ClearPath is capable to simulate up to 250 000 agents in a city environment in real-time on a 64 core machine. Shared-memory architectures were quite expensive in the past, but the recent emergence of multi-core architectures and GPUs (graphical processing units) may be attractive options. For first steps in these directions in the context of transportation/pedestrian simulations, see (Strippgen and Nagel, 2009) or (Bleiweiss, 2008).

Chapter 3

Evacuation problem

In the evacuation context a possible objective is to find a solution that minimizes the number of affected people. In general, this corresponds to finding a feasible routing solution for every single person inside the hazard zone. The evacuation problem can be formulated as a network flow problem in a graph $G = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is a finite set of nodes or vertices and $\mathcal{A}$ is a finite set of unidirectional links or arcs. In this work the link node terminology will be used. The evacuation network is a projection from the real world evacuation area to a graph. The real world evacuation area could for example be a street network of a city or a floor plan of a building. In the case of a street network the intersections would be modeled as nodes and the street segments connecting the intersections would be modeled as links. It is also possible to model the floor plan of a building as a graph (Gloor et al., 2004a), however, this would be more complicated since it is not straightforward to decide which parts of the floor plan are nodes and which parts are links. A sketch of an evacuation network is given in Figure 3.1. Nodes s_0 and s_1 denote source nodes, nodes n_0, n_1 and n_2 are intermediate nodes and t_0, t_1 and t_2 are sink nodes. The evacuees are departing from source nodes and evacuating to sink nodes. In general each intermediate node could also be a source node. In this work the physics of the real world is modeled as parameters of the link. An important parameter is the flow capacity $c(a)$ of a link a. The flow capacity describes how many flow units can leave a link per time unit. The flow capacity in the model could be seen as the flow capacity of the bottleneck in the corresponding street segment. It is assumed that the bottleneck is at the end of each link.

In the evacuation context there is a set of evacuees departing from a set of sources and evacuating to a set of destinations. In more formal terms, each evacuee $n = 1 \dots N$ is associated with a source node $s(n)$. The evacuation problem is to find for each evacuee $n = 1 \dots N$ a destination $t(n)$ and a path from $s(n)$ to $t(n)$ with respect to some optimization criterion. As discussed in chapter 1, there are several sub-problems that have to be addressed. This chapter discusses these sub-problems in detail. First of all, there is the routing problem to solve. Three different approaches to the routing problem are discussed. The most straightforward one is the shortest path solution, where every one takes the shortest path to the safe location. This will be discussed in

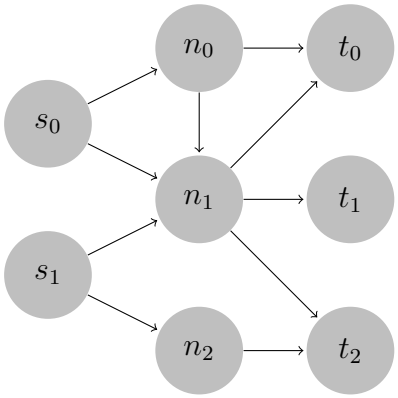


Figure 3.1: Sketch of the evacuation network.

Chapter 3.1.1. A game theoretic approach will be discussed in Chapter 3.1.2. An approach that leads towards a system optimal solution will be introduced in Chapter 3.1.3.

In case of an evacuation, the network may have time-dependent attributes. For instance, large-scale inundations or conflagrations do not cover the entire hazard zone at once. These time-dependent attributes need to be addressed in an appropriate manner. One way is to model it as a time-dependent network. A proposal for an implementation is discussed in Chapter 3.2.

The overall egress time is a crucial aspect in most evacuation situations. An aspect that is often neglected in evacuation route optimization is the risk. An approach how to model risk as a static travel cost offset is given in Chapter 3.3.

The general evacuation strategy is to move every one out of the hazard zone. However, in some situations there might be not enough time to get every one out. One way to deal with this problem is to build shelters within the hazard zone where evacuees can escape to. Shelters are sinks with limited capacity since each shelter has a limited storage capacity. Various problems arise with regard to sinks with limited storage capacity. Some of these problems will be discussed in Chapter 3.4. Finally, a brief discussion concludes this chapter (Chapter 3.5).

3.1 Basic evacuation routing

3.1.1 Shortest path solution

The most straightforward solution to an evacuation problem is the shortest path solution. Shortest path algorithms have been applied to transportation

research for a long time (see, e.g., (Whiting and Hillier, 1960)). In the literature there are a lot of shortest path algorithms, many of them are based on the most prominent one—the so-called Dijkstra shortest path algorithm (Dijkstra, 1959). The idea behind this algorithm is to explore the network in a successive way beginning at the origin. This procedure is comparable to the so-called best-first-search (see, e.g., Russel and Norvig, 1995). The worst-case runtime for Dijkstra's shortest path algorithm on a network with $|\mathcal{N}|$ nodes and $|\mathcal{A}|$ links is $O(|\mathcal{N}|^2)$ (Cormen et al., 2009). For sparse graphs, the runtime performance can be improved to $O(|\mathcal{A}| + |\mathcal{N}|\, log\, |\mathcal{N}|)$ by using a Fibonacci heap as a priority queue to store the already visited nodes (Fredman and Tarjan, 1987). The same runtime performance can be reached using a binary heap as priority queue and cost function that assigns costs to nodes, not to links. Therefore an equivalent transformation of a link based cost function to a node based cost function is needed. A proof of the existence of such a transformation and a complexity analysis of the node-costs based shortest path algorithm are given by (Barbehenn, 1998). Other common adaptions of Dijkstra's shortest path algorithm are heuristic approaches like A^* (e.g., Russel and Norvig, 1995). In most cases A^* improves the runtime considerably, but since this algorithm is based on a heuristic, a better runtime over Dijkstra's algorithm cannot be guaranteed. A discussion on the performance of A^* and other heuristic approaches in the multi-agent transport simulation context is given in (Lefebvre and Balmer, 2007).

All these kind of shortest path algorithms find the shortest path in a weighted graph from one node to any other, whereby the actual weights for a link are defined by a generalized cost function. However, there is no particular node as the target of the shortest path calculation, as the evacuees have more than one safe place to run to. Instead, in the underlying domain every node outside the evacuation area is a possible destination for an evacuee that is looking for an escape route. Thus the evacuation problem is in general a multi-destination problem. To resolve this, the standard approach (e.g. Ford and Fulkerson, 1962; Lu et al., 2005) is to extend the network in the following way: All exit links (i.e. links that are originating inside the evacuation area and terminating outside the evacuation area) are connected, using virtual links with infinite flow capacity and zero length, to a super-node, and all paths are routed to the super-node (see Fig. 3.2), meaning $t(n) = t^{super}$ for all agents $n = 1 \dots N$.

This way, the problem is reduced to a multi-source single-destination problem. Dijkstra's algorithm will always find the shortest path from any node inside the evacuation area to this evacuation node and, in consequence, to safety. In a multi-agent based evacuation simulation the shortest path solution can be achieved by calculating the shortest evacuation route (in terms of

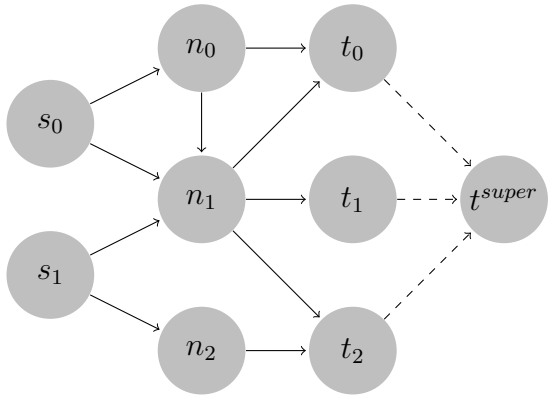


Figure 3.2: Sketch of the evacuation network with the super-node as sink and virtual (i.e. zero cost) evacuation links.

Algorithm 1 Shortest path routing

1. initialize τ_a with the free-flow travel time for all links a.
2. calculate routes based on link costs τ_a

time) to the super-node for every single agent. The algorithm to calculate the shortest path solution relies on the information about the free speed travel time τ_a for every link $a \in \mathcal{A}$. Algorithm 1 drafts the Shortest-path routing logic. However, the shortest path solution does not take congestion into consideration. In reality the link travel time depends on the level of congestion. In the underlying traffic flow simulation every link has a specific flow capacity. The travel time on a link depends on that flow capacity and the level of congestion. Since the demand on a link is not constant over time the link travel time is time-dependent. There are different optimization approaches to find better solutions than the shortest path solution. Two different optimization approaches will be discussed in the following.

3.1.2 Nash equilibrium approach

In most (but not all) evacuation situations, the Nash equilibrium leads to a shorter overall evacuation time than when everybody moves to the geographically nearest evacuation point. A Nash equilibrium means that nobody has an incentive to deviate. The Nash equilibrium in an evacuation situation can therefore be considered as a solution that could be reached by appropriate training.

The Nash equilibrium is named after John Forbes Nash and describes a state in a competitive two or more player game where no player can gain by unilateral deviation from her current strategy (Nash, 1951). In the evacuation context a Nash equilibrium describes a state where no evacuee can improve her evacuation performance by unilateral deviation from her current evacuation strategy. The concept of the Nash equilibrium corresponds to Wardrop's first principle for traffic on a street network, which states that at equilibrium the journey time of all routes actually used are equal or less than those that would be experienced on any unused route (Wardrop, 1952). In the literature it is often referred to as "user equilibrium", since every one chooses a route that is best for her under the given circumstances. In the remainder of this work it will be referred to as "Nash equilibrium". The Nash routing problem can be seen as an N-player non-cooperative network congestion game where each player $n \in N$ is associated with pair of nodes $s(n) \in \mathcal{S}$ and $t(n) \in \mathcal{T}$. Each of the player wishes to send one flow unit from $s(n)$ to $t(n)$ in the least amount of time (see, e.g., (Meyers, 2006) for a detailed discussion on network congestion games).

Early works treat this problem as a static network flow problem. Beckmann et al. (1956) introduced an objective function for this problem:

$$\sum_{a \in \mathcal{A}} \int_0^{x_a} C_a(x) dx \tag{3.1}$$

where $C_a(x)$ are the travel costs that will occur when traveling along link a under flow x (i.e. the link travel time). However, Beckmann's objective function holds only for the static case. In transport science it is often argued that the static representation is adequate to represent conditions that occur during rush hour. In general, one has to deal with time-dependent link travel times, meaning one has to deal with a flow over time with flow dependent link travel times problem. Janson (1991) proposed an algorithm to solve the dynamic generalization of the objective function introduced by Beckmann et al., which can be written as:

$$\int_t \sum_{a \in \mathcal{A}} \int_0^{x_a} C_a(x(t)) dx \, dt \tag{3.2}$$

However, Lin and Lo (2000) contradicted the validity of this dynamic generalization. The authors constructed a simple counter example where the system moves rather to system optimum than to Nash equilibrium by minimizing Equation (3.2). The reason for this inconsistency is that the proposed objective function is evaluated over the entire time span, so that an increase in the objective function value during some time period can be compensated through

a decrease in the objective function value during another time period. In order to have a consistent objective function it has to be guaranteed that Wardrop's first principle is valid for any time period. There are other approaches to find a dynamic generalization of Beckmann's objective function (see, e.g.,(Han and Heydecker, 2005). However, existing mathematical approaches are far away from being applicable for problems of realistic size (Köhler et al., 2009).

In theory a Nash equilibrium can also be reached through a simulation based approach. As above mentioned the Nash routing problem can be seen as a N-player non-cooperative network congestion game. If the network congestion game is repeated over and over again, then the players would learn the best strategy under the given circumstances (i.e. the behavior of the others.). In economics this is known as fictitious play (Brown, 1951). Fuldenberg and Levine (1998) showed that if a fictitious play converges, then this must be a Nash equilibrium. Cascetta (1989) has shown that a Nash equilibrium can be reached if the routing problem is modeled as fictitious play, where every participating traveler tries to minimize her individual travel time through an iterative best-response strategy. Gawron (1998) proposed a simulation based iterative learning algorithm with a (approximately) best-response strategy. The iterative learning algorithm starts with a given starting solution and tries to improve it through trial and error. The learning algorithm relies on a cost function that appraises the goodness of routing solutions. Each individual routing solution will be scored according to this cost function. After each iteration a certain fraction of the agents will be chosen for re-planning. While re-planning, a router produces a new route for every re-planning agent based on aggregated values of the experienced travel times. This means that the re-planning agents take the experienced travel times from previous iterations as the expected travel times for the next iteration. The algorithm stops when the generated new routes are exactly the same as before the re-planning. In this case the system is in a steady state. However, since the re-planning strategy is only approximately best-response the result is not necessarily a Nash equilibrium.

The experienced travel times in Gawron's algorithm are measured values for previously occurred link enter times. But for the time-dependent router it is necessary to have data about the expected travel time for arbitrary link enter times. Raney and Nagel (2004) discuss a method to deal with this problem by aggregating the experienced travel times over predetermined time slices. As a result, it could happen that the router systematical over- or underestimates the expected travel times. This bias leads to implausible results. Raney and Nagel demonstrate this issue with a simple example where all agents chose side roads instead of the freeway. To overcome this problem, they introduce a so-called agent database, which is essentially a memory for

Algorithm 2 NE approach

1. Initialize $\tau_a(k) = \tau_a^{\text{free}}$ for all a and k for all links a and time steps k
2. Calculate shortest path for all agents n according to Algorithm 1
3. Repeat for many iterations:
   (a) Load all agents on the network
   (b) Extract time-dependent link travel costs $C_a(k) = \tau_a(k)$ for all a and k
   (c) For every agent $n = 1 \dots N$, do:
      - With $P_{reroute}$:
        i. Compute a new route from $s(n)$ to $t(n)$ based on the experienced travel costs $C_a(k)$ from the previous iteration
      - With P_{select}:
        i. Select an already tested route out of n's memory

every single agent to store already tested routes for later execution. During re-planning the agents are split into two groups. For the first group, the smaller one, the router produces new best response routes. For every agent from the second group an already tested route will be selected based on the performance of the routes. Several different selection mechanisms are possible. The most straightforward one is always select the route with the (expected) least travel time. Better results have been achieved by a Metropolis sampling like (Metropolis et al., 1953) selection mechanism. Details on the selection mechanism will be discussed later in Chapter 4.1.

The iterative learning algorithm for the approximately Nash equilibrium (NE approach) can be characterized as follows. The probability for an agent to be chosen for re-routing is denoted by $P_{reroute}$, and P_{select} denotes the probability for plan selection. Note that $P_{reroute} + P_{select} = 1$. The cost function provided to re-planning agents in the NE approach only comprises of time-dependent link travel times. Formally, the real-valued time is discretized into K slices of length T, which are indexed by $k = 0...K-1$. The time-dependent link travel time when entering link a in time step k is denoted by $\tau_a(k)$. The costs function therefore is:

$$C_a(k) = \tau_a(k) \tag{3.3}$$

Algorithm 2 drafts the Nash-equilibrium routing logic.

3.1.3 Marginal social cost based approach

A system optimal routing solution minimizes the total travel time in the system. This corresponds to Wardrop's second principle (Wardrop, 1952). In principle, there are two different ways to find optimal solutions. One could apply a learning framework that converges towards an optimal solution or one could solve the evacuation problem by an analytic algorithm that guarantees to find the optimal solution. Analytical solutions to the evacuation problem are described in flow theory. E.g. Skutella (2009) describes a combinatorial approach to the evacuation problem. That approach relies on dynamic network flows. The usual network flow model is extended in a way that flow units need a certain amount of time to traverse an arc, and the flow rate of each arc is limited through capacity constraints. The solution to the evacuation problem is to send the supplies (demand) from a set of source nodes to a set of sink nodes in an optimal way. There are several definitions of "optimal". Optimality could mean:

- Minimization of the "egress time" or evacuation end time (i.e. the time needed until the last one reaches the sink).
- Minimization of the average (or total) travel time.
- Maximization of the amount of flow that has already reached the sink at each time step.

The triple optimization theorem formulated by Chalmet et al. (1982) and proved by Jarvis and Ratliff (1982) says that the solution which minimizes average evacuation time also maximizes the amount of flow that has reached the sink at each time step and therefore also minimizes the egress time. In the evacuation context the most obvious optimization goal would be the maximization of the amount of flow that has reached the sink at each time step. This would guarantee that at all times the maximum number of evacuees is evacuated. A flow that achieves this goal is called Earliest Arrival Flow.

Ford and Fulkerson (1962) discuss a dynamic flow that minimizes the average travel time by calculation of a Maximum Dynamic Flow in the time-expanded network. A time-expanded network holds a copy of the whole network for every discretized time step, meaning every node/edge is replicated for every time step $t = 0, 1, \ldots, T$, and additional links are connecting the nodes between the time steps. The quality of the solution only depends on the granularity of the discretization. The complexity of the network grows linear to the number of discretization steps and thus one would have to make a trade-off between computational expense and quality of the solution. A discrete time piecewise linear model to solve the system optimum problem is

given by Merchant and Nemhauser (1978). However, the model relies on a non-convex constraint set, which makes it complicated to prove optimality. Lu et al. (2005) introduce a heuristic approximation to the optimization problem. They introduce a capacity dependent generalized cost function for the Dijkstra routing algorithm. The proposed algorithm is called Capacity Constraint Route Planer (CCRP). The runtime of the CCRP algorithm is $O(p * n \, log \, n)$, where p is the number of evacuees and n is the number of nodes. Compared to the minimum runtime of $O((T * n)^6)$ using a linear programming approach and time expanded network over T time steps CCRP is much faster. The experimental results testify also a good performance regarding the quality of the approximation.

In a multi-agent simulation context the average travel time can be minimized through a learning algorithm. This is discussed by Lämmel and Flötteröd (2009). The procedure is called approximately system optimal assignment. It can be obtained by the same self-serving routing logic that is employed to calculate a Nash equilibrium. The only difference is that for a system optimum, the travel time based on which agents evaluate their routes needs to be replaced by the marginal travel time (Beckmann et al., 1956; Peeta and Mahmassani, 1995). The marginal travel time of a route is the amount by which the total system travel time changes if one additional vehicle drives along that route. It is the sum of the cost experienced by the added vehicle (τ) and the cost imposed on other vehicles. The latter is denoted here as the external cost (C^s). Thus, minimizing the marginal individual travel time will also minimize the total or average system travel time and therewith minimize the egress time and maximize the number of evacuees that have already reached the sink at each time step. If the simulation was performed with the "real" marginal travel times, then a reduction of the marginal travel times would also reduce the system travel time. Consequently, if the simulation would converge to the minimal marginal travel times, it would converge to the system optimum. Since it is neither guaranteed that the system converges nor is there a proof about the quality of the approximation it is referred here as the more general term of marginal social cost based approach (MSCB approach). In (Lämmel and Flötteröd, 2009) the external costs that one additional agent that enters a congested link a at time t_0 imposes on other agents are approximated by:

$$C_a^s(t_0) \approx t^e(t_0) - (t_0 + \tau^{free}) \tag{3.4}$$

where t_0 denotes the link entry time of the causative agent, $t^e(t_0)$ denotes the time at which the congestion dissolves if entered at t_0 and τ^{free} denotes the free speed travel time. Equation 3.4 gives a discrete approximation of a

continuous formulation for the external costs. Details are given in (Lämmel and Flötteröd, 2009). Subsequently, a simpler approximation that leads to better results is discussed. A comparative study of both approximations is given in Chapter 4.3.

The following considerations help to formulate an approximation of the external costs.

- External costs only occur, if there is a queue on the affected link ($\tau_a > \tau_a^{free}$), which means the demand is or has been higher than the flow capacity (q_a) of the link.
- Queues on links are induced by bottlenecks at the end of the links.
- If spill back is neglected, the outflow rate of the affected link is equal to q_a (flow capacity) until the queue dissolves.
- The "causative" agent n for which the external costs are to be calculated delays every agent further upstream that leaves link a before $t_a^e(t_0)$.
- Since the outflow rate is assumed constant during the whole period of congestion, each upstream agent is delayed by the same amount of time.
- The amount of delay that n imposes on others is equal to the time between two consecutive agents ($1/q_a$).
- The "causative" agent n delays others from the time it enters the bottleneck.

Because of the constant outflow rate, the time that an agent spends in the bottleneck is $1/q_a$. Let $t_a^{lv}(t_0)$ denote the time the "causative" agent n enters the bottleneck at the end of link a, if it had entered a at t_0. The number of affected agents is equal to the number of agents that enter the bottleneck of a in the time period from $t_a^{lv}(t_0)$ to $t_a^e(t_0)$. This is equal to $q_a * (t_a^e(t_0) - t_a^{lv}(t_0))$. Taking all together, the external costs can be approximated by:

$$C_a^s(t_0) \approx q_a * (t_a^e(t_0) - t_a^{lv}(t_0)) * 1/q_a \tag{3.5}$$

$$\approx t_a^e(t_0) - t_a^{lv}(t_0) \tag{3.6}$$

If there were no spill back in the system, then the outflow rate of a congested link would be constant and $C_a^s(t_0) = t_a^e(t_0) - t_a^{lv}(t_0)$ would hold. Since there is, however, no guaranty that spill back does not occur, Equation 3.6 gives only an approximation of the external costs.

For the application of this result to an MSCB approach route assignment the real-valued time is discretized, similar to the time-dependent link travel

Algorithm 3 MSCB approach

1. Initialize $C_a^s(k) = 0$ and $\tau_a(k) = \tau_a^{\text{free}}$ for all links a and time steps k
2. Calculate shortest path for all agents n according to Algorithm 1
3. Repeat for many iterations:
   (a) Load all agents on the network
   (b) Extract time-dependent link travel costs $C_a(k) = \tau_a(k) + C_a^s(k)$ for all links a and k
   (c) For every agent $n = 1 \dots N$, do:
      - With $P_{reroute}$:
        i. Compute a new route from $s(n)$ to $t(n)$ based on the experienced travel costs $C_a(k)$ from the previous iteration
      - With P_{select}:
        i. Select an already tested route out of n's memory

times for the Nash equilibrium approach, into K slices of length T, which are indexed by $k = 0...K-1$. The time-dependent link external cost when entering link a in time period k is denoted by $C_a^s(k)$, which corresponds to the average amount of external costs during this period. The time-dependent link travel costs for link a if entered in time period k then are approximated by:

$$C_a(k) = \tau_a(k) + C_a^s(k) \tag{3.7}$$

Algorithm 3 outlines a straightforward implementation of this approach in a time-discrete multi-agent simulation.

3.2 Time-dependent networks

In the case of an evacuation simulation the network has time-dependent attributes. For instance, large-scale inundations or conflagrations do not cover the entire hazard zone at once. These time-dependent aspects must not be neglected in evacuation planning, otherwise inappropriate recommendations may be the result. One solution to model the spreading of inundations or conflagrations is the application of time-dependent networks where parameters like walking speed can vary over time. Time-dependent networks have been applied to evacuation planning (Lu et al., 2005) and are often mod-

eled as time expanded graphs (Kaufman and Smith, 1993; Pallottino and Scutella, 1998; Köhler et al., 2002). In contrast, so-called time-aggregated graphs (George et al., 2007) omit the explicit time expansion, and rather use a time-dependent look-up of the link cost. The notation of that reference implies T time intervals which apply uniformly to all links; their $O(\log T)$ time complexity of the link cost lookup implies that these time intervals need not to be equally spaced.

Yet, in the case of a tsunami inundation, the structure of the link cost changes is quite different: There is, at least in an abstract interpretation, only a switch from "passable" to "non-passable", but that switch can happen at arbitrary times. With the above techniques, one would either need as many time intervals T as there are such switches, or several switches would need to be combined into one time-slice. In order to avoid these restrictions, this work introduces the following approach:

- The *interface*, containing the calls to the network attributes, in particular link speeds, link capacities, and link widths, is made time-dependent, allowing the implementation of arbitrary time-dependent functions behind the interface. This corresponds to the time-aggregated graph technique.

- The *implementation* presented here uses so-called network change events. The change events are applied to the edges, modifying their attributes (free speed, flow capacity) at arbitrary points in time. A change event is valid until the next change event will be applied.

A similar technique has also been used for the implementation of a fast combinatorial optimization algorithm for large-scale evacuation problems (Dressler et al., 2011).

The simulation network represents the area that is accessible by the evacuees. In the case of a vehicular evacuation this network consists of all accessible streets. Each street segment defines a link. The parameters of the links are the length, flow capacity and the free speed. The most obvious parameter to model the time-dependent disaster related blocking of links would be the flow capacity parameter. One only has to set this parameter to zero as soon as a link gets blocked. However, the flow capacity has no direct impact on the routing algorithm and therefore the agents would recognize the blocking of a link through their experienced travel times only. This would mean that at least for the shortest path solution no time-dependent aspects would be taken into account. Therefore, it seems to be more practical to set the free speed to zero instead of the flow capacity. If the free speed of a link is zero then its travel costs become infinite. More formal, if link a becomes non-passable dur-

ing time interval i, then the travel costs for the NE approach are calculated as follows

$$C_a(k) = \begin{cases} \infty, & \text{if } k >= i \\ \tau_a(k), & \text{otherwise.} \end{cases} \tag{3.8}$$

and for the MSCB approach

$$C_a(k) = \begin{cases} \infty, & \text{if } k >= i \\ \tau_a(k) + C_a^s(k), & \text{otherwise.} \end{cases} \tag{3.9}$$

Algorithms 1, 2 and 3 have to be adapted accordingly.

3.3 Risk reduction

The overall egress time is a crucial aspect in most evacuation situations. There are many models that find optimal routing strategies for evacuation scenarios. An aspect that is often neglected in evacuation route optimization is the risk. Risk is related to safety. Safety is a basic need for individuals and societies. Safety can be roughly defined by: existing risk < acceptable risk. It can also be discriminated from security by dealing with non-intentional threats. The risk, and consequently also the safety if the acceptable risk is specified can be quantified based on the following formula (Lämmel et al., in press):

$$R = \int D(1 - C)P(t)dt \tag{3.10}$$

The damage is denoted by D, the coping capability by C, and $P(t)$ is the occurrence probability of the event. The criterion usually applied to assess a risk is: $R <$ acceptable risk. Note that there is always a residual risk ($RR > 0$), which cannot be reduced by technical or management means. In case of a tsunami, the physical safety or lives of people are at risk. Evacuation is one means in ensuring the safety, especially to avoid the risk and threat to human life. Another strategy would be to build tsunami safe buildings (shelters), which would increase C. The implementation of shelters within the simulation framework is discussed later.

The risk and therewith the utility of an evacuation path often depends on uncertain aspects. One uncertain aspect in the underlying domain is the advance warning time τ^{warn}. It is for the following assumed that τ^{warn} follows an unknown probability distribution with $P(\tau^{warn} > 0) = 1$, i.e. there is always a warning *before* the event. Consider an example as depicted in Figure **??** with two different evacuation paths (p_0 and p_1). Evacuation path p_0 works independently of the advance warning time τ^{warn}, but takes considerably more

Figure 3.3: Illustration of two different evacuation paths.

travel time than evacuation path p_1. The travel time of a path p is denoted by τ_p^{travel}. If the advance warning time is long enough, the travel time for p_1 is smaller than the travel time for p_0 ($\tau_{p_0}^{travel} > \tau_{p_1}^{travel}$). It is assumed that path p_0 leads straight away from the danger. This means, the utility of path p_0 is independent of the advance warning time. In contrast, path p_1 first moves towards the danger for a time period of T before it leads to safety. This situation could be compared to an inundation scenario where near to the shore is a bridge that leads to safety. If an evacuee starts further inland, then she would first have to move towards the shore (danger) in order to reach this bridge. As a result, the utility of p_1 then depends on τ^{warn}. Under the assumption that there is always a warning *before* the event, the utility for p_0 and p_1 can be formulated as follows:

$$U(p_0 \mid \tau^{warn}) = -\tau_{p_0}^{travel} \tag{3.11}$$

$$U(p_1 \mid \tau^{warn}) = \begin{cases} -\infty, & \text{if } \tau^{warn} < T \\ -\tau_{p_1}^{travel}, & \text{otherwise.} \end{cases} \tag{3.12}$$

The expected value for each utility can be calculated as follows:

$$E(U(p_0 \mid \tau^{warn})) = E(U(p_0)) = -\tau_{p_0}^{travel}. \tag{3.13}$$

$$E(U(p_1 \mid \tau^{warn})) = P(\tau^{warn} < T) * -\infty + (1 - P(\tau^{warn} < T)) * -\tau_{p_1}^{travel} = -\infty. \tag{3.14}$$

Based on this consideration risky evacuation paths p_1 should be banned, as long as a non-risky solution p_0 exists. If no risk-free path exists, then the solution with the lowest risk should be chosen. This can be achieved by adding risk costs to links. This is similar to the priority evacuation discussed by Hamacher and Tufekci (1987). In a priority evacuation priority levels are assigned to different areas of the hazard zone depending on the level of endangerment. The objectives in the priority evacuation context are:

- The total evacuation time has to be minimized.
- High priority levels have to be cleared before lower priority levels.
- Movements from a lower priority level to a higher priority level have to be avoided.

The authors introduced the lexicographic min cost flows to solve this multi-objective problem in a time expanded graph. The lexicocgraphic min cost flow problem is an extension to the well-known dynamic network flow problem (e.g. (Ford and Fulkerson, 1962; Chalmet et al., 1982)). The basic idea behind the priority evacuation is to penalize links that are leading from a lower priority level to a higher priority level with an extra cost. In the current context an additional link dependent risk term (C_a^r) can be defined for every link a (see, Chapter 6.1 for details). This term then has to be added to the corresponding link travel costs $C_a(k)$ defined in Equations (3.3) and (3.7).

3.4 Shelter assignment

The evacuation strategies discussed so far require all evacuees to leave the hazard zone. However, if the available egress time was too short, not everyone would make it. One way to deal with this problem is to build shelters within the hazard zone where evacuees can escape to. In the underlying situation (i.e. escape from a tsunami) some strong buildings or explicitly designed shelters could serve as a place of refuge. However, in most situations there will be not enough shelter space to host everyone. This means despite of shelter buildings within the hazard zone some evacuees nevertheless may need to evacuate to the safe hinterland. This constitutes a different evacuation problem for which the single destination evacuation network shown in Figure 3.2 is no longer adequate. The reason is, when it comes to shelters one has to deal with a limited capacity sinks problem, where only nodes in the safe hinterland can be connected with zero cost links to a super sink. A more adequate representation is given in Figure 3.4, where node t^{super} again is a super sink

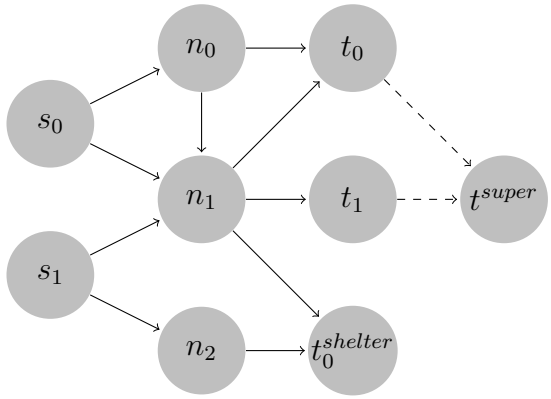


Figure 3.4: Sample evacuation network with shelter nodes

with unlimited storage capacity and node $t_0^{shelter}$ represents a shelter sink with limited storage capacity. Several questions arise when it comes to shelters:

- Given the locations and capacities of the shelter buildings, how to assign the evacuees to the shelters (assignment problem).
- What are the best locations for yet to build shelter buildings (allocation problem).
- Given the locations of the shelter buildings, what are the appropriate storage capacities (capacity problem).

This work discusses the assignment problem only. For a discussion of the capacity problem, see (Flötteröd and Lämmel, 2010). The shelter allocation problem can be formulated as a generalization of the shelter capacity problem. This, however, will not be discussed in detail. The interested reader is referred to (Lozano et al., 1998; Chowdhury et al., 2001; Hauskrecht and Singlair, 2003) for discussions of allocation problems.

The shelter assignment problem is related to the so called generalized assignment problem (GAP), which is known to be $\mathcal{NP}$-hard (cf. (Garey and Johnson, 1979; Marshall et al., 1986)). For evacuations, the assignment problem needs to be solved on top of a dynamic network flow problem, which makes the underlying cost function of assigning an agent to a given shelter interdependent with the assignment and route choice of the other agents. Therefore an analytical solution seems to be impractical in terms of computational costs. Learning algorithms often find good approximations to complex problems where analytical solutions are impractical. Usually, learning algorithms start with a random or simple start configuration, which then is gradually improved until the desired solution or some predefined stopping criterion has

Algorithm 4 Metropolis algorithm to minimize f

1. Begin with a random starting configuration i
2. **repeat**
   (a) Generate a candidate solution j by mutating i
   (b) **if** $f(j) \leq f(i)$ **then** $i = j$
   else if $\exp(\frac{f(i)-f(j)}{c}) > \text{random}[0, 1)$ **then** $i = j$

   until stop criterion reached

been reached. Many learning algorithms are based on a so called natural selection mechanism, which has been inspired by the natural process of evolution, where

> "... structural changes do occur [...] by chance only. Nevertheless, the changes which register will be mostly in a given direction, since those less efficiently designed will not survive. In this way, highly efficient and extraordinarily complicated designs will 'evolve' without any intelligent planning simply because (1) the capacity to survive is a 'scoring' mechanism and (2) in a long interval of time there will be many generations" (Dunham, 1963).

The simplest procedure to emulate this natural process is a so called hill climbing or iterative improvement algorithm (cf. Russel and Norvig, 1995). Iterative improvement algorithms accept proposed mutations of the system as a new intermediate state only if it would be better as the previous state according to some kind of scoring function. This can be compared to hill climbing where the climber always goes uphill. It is obvious that, when only going uphill, one can get stuck at a local optimum. One way to deal with this problem is to repeat the hill climbing procedure with random starting configurations over and over again until a stopping criterion has been reached. Another approach to overcome the local optimum problem is to accept lower score states with a small probability. A well established method of this kind is the so called Metropolis algorithm (Metropolis et al., 1953), which gives an algorithm that describes the interactions between molecules for any substance when moving towards a low energy state as it is known from statistical thermodynamics. An outline of the algorithm is given in Algorithm 4, where $f(i)$ denotes the energy (disutility) of configuration i and c denotes a control parameter sometimes called temperature. The main problem of this algorithm is to find the right value for the control parameter c. A higher value leads to

Algorithm 5 Simulated annealing algorithm

1. Begin with a random starting configuration i
2. Initialize control parameter c and the number of transitions k for the metropolis algorithm
3. **repeat**
   (a) **for** $l = 1$ **to** k
      i. Generate a candidate solution j by mutating i
      ii. **if** $f(j) \leq f(i)$ **then** $i = j$
      else if $\exp(\frac{f(i)-f(j)}{c}) > \text{random}[0, 1)$ **then** $i = j$
   (b) Calculate new c and new k

   until stop criterion reached

a higher probability of accepting a utility decreasing move. If c goes towards zero, the probability of accepting a utility decreasing move also goes towards zero and the algorithm performs simple hill climbing. In the early 1980s it has been independently shown by two different research groups that simulated annealing, a procedure derived from statistical thermodynamics, gives near optimal solutions to a variety of different combinatorial optimization problems with reasonable computational effort (Kirkpatrick et al., 1983; Černý, 1985; Kirkpatrick, 1984). A straightforward implementation of the simulated annealing algorithm is outlined in Algorithm 5. It can be guaranteed that simulated annealing converges asymptotically to the optimal solution, meaning that it may take an infinite number of iterations to obtain the optimal solution (Aarts and Korst, 1989).

Simulated annealing has been applied to assignment problems in the past. A combination of simulated annealing with tabu search to solve the GAP has been proposed in (Osman, 1995). In another study, the related, quadratic assignment problem has been approximately solved using a simulated annealing approach (Wilhelm and Ward, 1987). In (Klügl, 1995) the performance of a simulated annealing algorithm has been compared to a genetic algorithm approach on a timetable assignment problem. Genetic algorithms have also been applied to assignment problems in other studies (see e.g. (Bernardino et al., 2010)). A survey on algorithms to solve the GAP is given in (Cattrysse and Van Wassenhove, 1992).

Based on the investigated literature, a simulated annealing algorithm seems to be an appropriate candidate to solve the shelter assignment prob-

lem. However, in this thesis a different, but similar approach is chosen. The Metropolis algorithm and simulated annealing rely on the so-called Metropolis sampling to allow energy (disutility) increasing moves with a small probability. As discussed, this is necessary in order not to become stuck in a suboptimal local state. In the discussed iterative learning cycle for the NE approach and the MSCB approach the particular agent chooses a new route based on the experienced travel costs from the previous iteration, meaning it is assumed that the experienced travel costs from the previous iteration do not change in the upcoming iteration. However, since the choice of new routes influences the travel costs of the upcoming iteration, the assumed travel costs are only an approximation of the resulting travel costs in the upcoming iteration. The assumed travel costs sometimes overestimate and sometimes underestimate the real travel costs. From a behavioral point of view this can be seen as a random perception error of the agents when trying to make a best response decision. In the following, a shelter assignment procedure is discussed where the random perception error helps to prevent the system to become stuck in a suboptimal state too early.

The shelter assignment problem is to find an assignment of the evacuees to the shelters that minimizes individual travel times (NE approach) or the system travel time (MSCB approach). In either case, the (re-)assignment of an agent to a shelter requires also to re-compute its route. Consistency is maintained here in that the NE approach shelter assignment is combined with a NE approach route assignment and the MSCB approach shelter assignment is combined with an MSCB approach route assignment.

The shelter assignment learning procedure is seamlessly integrated in the iterative learning cycle, which has been introduced for the route assignment in Chapters 3.1.2 and 3.1.3, and consists of two basic operations. A naive operation is to select a non-full shelter t' and **shift** agent n to that shelter if it is expected, based on the experienced travel costs from the previous iteration that the agent would gain. As long as there is a small positive probability that the expected travel costs underestimate the actual travel costs of the upcoming iteration (i.e. positive probability for a cost increasing move), there is also a positive probability that with the **shift** operation every possible configuration can be reached in the long run. However, it is likely that a high number of cost increasing shifts are necessary to switch the shelter assignment of two agents from which both would gain [1]. Therefore a second operation is introduced here. The **switch** operation selects for an agent n who re-plans her shelter assignment a "switching partner" n' in another shelter randomly

[1] This is similar to problems that occur when solving the well-known traveling salesman problem by simulated annealing. (see (Černý, 1985))

Algorithm 6 Shelter assignment algorithm

1. Initialize $\tau_a(k) = \tau_a^{\text{free}}$ for all a and k for all links a and time steps k
2. **For** every agent $n = 1 \dots N$, **do**:
   (a) Randomly assign a non-full shelter t as destination $t(n)$
   (b) Calculate shortest path from $s(n)$ to $t(n)$ according to Algorithm 1
3. **Repeat** for many iterations:
   (a) Load all agents on the network
   (b) Extract time-dependent link travel costs
   (c) **for** every agent $n = 1 \dots N$, **do** with P_{replan}:
      - With $P_{reroute}$:
        i. compute a new route from $s(n)$ to $t(n)$ based on the experienced travel costs from the previous iteration
      - With P_{shift}:
        i. Randomly select a non-full shelter t'
        ii. **if** $\hat{C}(s(n), t') < C(s(n), t(n))$ **then** shift agent n to t'
      - With P_{switch}, randomly selected agent n' from $1, \dots, N$
        i. Calculate the expected benefit of the switch δ
        ii. **if** $\delta > 0$, **then** switch the destinations of n and n' and re-route both agents

and decides, whether the agents have to switch their shelters. The decision for the **switch** depends on the expected gains. For the NE approach both agent would switch their shelters if both expect to gain [2]. For the MSCB approach a switch will be performed, if it is expected that the system would gain. The system would gain, if the expected average travel costs of both agents would decrease because of the **switch**.

The iterative simulation conducts a shelter **shift** with a certain probability P_{shift}, with probability P_{switch} it conducts a shelter **switch**, and it also maintains the option of a plain route re-computation with $P_{reroute}$. Algorithm 6 defines the details of this logic, where $C(i, j)$ denotes the experienced travel

[2] Assuming that the impact of the switched agents on the link travel times is negligible, then one can say that if the situation before the switch operation was a Nash equilibrium, the situation after the switch is also a Nash equilibrium, albeit a different one.

costs from node i to node j in the previous iteration, and $\hat{C}(i,j)$ denotes the expected travel costs from i to j in the upcoming iteration, which are assumed to be equal. The benefit function *delta* depends on the objective. For NE approach it is:

$$\delta = \min(C(s(n),t(n)) - \hat{C}(s(n),t(n')), C(s(n'),t(n')) - \hat{C}(s(n'),t(n))) \quad (3.15)$$

For MSCB approach it is:

$$\delta = C(s(n),t(n)) + C(s(n'),t(n')) - \hat{C}(s(n),t(n')) - \hat{C}(s(n'),t(n)) \quad (3.16)$$

If an assignment would not leave enough time to reach the safe location, then its costs are set to ∞. In the current implementation of Algorithm 6 the agents do not have a memory, meaning the agents cannot select an already tested plan from earlier iterations. The memory is useful to revise already scored plans again, which leads to a smoother relaxation of the system. However, different plans in an agent's memory could have different shelters as their destination and therefore selecting plans out of the memory of an agent without considering the actual shelter occupancy could lead to overloaded shelters. For this reason the agents' memory is switched off.

3.5 Conclusion

Different aspects of the evacuation problem have been discussed in this chapter. Most of them, in particular the proposed routing strategies, are not only important for evacuation planning but also for general transportation tasks. So far it is not obvious which routing strategy is the most appropriate one in the evacuation context. The shortest path solution (SP solution) is the most straightforward to implement. This solution has two interesting aspects:

- The shortest path solution is a unique solution, because for any evacuee there exists normally only one shortest path.
- The flow in the shortest path solution is confluent, meaning that at any node all the flow leaves all the time over one single edge (see, e.g. Chen et al. (2004) for a formal definition). This would make it straightforward to implement the shortest path solution to a real world scenario. One would only need to establish evacuation signs at each crossing of the street network.

However, the disadvantage of the shortest path solution is that it does not take congestion into consideration. As a result the expected evacuation time can be much less than the resulting evacuation time. In this work the shortest path solution will be investigated through different scenarios and it will be shown that the shortest path solution does not solve the evacuation problem.

The second routing approach that has been discussed is the Nash equilibrium approach (NE approach). The NE approach has the advantage that nobody can gain by unilateral deviation and therefore nobody has an incentive to deviate once a Nash equilibrium has been established. In a real world situation a Nash equilibrium can be established by appropriate training.

The third routing strategy that has been discussed is the marginal social cost based approach (MSCB approach), where the system travel time is approximately minimized. However, this approach requires cooperative behavior (individual minimization of the social impact of one's behavior) and does therefore not follow an intrinsic motivation of the evacuees, meaning this kind of behavior has to be enforced externally. This makes the MSCB approach a more efficient approach compared to NE approach but at the same time also more unfair and harder to implement.

Independent of the routing strategy, time-dependent aspects have also to be considered, otherwise an apparently qualified routing solution might lead evacuees out too late of the hazard zone. Time-dependent aspects that for example occur in inundation related evacuations can be modeled as time-dependent networks. In a time-dependent network, a link is only as long passable as long it is not blocked due to the spreading of the threat. The blockages of links as a time-dependent function of the threat's spreading works only as long as the spatiotemporal aspects are known beforehand. For that reason a risk reducing strategy has been proposed. The risk reducing strategy tries to avoid risky evacuation paths under uncertainty of the actual timing of the threat's spreading.

In many situations even the best evacuation strategy might not leave enough time to evacuate everyone. In those situations, additional shelters inside the hazard zone can help to improve the situation. If the evacuation problem is seen as a network flow problem shelters are sinks with limited storage capacity. When it comes to sinks with limited storage capacity, common solution approaches are not longer applicable. The discussed shelter assignment algorithms find a shelter assignment for every evacuee with reasonable computational effort.

In the subsequent chapters the three discussed routing strategies (SP solution NE approach and MSCB approach) will be tested on a real world scenario. The simplest evacuation that does not consider any time-dependent aspects of the threat is discussed in Chapter 4. This chapter also introduces

the simulation framework and evacuation scenario in detail. Simulation results and implementation details of the time-dependent networks are discussed in Chapter 5. The risk-reducing evacuation strategy that also works if the actual advance warning time is uncertain is introduced in Chapter 6. The shelter assignment problem is discussed in Chapter 7 based on a given set of shelter buildings.

Chapter 4

Basic evacuation simulation approach

This chapter introduces a basic approach of a multi-agent evacuation simulation. An evacuation simulation has a lot in common with transport simulations under regular conditions. Each evacuee has an origin. The origins are the whereabouts of the people at the time when the evacuation starts. Each evacuee has also a destination. However, contrary to transport simulations under regular conditions, the destinations are not known beforehand, since every location outside the hazard zone is seen as to be safe and could therefore be a possible destination. Consequently, the destinations are part of the simulation result itself and not part of the input data. The other important difference is the existence of a hazard zone that has to be evacuated. The hazard zone is defined by the (expected) threat for which the evacuation simulation has to be performed. Thus, the main differences of an evacuation simulation to a transport simulation under regular conditions are:

- The absence of predefined destinations for the evacuees.
- The existence of a hazard zone that has to be evacuated.

As discussed in Chapter 1 the three, sometimes conflicting, objectives in an evacuation context are efficiency, fairness and risk avoidance. Formal definitions and solution approaches to these objectives have been discussed in the previous chapter. This chapter addresses the fairness and efficiency objective based on a basic scenario in a multi-agent evacuation simulation performed with the MATSim framework. MATSim stands for **M**ulti-**A**gent **T**ransport **S**imulation and provides a toolbox to implement large-scale agent-based transport simulations[1].

The remainder of this chapter is organized as follows. Chapter 4.1 introduces MATSim and the extensions that has been made in order to perform evacuation simulations. The performance of the simulation will be demonstrated based on a tsunami evacuation scenario for the city of Padang. The scenario description is given in Chapter 4.2. Simulation results are discussed

[1]For more information see http://matsim.org (accessed December 2010)

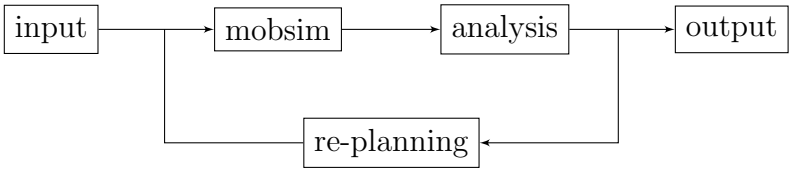


Figure 4.1: General work flow of a MATSim run.

in Chapter 4.3 followed by a brief discussion in Chapter 4.4 and conclusion in Chapter 4.5. The material in this chapter was partially published in Lämmel et al. (2009, 2010b).

4.1 Implementation

The simulation framework is based on MATSim. MATSim is an iterative learning framework for transport problems. MATSim consists of several modules. The modules and the general work flow of the learning cycle are illustrated in Figure 4.1.

A simulation starts with an initial demand as *input*. In the base case the *input* consists of a simulation network and a set of agents. Every agent corresponds to a person in the real world. Each agent has an initial plan. A plan in its simplest form consists of an origin, a destination and a route from the origin to the destination. The agents' plans are executed in the traffic flow simulator, also called mobility simulation (*mobsim*). Afterwards the *analysis* module calculates the score of each agent's plan based on its performance in the *mobsim*. In addition, the *analysis* module also aggregates the experienced travel costs. These costs are needed for *re-planning*. After the *analysis* is finished MATSim can either terminate if the predetermined number of learning iterations has been reached, or continue the learning cycle by running the *re-planning* module. During *re-planning* the agents adapt their plans based on the experienced travel costs or select already existing plans out of their memory for renewed execution. After a simulation run is finished, *output* data will be dumped so it can be used for further appraisal.

The general work flow of MATSim also works for evacuation scenarios. However several modules have to be adapted depending on the desired objective. As mentioned above, in this chapter the Nash equilibrium approach (NE approach) and the marginal social cost based approach (MSCB approach) will be addressed. As discussed in Chapter 3 exact solutions are hard to calculate. However, Algorithms 2 (NE approach) and 3 (MSCB approach) are giving approximate solutions to these problems. The NE approach corresponds to the

standard MATSim approach of finding routing solutions. Nevertheless, also the MSCB approach can be implemented within the MATSim framework by adaptation of the underlying modules. In the following, general implementation details and necessary adaptations are discussed.

4.1.1 Input data

4.1.1.1 Simulation network

The simulation network is an abstraction from the real world street network, where each street segment defines a link and each crossing defines a node. From a formal standpoint the network is a graph $G = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is a finite set of nodes and $\mathcal{A}$ is a finite set of unidirectional links. In MATSim the physics of the network flow are defined on the links. The defining characteristics are:

- The **length** of a link roughly defines the distance between its originating node and its target node.
- The **free speed** of a link defines the speed that will be achieved on that link if there is no congestion. It is a characteristic of the underlying queue model (which will be described later) that the free flow speed is a parameter of the links and not of the traveling agents.
- The **flow capacity** parameter defines the number of agents that can leave the link per time unit. Thus, the flow capacity of a link limits the maximum outflow. It corresponds to the flow capacity of the street segment's bottleneck.
- The **number of lanes** as a link's parameter does not influence the flow directly but limits the storage capacity of the link (i.e. the maximum number of vehicles that fit on the link). The storage capacity is defined as number of lanes times length of the link divided by length of a vehicle, where in the underlying queue model all vehicles are assumed to be of equal length.

A detailed description about the interrelation of the different parameters and their influence on the traffic flow is given later on in Chapter 4.1.2.

In the given context the simulation network represents the evacuation area. In the case of a vehicular evacuation this network consists of all accessible streets. Correspondingly, for a pedestrian evacuation the links in the simulation network also consist of squares and sidewalks. As discussed above, in an evacuation scenario there is not necessarily a predefined destination for

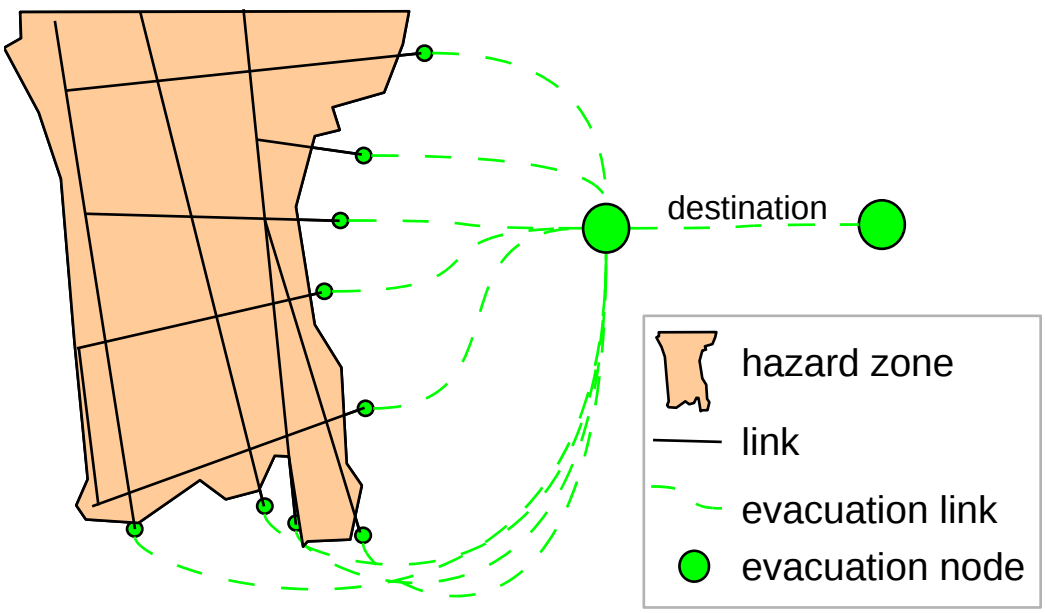


Figure 4.2: Schematic representation of an evacuation network.

the evacuees. Instead every location outside the hazard zone could be a destination, i.e. the evacuation problem is a multi-destination problem, which is not straightforward to solve by existing routing algorithms. However, in Chapter 3.1.1 it has been illustrated how the multi-destination problem can be reduced to a single-destination problem. This can be done by adding a special node as super sink to the network and connecting all safe locations to this super sink by zero cost links. Having done so, all evacuees are to be routed to the super sink. In MATSim, however, a journey starts and ends on a link and not at a node. For that reason the additional super sink has to be a link and not a node. An illustration of a MATSim evacuation network is depicted in Figure 4.2. Taking a basic MATSim network, an evacuation network can be generated in 3 steps:

1. Classify all nodes into "normal" (nodes within the evacuation area), "redundant" (nodes outside the evacuation area, not directly reachable from inside), and "safe" nodes (nodes outside the evacuation area but directly reachable from inside).

2. Remove all redundant nodes and their incoming and outgoing links.

3. Create evacuation nodes and links (according to Figure 4.2).

This procedure ends up in a network that serves as an input for MATSim. The network is stored in an XML file. Listing 1 shows a sample evacuation network in its self-explanatory XML representation.

Listing 1 Sample network with 5 nodes and 7 links. Nodes with $id = 0$ and $id = 1$ are within the evacuation area. Nodes with $id = 2$ and $id = 3$ are safe nodes. Their outgoing links are evacuation links with unlimited flow capacity (indicated by $capacity = 100000$).

```
<nodes>
  <node id="0" x="0" y="0" />
  <node id="1" x="0" y="10" />
  <node id="2" x="10" y="0" />
  <node id="3" x="10" y="10" />
  <node id="en1" x="15" y="15" />
  <node id="en2" x="20" y="15" />
</nodes>
<links>
 <link id="0" from="0" to="2" length="10" freespeed="1.66"
   capacity="4" permlanes="6"  />
 <link id="1" from="0" to="3" length="14.14" freespeed="1.66"
   capacity="2" permlanes="3"  />
 <link id="2" from="1" to="2" length="14.14" freespeed="1.66"
   capacity="5" permlanes="7"  />
 <link id="3" from="1" to="3" length="10" freespeed="1.66"
   capacity="1" permlanes="1.5"  />
 <link id="el1" from="2" to="en1" length="10" freespeed="100000"
   capacity="100000" permlanes="1"  />
 <link id="el2" from="2" to="en1" length="10" freespeed="100000"
   capacity="100000" permlanes="1"  />
 <link id="el3" from="en1" to="en2" length="10" freespeed="100000"
   capacity="100000" permlanes="1"  />
</links>
```

4.1.1.2 Synthetic population, agents and plans

The synthetic population is an abstraction of the real world population that performs their trips in the area of interest, i.e. the area of the simulation network. A synthetic population is a set of individuals, which is based on existing information such as census data or surveys. The synthetic individuals are called agents. Every agent possesses one or several daily plans in her

memory. A plan comprises activities and legs. Activities describe the agents' daily activities like work, leisure or shopping performed on specific locations for a given time span. A leg describes the trip from one activity to the next. In the MATSim logic a plan starts with an activity followed by an arbitrary number of leg-activity pairs. For instance, if one wants to model commuter traffic only, all agents would have a plan consisting of the activity/leg chain *home-work-home*. Important attributes of an agent's plan are:

- The **score** of the plan, which is calculated in the *analysis* module. In the simplest case, if the agents are to minimize their individual travel times, the score is inversely proportional to the travel time.
- The **activity type** describes which kind of activity it is (e.g. *home* or *work*)
- The **activity location** describes where the activity is to be performed.
- The **activity end time** tells when an activity ends. If a leg-activity pair follows an activity, then the agent starts to travel to the next activity when the **activity end time** is reached.
- The **leg mode** describes the mode of transport the agent uses for the next trip. As it is now in MATSim, only the *car* mode makes sense for evacuations. Other modes of transport, therefore, will neither considered nor discussed in this work. However, the aim of this work is to develop a pedestrian evacuation framework. For that reason, the pedestrians are represented by cars with some special physical parameters. Details are discussed later on in Chapter 4.1.2.
- The **route** of a leg describes the route connecting the encompassing activities. A **route** is described by a sequence of nodes $n_1, n_2, \ldots, n_n$, where each pair n_i, n_{i+1} is connected by a link. This formulation is unambiguous as long as there is at most one link connecting nodes n_i, n_j. If there is more than one link connecting nodes n_i, n_j one would either have to switch to a link based route representation or one could split up links and add dummy nodes in way to make the node based route representation unambiguous.

An agent can posses several plans in her memory. In each iteration of the *mobsim* module only one plan per agent is executed. The plan that is to be executed has to be selected before the mobility simulation starts. The selection is performed in the *re-planning* module. The plan selection depends usually on the score of the plans. Details are discussed in Chapter 4.1.4

For evacuations, the synthetic population is a collection of all affected synthetic evacuees. An evacuation plan consist of two activities connected by a leg. The first activity is the activity that the agent performs before the evacuation starts. In other words, the whereabouts of the agents at the beginning of the evacuation are encoded in a *pre-evacuation* activity. This activity is followed by a leg, which describes the evacuation route. Finally, the leg is followed by a *post-evacuation* activity that is located, as discussed, in the super sink. For the evacuation itself only the leg (i.e. evacuation route) is of interest. However, MATSim always needs a leg to be encompassed by activities. Therefore the *pre-evacuation* and the *post-evacuation* activity also belong to an evacuation plan. Individual behavioral parameter regarding the departure time are not explicitly modeled, instead every agent starts her evacuation immediately after the quake. For that reason the departure time is uniform for all agents and plans. The agents' plans are stored in a XML file. A sample plan in its XML notation is shown in Listing 2. The evacuation

Listing 2 A sample plan where the evacuation starts at 03:00 AM at link 2492. The evacuation route leads via node 148156, 148157, 148158 and en1 to the super sink link el1.

```
<plan score="-0.85" selected="yes">
  <act type="pre-evacuation" link="2492" end_time="03:00:00" />
  <leg mode="car" dep_time="03:00:00" trav_time="00:08:36"
      arr_time="03:08:36">
    <route dist="862.64" trav_time="00:08:36">
      148156 148157 148158 en1
    </route>
  </leg>
  <act type="post-evacuation" link="el1" />
</plan>
```

routes in the initial plans are calculated with a simple shortest path router as discussed in Chapter 3.1.1.

4.1.2 Mobility simulation

In the mobility simulation the actual selected agents' plans are executed. This corresponds to step 3.(a) in Algorithm 2 and Algorithm 3, respectively. The mobility simulation is implemented as a queue simulation, where each street (link) is represented as a FIFO (first-in first-out) queue with three restrictions (Gawron, 1998). First, each agent has to remain for a certain time on the link,

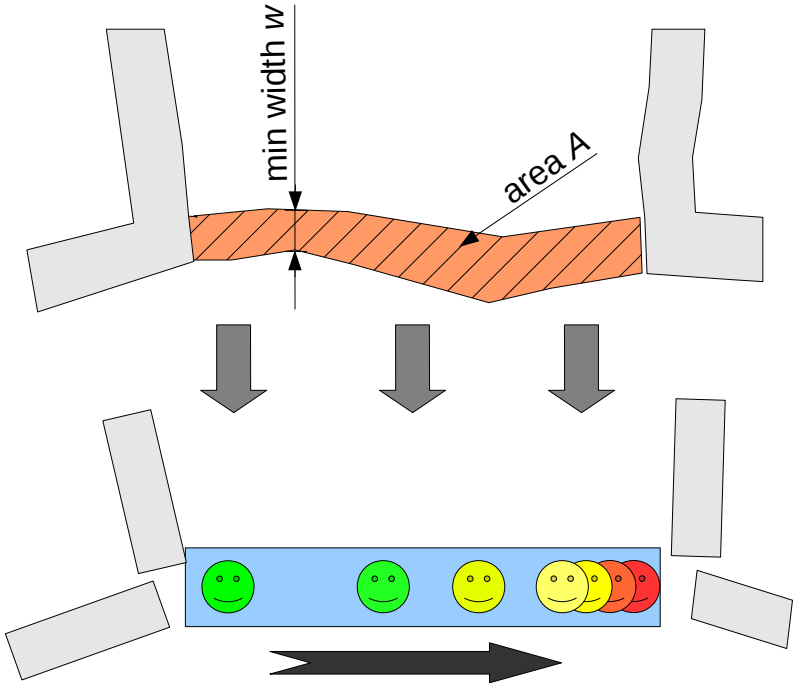


Figure 4.3: Illustration of the creation and functioning of a queue link.

corresponding to the free speed travel time. Second, a link flow capacity is defined, which limits the outflow from the link. If, in any given time step, that capacity is used up, no more agents can leave the link. Finally, a link storage capacity is defined that limits the number of agents on the link. If it is filled up, no more agents can enter this link. The difference to standard queueing theory is that agents (particles) are not dropped but spill back, causing congestion. The queue model—and, by implication, MATSim too—originally was designed for vehicular traffic simulations only. By now the queue model has not been applied to pedestrian simulations. Nevertheless, the evacuation simulation in this work is designed for pedestrians. In principle pedestrian traffic flow relies on the same parameters as vehicular traffic flow. For instance, for both one can define a free flow speed, a storage capacity for links, and a flow capacity in bottlenecks. The derivation of the queue model for pedestrians is given below.

An illustration of the queue model is shown in Figure 4.3. This figure shows the abstraction from the original (polygon based) representation of a street segment to a link with a queue of travelers on it. Important parameters that are defined by the polygon of the street segment are:

- Link area A

- Link length l
- Link minimum width w

There are efficient methods to calculate the area of an arbitrary polygon[2]. The length of a street segment essentially is the length of the median, which can be derived from the medial axis of a polygon (see, e.g., (Preparata, 1977) for a "medial axis" algorithm). Given the median of the street segment its minimum width corresponds to the minimum width over all vertices to the opposite edge measured perpendicular to the median. The flow related parameters of a link a are the length l_a, the free flow travel time τ_a^{free}, the flow capacity q_a and the storage capacity c_a. The storage capacity parameter is not directly derived from the original polygon representation of a link. Historically the storage capacity in MATSim is defined as the number of vehicles that fit on a link and is determined by $c_a = n_a^{lanes} * l_a / l^{veh}$, where n_a^{lanes} denotes the number of lanes and l^{veh} the length of the space occupied by an average vehicle. In order to use the MATSim traffic flow model for pedestrians the length of the space occupied by an average pedestrian (l^{ped}) and the number of "pedestrian lanes" need to be known. However, pedestrians are not exactly walking in lanes. According to Weidmann (1993) one nevertheless can define an average lane width of $w^{ped} = 0.71\,m$. The storage capacity for pedestrians usually is given in persons per area. In Weidmann's work a pedestrian flow comes to a stand still at a density of $\rho^{max} = 5.4\,/m^2$. The missing parameter to calculate the number of pedestrian lanes is the length of the space occupied by an average pedestrian (l^{ped}). Numerical speaking it must be:

$$l^{ped} = \frac{1}{5.4\,\frac{1}{m^2} * 0.71\,m} \approx 0.26\,m \tag{4.1}$$

Given these parameters, the number of lanes can by calculated as follows:

$$n_a^{lanes} = \frac{c_a * l^{ped}}{l_a} = \frac{\rho^{max} * A * l^{ped}}{l_a} \tag{4.2}$$

The free flow speed is set to 1.66 m/s. Consequently, the free flow travel time of link a is $\tau_a^{free} = l_a/1.66\ m/s$. The free flow speed is slightly higher then the 1.34 m/s proposed by Weidmann, but the values presented by Weidmann reflect the pedestrian flow under normal conditions and not in a case of emergency. The flow capacity q_a of link a with minimum width w_a is set to:

$$q_a = w_a * 1.33 \frac{1}{m * s}, \tag{4.3}$$

[2]See, e.g., http://en.wikipedia.org/wiki/Polygon (accessed June 2010)

which reflects a generally accepted value (see, e.g., Predtetschenski and Milinski, 1978). The queue model generates a speed-density relationship comparable to $v = \min[v^{free}, K/\rho]$, where $K = l^{ped} * q/n^{lanes}$ (Simon and Nagel, 1999).

4.1.3 Analysis

The *analysis* module collects all relevant data during the mobility simulation and analyzes them after the mobility simulation finishes. MATSim provides an events framework for that purpose. The most basic events are the *link-enter* and *link-leave* events that are emitted whenever an agent enters or leaves a link. From these events it is straightforward to calculate the experienced travel times and external costs. For the NE approach, discussed in Chapter 3.1.2, the score of a performed plan only depends on the experienced travel time. And for the MSCB approach, discussed in Chapter 3.1.3, the score of a performed plan depends on the experienced travel time and on the caused external costs (as defined in Equation (3.7)). In the following it is referred to as the general term costs for both NE approach and MSCB approach. The score of plan p is calculated by:

$$s(p) = -\beta_{travel} * \sum_{(i,j)\varepsilon p} C_{(i,j)}, \tag{4.4}$$

where (i, j) denotes the link from node n_i to n_j and $(i, j) \in p$ is the set of all links that are part of plan p's route. $C_{(i,j)}$ are the experienced travel costs when traversing link (i, j). Note, the cost term in this equation is given without any time reference as it was introduced in Chapter 3.1.2 for NE approach, and in Chapter 3.1.3 for MSCB approach. This is because the score calculation is based on the actually experienced costs. The parameter β_{travel} is the utility of traveling, it is set to $-6/h$[3]. The score of a plan is important for the plan selection in the *re-planning* module, which will be discussed later. As pointed out, the experienced travel times are used to calculate new routes in the *re-planning* module. The extraction of time-dependent link travel costs corresponds to step 3.(b) in Algorithm 2 and Algorithm 3, respectively. As discussed in Chapter 3.1.2 the time for time-dependent link travel costs is discretized into K slices of length T, meaning the simulation time span is divided into K time-slices of equal length T. For each time-slice an aggregated value of the experienced travel time is stored. The standard MATSim approach

[3]The actual value of this parameter is less important for this work. There is also a parameter called β_{travel} in MATSim. This parameter is set to 0 throughout this work, since it is of no relevance for the underlying problem. The interested reader, however, is referred to (Grether et al., 2010) for a detailed discussion.

is to average the experienced travel times over all agents that enter a link during a given time-slice. The standard time-slice size is $15\,min$, which seems to be reasonable when simulating vehicular traffic over a whole day. When it comes to an evacuation scenario where the entire evacuation time span is less than, let's say, two hours, a time-slice size of $15\,min$ is to coarse to capture the dynamics in travel time changes adequately. However, if the time-slice size is chosen too small it may occur that for some time-slices no agent enters the link in question. Consequently, there would be no measurements for that link and those time-slices. Nevertheless, there could be congestion caused by agents that entered the link in earlier time-slices. To avoid these problems the time-slice size has been set to $3\,min$ for this work, which seems to be a good compromise to have a fine-grained temporal resolution to capture the dynamics on the link adequately but to avoid problems with time-slices without measurements. A consequence of the finer temporal resolution is an increase in the amount of data that has to be stored. This can be handled efficiently by using an associative data structure. Details on this matter are given in Appendix B. Not only the time-slice size but also the aggregation procedure has been modified. As discussed, usually MATSim averages the experienced link travel times. However, in this work the time-dependent link travel times are equal to the worst experienced link travel time in the corresponding time-slice. This is a more pessimistic approach, which means it is likely that the link travel times are systematically overestimated. Such an approach helps to avoid situations where, caused by underestimation of the travel time, an evacuation route does not leave enough time to traverse a critical part of the hazard zone until it becomes impassable. The critical part could for example be a bridge that is only passable for a certain time. This is a very important problem in evacuation situations. Therefore there will be a detailed discussion on this matter in Chapter 5. Also the time-dependent link external costs have to be aggregated over the time-slices. The original MATSim averaging approach has been retained for the aggregation of the time-dependent link external costs with one modification: If there is congestion, then the external costs are usually higher by orders of magnitude than the travel times. A fluctuation in external costs from one iteration to the next would also mean a high fluctuation in total travel costs. Those fluctuations worsen the quality of the solution that can be found with the iterative learning approach. Therefore the changes in time-dependent link external costs over the iterations are smoothed by the method of successive average (MSA, see Robbins and Monro, 1951). MSA is an inductive definition of the mean value for a set of measurements. Let $C^s_{a,i}(k)$ denote the aggregated time-dependent link external costs for link a, time-slice k in learning iteration i that have been smoothed with MSA. The MSA smoothed costs fore *re-planning* are calculated by

if $i == 0$ **then** $C^s_{a,i}(k) = C^s_a(k)$

if $i > 0$ **then** $C^s_{a,i}(k) = i/(i+1) * C^s_{a,i-1}(k) + 1/(i+1) * C^s_a(k)$

For the sake of simplicity the iteration index i will be omitted in the following and the term $C^s_a(k)$ referrers to the MSA smoothed external costs for link a and time-slice k in the corresponding iteration.

4.1.4 Re-planning

The purpose of the *re-planning* module is to revise the agents' plans that have been executed in the *mobsim* module and scored in the *analysis* module. During re-planning the agents produce new best response routes (plans) that will be executed in the next iteration. However, if all agents would produce new best response plans in every iteration than the system would fluctuate. Therefore, only a fraction of all agents will produce new plans by executing the re-routing strategy. This procedure corresponds to step 3.(c) in Algorithm 2 (NE approach) and Algorithm 3 (MSCB approach), respectively. In the algorithms discussed here each agent is chosen with a certain probability to create a new route. In the application the agents are not chosen with a certain probability but a certain number of agents is randomly chosen out of the population for re-routing. There is an additional modification of the Algorithms. As mentioned, the agents can possess several plans in their memory. Agents that have not been selected for re-routing choose old plans out of their memory. The plan selection strategy is important to stabilize the iterative learning cycle and leads the system to better results.

The `ReRoute` strategy generates new plans with new evacuation routes based on the information of the experienced travel costs from the last iteration. The cost term depends on the objective function (e.g. NE approach and MSCB approach). For the NE approach the costs comprises the time-dependent link travel times. For the MSCB approach the costs comprises the time-dependent link travel times and the time-dependent link external costs (as defined in Equation (3.7)). The re-routing is performed based on a time-dependent least cost path calculator, discussed in Chapter 3.1.2 (NE approach) and Chapter 3.1.3 (MSCB approach), respectively.

The other strategy is called `ChangeExpBeta`. This strategy decides if the just performed plan should be used again, or if a random plan out of the memory should be selected for the next iteration. The probability to change the selected plan is calculated by:

$$P_{change} = min(1, \alpha * e^{\beta*(s(p_{random})-s(p_{current}))/2}) \tag{4.5}$$

This is similar to the probability of accepting a random mutation in the metropolis algorithm (Metropolis et al., 1953), with:

- α: The probability to change if both plans have the same score
- β: A sensitivity parameter
- $s(p_{\{random,current\}})$: The score of the current/random plan

If the system is "well-behaved", this set-up converges to a steady state where the probability that an agent uses plan p_i is

$$P(p_i) = \frac{e^{\beta * s(p_i)}}{\sum_j e^{\beta * s(p_j)}} \ , \tag{4.6}$$

i.e. the standard multinomial logit model (e.g. Ben-Akiva and Lerman, 1985).

Each strategy is selected with a certain probability. These probabilities are assigned before the simulation starts. Typically, `ReRoute` is called with a relatively small probability, say 10%, and `ChangeExpBeta` is called in the remaining cases. After re-planning every agent has a selected plan that will be executed in the next iteration.

4.1.5 Simulation shutdown and output data

Repeating this iteration cycle of learning, the agents' behavior will move towards a Nash equilibrium (NE approach) or system optimum (MSCB approach), respectively. The term "towards" means that both approaches will never converge to a real Nash equilibrium or system optimum. However, if the learning iterations are repeated often enough the system will move close enough to their respective objectives from the practical point of view. The number of iterations needed until the found solution meets the requirements is not known beforehand. Nevertheless, MATSim relies on a predetermined number of learning iterations. The number of learning iterations, therefore, has either to be determined by an educated guess or by practical experience. After the last iteration is finished MATSim performs a shutdown sequence. During shutdown essential results will be dumped into data files. Important for later appraisal are the plans file containing the final set of evacuation plans and the events file from the last iteration. The events file gives a complete description of the *mobsim*'s execution. Most of the simulation results discussed in this and subsequent chapters are entirely generated from events files.

4.2 Padang scenario

The performance of the simulation framework and the proposed evacuation strategies is demonstrated on a real world scenario for the Indonesian city of Padang. Some key points of the city have already been introduced in Chapter 1. The following section discusses the generation of the input data from existing real world data.

The evacuation simulation depends on several input sources. The Padang scenario is generated from three different types of input data:

- Geographical information derived from remote sensing data (street network, safe places)
- Inundation scenarios (time- and location-dependent flooding information)
- Socio-economic data (e.g. population distribution as a function of time)

Below the input sources, methods of data editing, and integration of the geographical information and socio-economic data are discussed. Since the integration of inundation scenarios is much more complex compared to the rest of the data, it will be discussed separately in Chapter 5.

4.2.1 Geographical information

The geographical information mainly depends on remote sensing data. For the "Last-Mile" project, it was decided to use high resolution satellite imagery made by the Ikonos satellite. Ikonos data features a geometric quality of $1\,m$. The most important geographical input is the information about the street network and all the other walkable areas like open spaces or meadows. Using an object-oriented hierarchical classification approach in combination with manual enhancement, an area-wide and up-to-date land-cover classification has been derived (Taubenböck and Roth, 2007). In a first step the street map has been extracted from the satellite data. This work has been performed by "Last-Mile" project partners and delivered as a so called shapefile[4]. A cut-out of the street map is shown in Figure 4.4. However, in order to make this street map applicable for MATSim it has to be converted into a graph. This has been performed in two steps. In a first step the street map has been segmented into links and nodes. An example how the segmented links and nodes look like is shown in figure 4.5(a). In a second step all redundant

[4]For more information about shapefiles see, e.g., http://en.wikipedia.org/wiki/Shapefile (accessed June 2010)

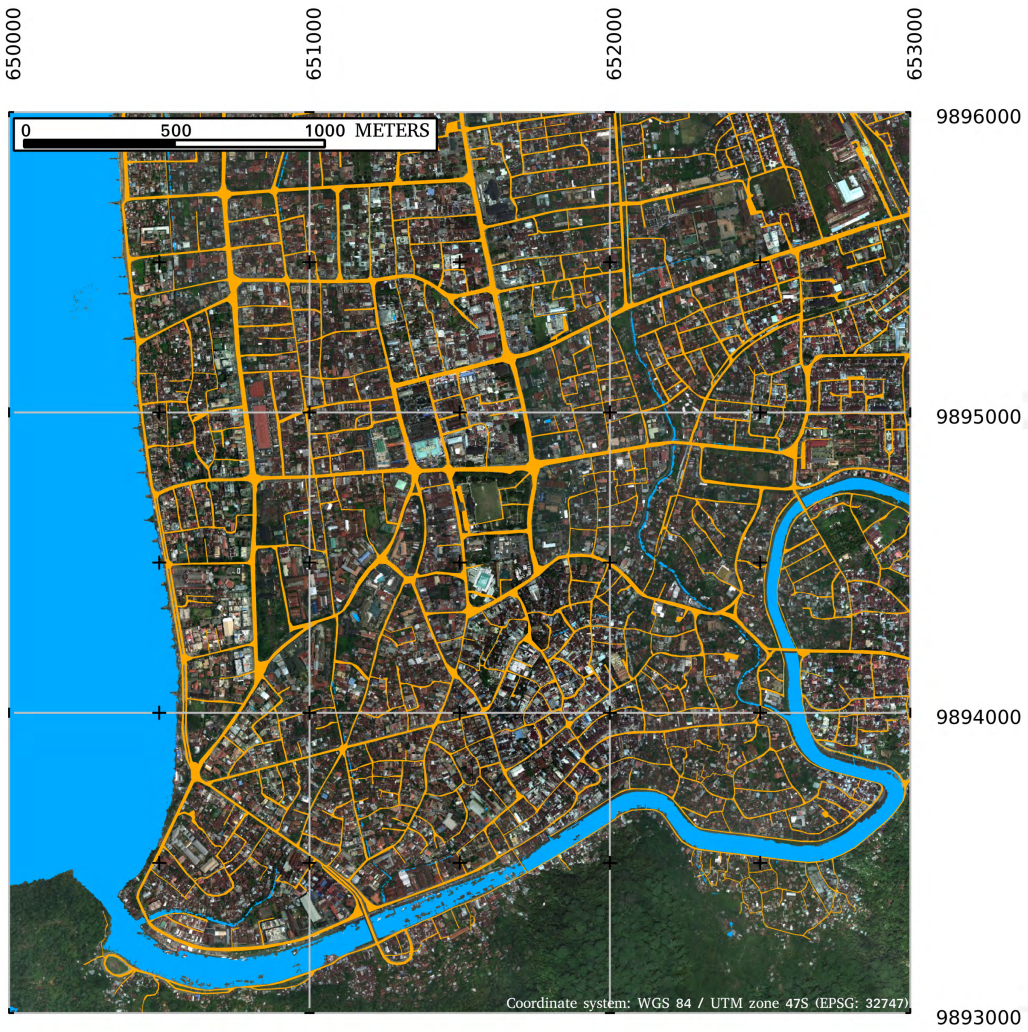
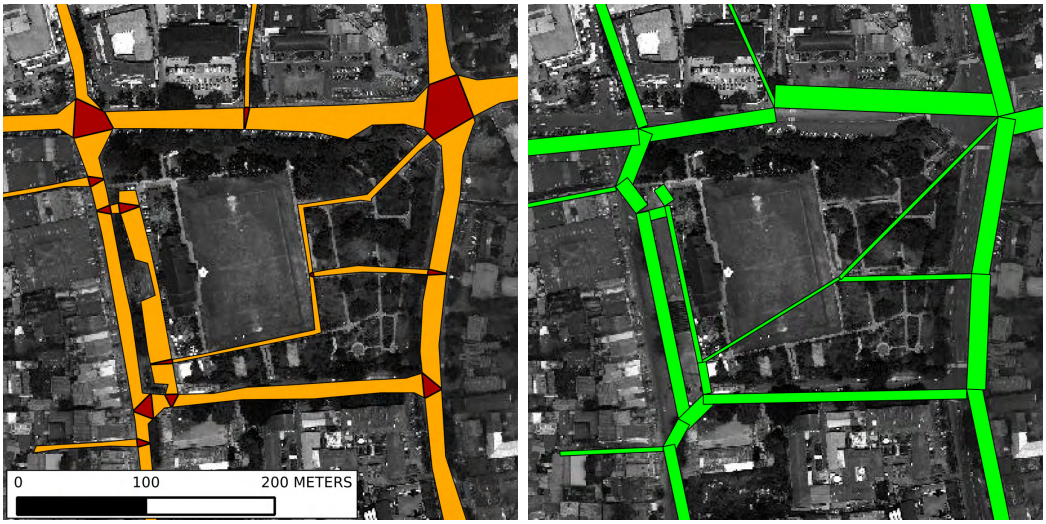


Figure 4.4: Street map of Padang as it has been extracted from satellite imagery.



(a) Links and node based polygon network (b) Graph representation of the network

Figure 4.5: Two steps of network generation.

information, such as the actual shape of street segments, has been stripped off. The only information that has been retained is the original length, width and area of the street segments and the from- and to-node relations; ending in a graph representation of the street network as shown in figure 4.5(b). The resulting network consists of 23 012 unidirectional links and 8 886 nodes.

4.2.2 Socio-economic data

The socio-economic profile of the city has a major impact on the evacuation itself and is therefore an important input for the simulation framework. Moreover, a detailed knowledge on the socio-economic vulnerability of the population will also help to give recommendations of preparedness strategies for potential tsunami events. In the scope of the "Last-Mile" project, a detailed assessment of socio-economic vulnerability components according to the tsunami early warning chain is carried out. It focuses, among others, on dynamic exposure based on performed daily activities in the different tsunami hazard zones, predicts the evacuation behavior and the evacuation capability of different social groups. A detailed discussion on that work can be found in (Setiadi et al., 2010). In order to model complex behavioral aspects of the evacuees, spatially highly resolved data is needed. Of particular interest would be data on evacuation behavior from former tsunami related evacuations. The last two major earthquakes took place on September 12th 2007 and on September 30th 2009. While the 2007 earthquake with a moment magnitude of 7.9

M_w was located about 185 km south of Padang, the 2009 earthquake with a moment magnitude of 7.5 M_w was located only about 60 km west of Padang. The former earthquake caused medium damage in Padang and there are no reports about casualties. The latter one with its closeness to the city caused heavy damage and more than 1000 casualties have been reported[5]. In both cases tsunami warnings were triggered and later recalled. In the aftermath of both earthquakes the Gesellschaft für Technische Zusammenarbeit (GTZ) conducted surveys where about 200 affected people have been interviewed regarding their evacuation behavior and experiences (Hoppe, 2007; Hoppe and Marhadiko, 2010). The result of these surveys together with other findings of the "Last-Mile" project (see, e.g., (Setiadi et al., 2010)) could be taken to model autonomous agents with high level behavior to reflect the population of Padang in an appropriate manner. However, since the focus in this work is less on individual behavioral aspects and more on the general solving of the evacuation problem only data about the spatio-temporal distribution of the population has become an input source for the simulation framework. The spatio-temporal population data has been provided as a shapefile. The shapefile comprises a building mask for the city with 56 592 building shapes. For each building an estimation of the number of persons that are inside the building is given for the night, morning and afternoon. The total number of persons for the different times of day is shown in table 4.1. This work, however, only discusses the morning scenario. For each building of the buildings mask the given number of agents with initial plans have been generated and compiled to the input plans file. The origin or *pre-evacuation* activity of an agent's initial evacuation plan is located at the link next to the building's centroid. The destination or *post-evacuation* activity is located in the super sink, as discussed in Chapter 4.1.1.2. Figure 4.6 shows a map with the buildings mask as an overlay. The spatial extent of the buildings mask characterizes the evacuation area. The boundary to the north and to the south corresponds to the boundary of the study area in the "Last-Mile" project, where the expansion to the east is chosen to cover all the area with an elevation of less than $10\,m$ above sea level. This approach is chosen based on the assumption that even the strongest tsunami will not inundate areas with an elevation of more than $10\,m$. Later on the expansion of the area will be shrunk based on results of microscopic inundation simulations. However, since in this chapter inundation effects are neglected it was decided to stick with the simpler "$10\,m$" approach.

[5]For more information see e.g. http://earthquake.usgs.gov/earthquakes/world/world_deaths.php (accessed June 2010)

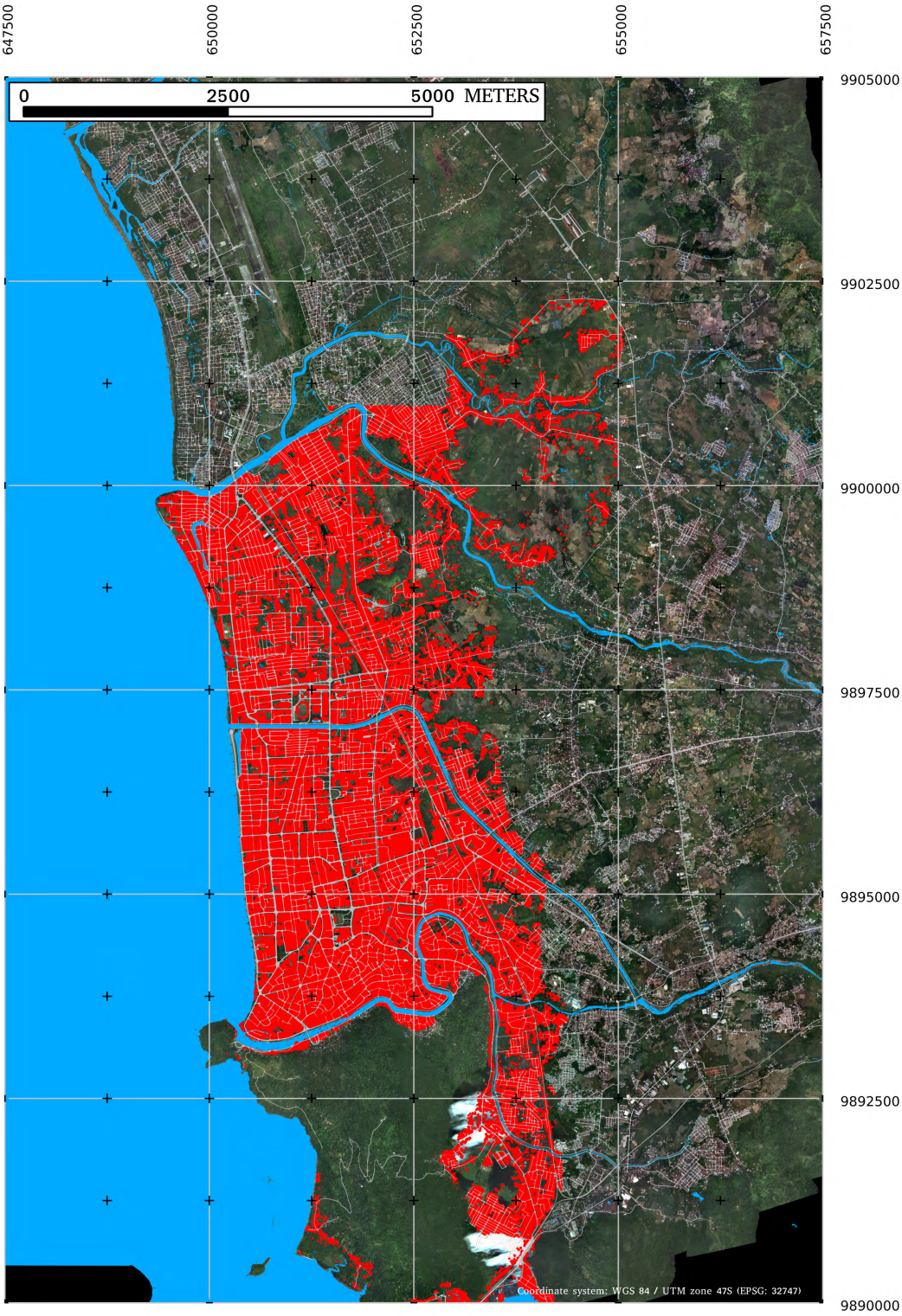


Figure 4.6: The city map with buildings mask indicating the evacuation area. Buildings are red.

time of day	number of persons
night	267 500
morning	328 617
afternoon	291 517

Table 4.1: Number of persons inside the evacuation area for the night, morning and afternoon situation.

4.3 Experiments

4.3.1 Comparison with a combinatorial optimization solution

Three different routing approaches are introduced in Chapter 3.1. The routing approaches are:

- The shortest path solution (SP solution), which puts everyone on the shortest path.
- A Nash equilibrium approach (NE approach). In a Nash equilibrium no one can gain by unilateral deviation from her current evacuation route.
- A marginal social cost based approach (MSCB approach). Minimizing the marginal social costs also minimizes the average travel time. The minimization of the average travel time leads to the system optimum.

Besides the shortest path solution, which is trivial to compute, it is not clear how good the proposed routing approaches meet their respective objective. For that reason it would be of interest to compare the simulation based results with results from combinatorial optimization. As already mentioned (see Chapter 3.1.2 or (Köhler et al., 2009)), there exist no mathematical programming algorithm that can compute a Nash equilibrium for a dynamic network of non-trivial size. However, there are combinatorial optimization approaches for optimal flows, which are capable to deal with instances of the Padang scenario's size. Dressler et al. (2011) discussed the performance of the NE approach and the MSCB approach compared to an earliest arrival flow computed by a combinatorial optimization algorithm. To be exact, the earliest arrival flow (EAF) is computed on a deterministic abstraction of the physical model used in MATSim's *mobsim* module. Afterwards a MATSim plans file is generated from the earliest arrival flow, which in turn is used to perform one iteration in MATSim (EAF-MATSim). Thus, there are two solutions with

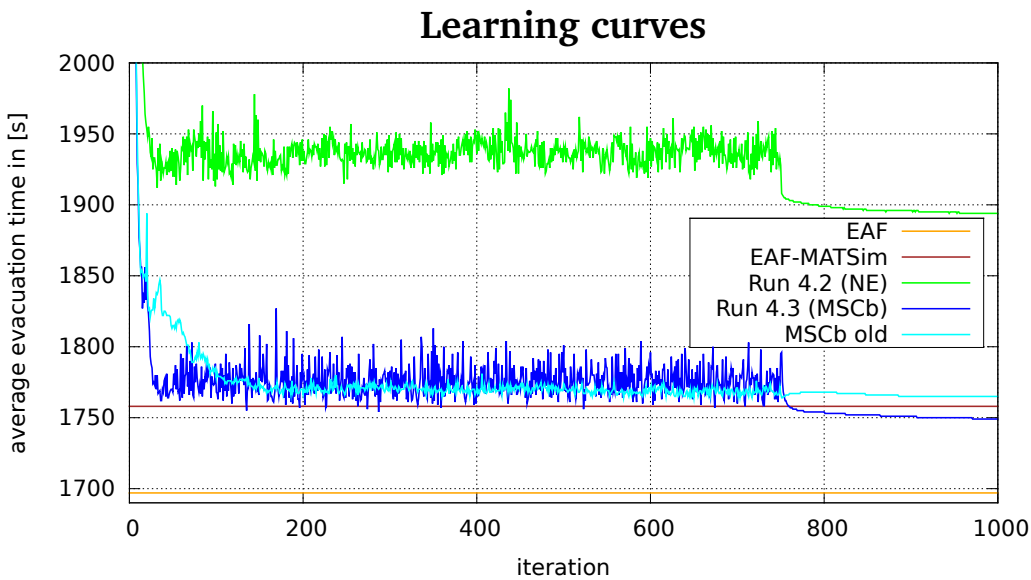


Figure 4.7: Average evacuation time over learning iterations of the NE approach, the MSCB approach and the old version of the MSCB approach compared to the earliest arrival flow solution (EAF) and its validation in MATSim (EAF-MATSim).

which the NE approach and the MSCB approach can be compared. Furthermore, another interesting comparison is the performance of the MSCB approach as introduced in Chapter 3.1.3 with the former approach (MSCB-old) introduced in (Lämmel and Flötteröd, 2009). Both approaches differ in their respective method of external costs approximation. The MSCB approach calculates the external costs according to Equation 3.7 and for MSCB-old the external costs are calculated according to Equation 3.4. The results presented in (Dressler et al., 2011) are based on the old approach and not on the newly introduced MSCB approach. Simulation runs have been performed based on the scenario introduced in this chapter. The number of learning iterations for the simulation based approaches (*Run 4.2* (NE approach), *Run 4.3* (MSCB approach) and MSCB-old) is 1 000. More setup details are discussed in the next section. The average evacuation time over learning iterations for the NE approach, the MSCB approach and the old version of the MSCB approach is shown in Figure 4.7. The average evacuation time for the EAF and the EAF-MATSim is also plotted over the 1 000 learning iterations in this figure even though no learning has been performed for these solutions.

It is shown that the newly introduced MSCB approach not only outperforms the NE approach and the MSCB-old but also the EAF-MATSim, which

is hardly surprising, since the average evacuation time for EAF-MATSim results from plans that have been calculated on deterministic abstraction of the physical model used in MATSim's *mobsim* module. In the abstracted model the average evacuation time is much smaller (EAF curve). The jump at iteration 750 followed by much smother curves results from the deactivation of the `ReRoute` strategy at iteration 750. This results in a stationary process, which is, however, limited to the plans that each agent already possesses[6].

4.3.2 Padang scenario

The basic evacuation setup has been investigated through three different runs. *Run 4.1* implements the basic shortest path solution (SP solution), *Run 4.2* the basic Nash equilibrium approach (NE approach), and *Run 4.3* the basic marginal social cost based approach (MSCB approach). The evacuation area encloses the area with an elevation of up to 10 m above see level. The synthetic population consist of 328 617 agents. This number corresponds to the morning population in the evacuation area. As discussed in Chapter 4.2.2, there is also data for the night time and afternoon population available. However, in this thesis the simulation approaches will be discussed based on the morning population only. *Run 4.1* (SP solution) corresponds to the first iteration of *Run 4.2* (NE approach) and *Run 4.3* (MSCB approach). This means, that no extra run has to be performed in order to get the results for the shortest path solution. However, for consistency reasons the basic shortest path solution is called *Run 4.1*. *Run 4.2* and *Run 4.3* have been performed with the following setup:

- For the first 750 iteration 10% of the population produce new routes using the `ReRoute` module during *re-planning*. For the remainder (90% of the population) an existing plan is chosen out of their memory by the `ChangeExpBeta` module.
- From iteration 751 `ChangeExpBeta` is performed for all agents, meaning no new routes are produced from iteration 751 on.
- The simulation stops after iteration 1000 finishes.

The simulation runs have been performed on *cluster 7* hosted at the math institute of TU Berlin[7]. Each node of the cluster is equipped with 2 QuadCore-

[6]Implications of this are not exactly clear at the moment. However, this approach reduces the average evacuation time significantly. (see also http://matsim.org/node/542 – accessed March 2011)

[7]For more information about the cluster look at http://www.naturwissenschaften.tu-berlin.de/iuk/computeserver/hardware/hardwareuebersicht (in German, accessed July 2010)

parameter	reading *Run 4.2*	reading *Run 4.3*
wall time	2:00:42:50	4:03:13:14
CPU time	2:07:15:51	4:13:22:23
max memory	8.315 GiB	8.272 GiB

Table 4.2: Performance measurement of *Run 4.2* and *Run 4.3*.

Xeon X5550 (2.67 GHz) CPU and 24 GiB or 48 GiB RAM. To get an idea of the complexity of the underlying large-scale scenario, measurements of the computing time and memory consumption are important. Table 4.2 gives basic performance measurements for *Run 4.2* and *Run 4.3*. The wall time describes the "real" time that the simulations take, i.e. the result depends on other processes running at the node. For that reason the wall time gives only an estimate how long one has to wait for a simulation run to complete. The CPU time is higher than the wall time. This is because the re-planning modules are running in parallel and so the simulation utilizes more than one CPU. The run time and memory consumption for both runs is reasonable. However, with the social cost optimization enabled (*Run 4.3*), the CPU time doubles. Still, to perform a large-scale simulation for Padang in 4 days seems acceptable.

As discussed, the mobility simulation is stochastic and therefore no fixed point will be found in the iterative dynamic. Instead the simulation will be stopped after no noticeable improvement of the goodness of the found solution can be seen for a while. Therefore a criterion for goodness is needed. Based on the discussion in Chapter 3.1.3 the average evacuation time is an appropriate measure of optimality. The observation of the average evacuation time over the iterations results in learning curves that can be taken to decide heuristically whether the simulation can be stopped. The learning curves for *Run 4.2* and *Run 4.3* in Figure 4.8 indicate that 1000 iterations of learning are sufficient. The figure shows also that, as expected, the average evacuation time for the MSCB approach (*Run 4.3*) is less than the average evacuation time for the NE approach (*Run 4.2*). For comparison the average evacuation time for the SP solution (*Run 4.1*) is also plotted over the 1000 iterations. It is clearly shown that the SP solution is much worse compared to the solutions obtained by the NE approach and MSCB approach. This is because of the negligence of congestion in the SP solution case. The goodness of evacuation solutions can also be investigated based on evacuation curves. An evacuation curves describes the number of evacuated persons over the time. The steeper the gradient of the curve, the higher is the outflow rate (i.e. number of evacuated persons per time). The evacuation curves of the three runs discussed in this

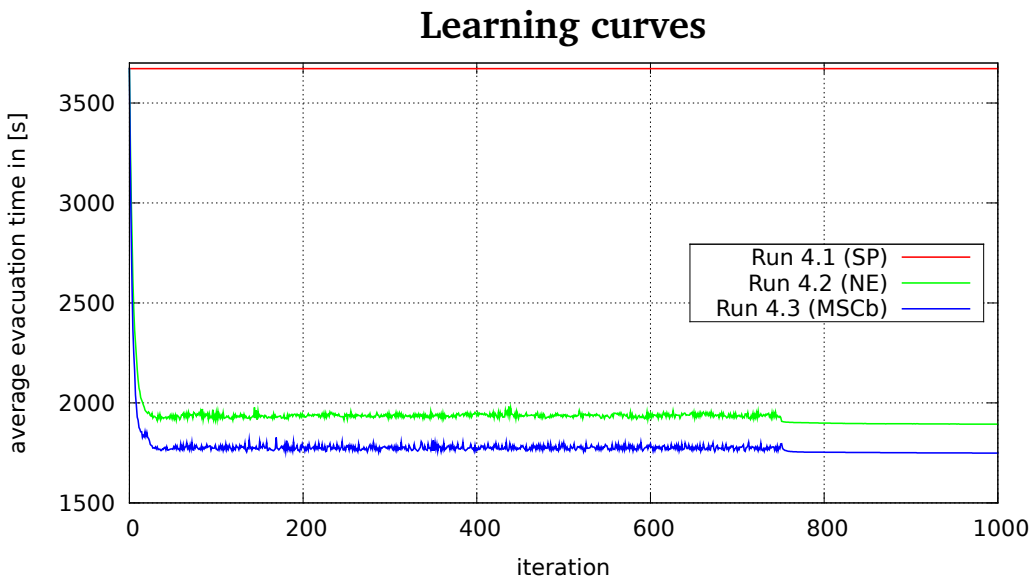


Figure 4.8: Average evacuation time over iteration number for *Run 4.1* (SP solution), *Run 4.2* (NE approach), and *Run 4.3* (MSCB approach). Note, that no learning iterations have been performed for *Run 4.1*. Therefore, its average evacuation time remains constant over the iterations. For the SP solution the average evacuation time is 3 672 seconds, for the NE approach it goes down to round about 1 894 seconds and for the MSCB approach to round about 1749 seconds.

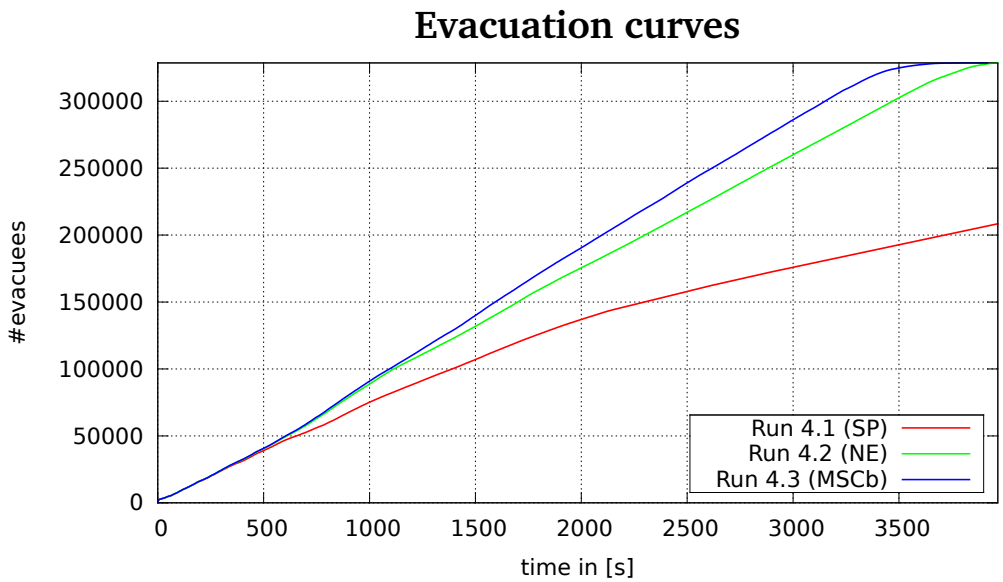


Figure 4.9: Evacuation curves of *Run 4.1* (SP solution), *Run 4.2* (NE approach), and *Run 4.3* (MSCB approach). The SP solution run results in the flattest evacuation curve. This results in an egress time of 14 813 seconds (Note: the curve is truncated). The gradients for the NE approach and the MSCB approach are steeper and both curves almost coincident in the early stages of the evacuation process. However, later on the outflow rate for the NE approach is smaller than for the MSCB approach. This leads to a considerable longer egress time for the NE approach (3 967 seconds) compared to the MSCB approach (3 918 seconds). Another observation is that with the MSCB approach are more agents evacuated at every time step than with the SP solution and the NE approach.

chapter are given in Figure 4.9. The curves of *Run 4.2* and *Run 4.3* correspond to the solution found after 1000 iterations of learning. Again the SP solution performs worst. But even the NE approach needs a lot more time compared to the MSCB approach for a total clearance of the evacuation area. Taking all the results together it can be stated that *Run 4.3* (MSCB approach) has the lowest average evacuation time, the maximum number of evacuated agents at each time step and the lowest total clearance time. However, the promising results discussed so far do not show if any of the approaches is appropriate for an implementation in the real world. One method to appraise the practicability is to visualize the evacuation routes of the population. Therefore MATSim provides an OpenGL based visualizer (Strippgen, 2009) to show the

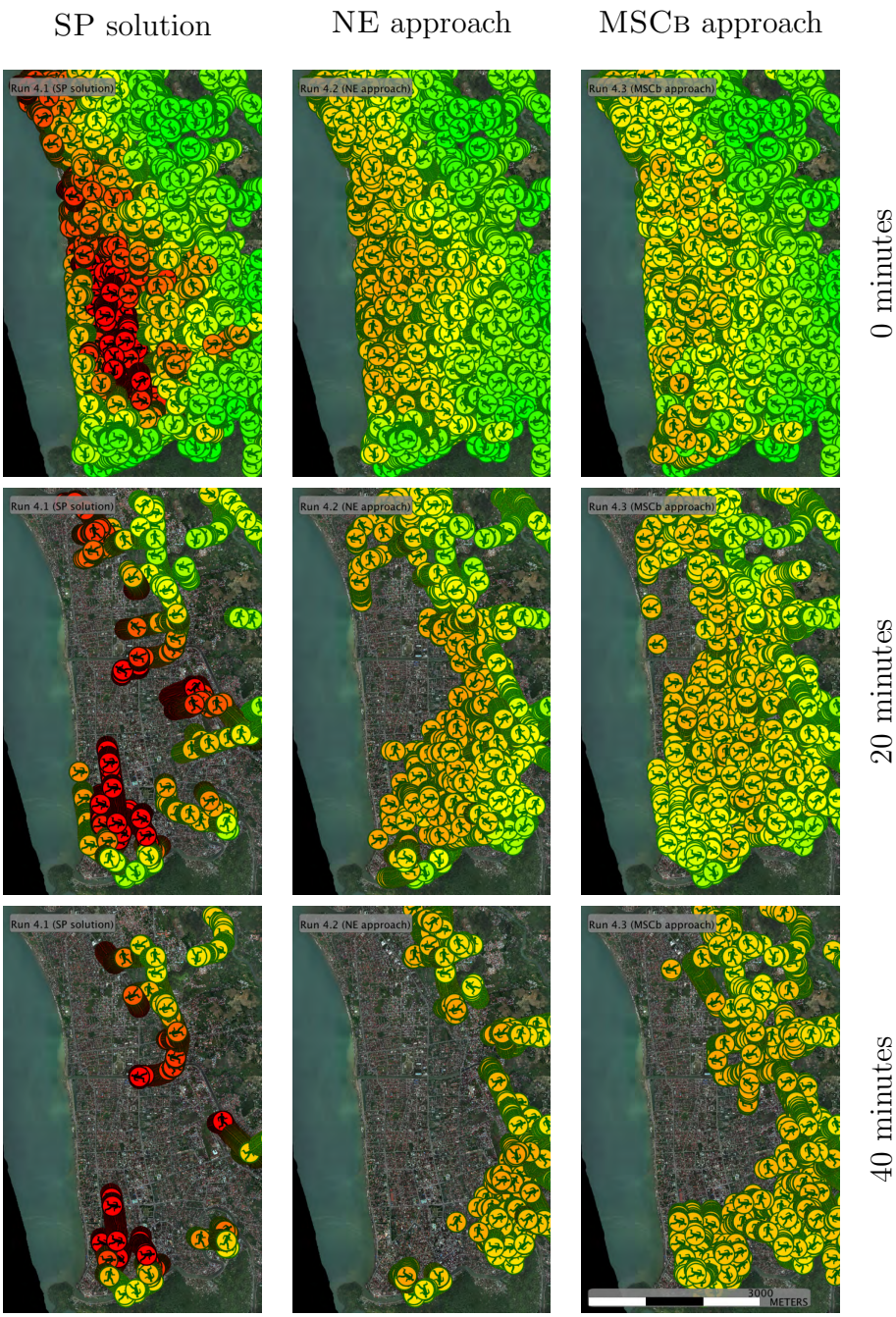


Figure 4.10: Visualizer snapshots of *Run 4.1*, *Run 4.2*, *Run 4.3* at the beginning of the evacuation (top) after 20 minutes of evacuation (center) and after 40 minutes of evacuation (bottom). The agents are colorized depending on the time they need for the evacuation. A green color indicates that they escape quickly while a red color means they need rather long time for the evacuation. It is shown, in all runs are many agents that are evacuating along the coastline, which is obviously not recommendable.

evacuation behavior of every single agent. Visualizer snapshots of *Run 4.1*, *Run 4.2*, and *Run 4.3* are given in Figure 4.10.

The snapshots clearly show that even after 40 minutes for all three runs some agents are still near to the coastline. This is because the agents are taking the least cost path to the safe area and this path leads independent of the approach (SP solution, NE approach or MSCB approach) for many agents along the coastline. This also shows that none of the approaches seems to be recommendable for the real world. This issue will be discussed in the following section.

4.4 Discussion

A first point of discussion is that many agents are evacuating along the coastline. This is because the agents are not "aware" of the tsunami wave that will approach the coastline some 30 minutes after the earthquake. An awareness of the danger could be modeled as high-level decision making of the agents. The agents could reject evacuation routes that put them at risk of being affected by the inundation. However, since the `ReRoute` strategy is based on a least cost router no feasible routes are found for the red agents in Figure 4.10. Therefore the existing cost functions for the various approaches needs to be adapted. This can be done by penalizing routes that are parallel to the coast. Indeed, the penalty has to be time-dependent since some links will be flooded early than others. To make sure that the `ChangeExpBeta` strategy will avoid selecting inappropriate plans the *analysis* module has to penalize such plans too. The integration of this time-dependent aspect into the simulation framework is discussed in Chapter 5. Another interesting finding is that the most naive approach, the shortest path solution, performs worst. This aspect has already been anticipated in Chapter 3. The reason for the bad performance of the SP solution is because the SP solution does not take congestion into consideration. In other words, the SP solution pretends travel times that do not correspond to the resulting ones. Figure 4.11 compares the pretended with the resulting travel times. For the real world this means: The shortest path solution is likely to underestimate the travel time and therefore it is not qualified for evacuation planning.

4.5 Conclusion

MATSim is a powerful simulation tool for large-scale transport simulations. This chapter introduces the MATSim framework briefly. Since MATSim is developed for vehicular traffic under regular conditions several adaptions are

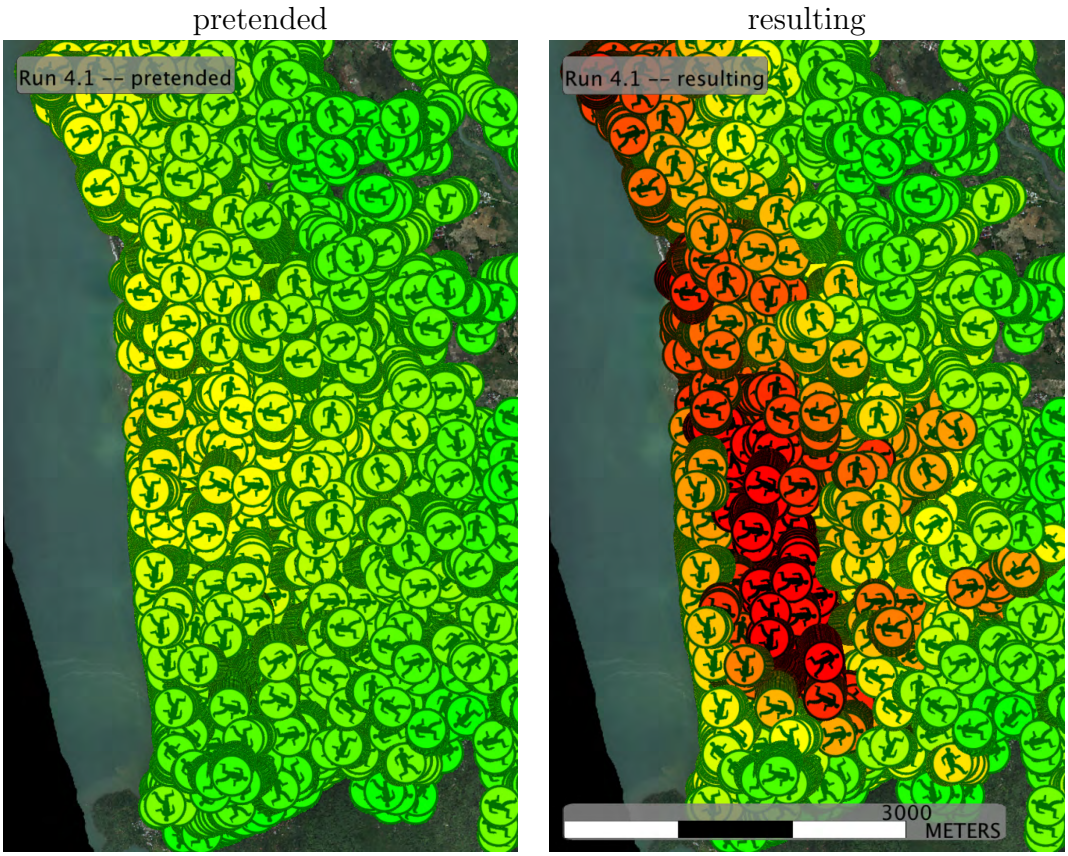


Figure 4.11: Visualizer snapshot comparing the pretended travel times with the resulting ones for the SP solution solution (*Run 4.1*). The agents' color corresponds to the pretended travel times on the left and to resulting ones on the right.

necessary to make it applicable for pedestrian evacuation simulations. The needed adaptations are introduced and the validity is shown by comparison with other well-known and generally accepted pedestrian models. The most important parts of MATSim are the mobility simulation (*mobsim*), the *analysis* module, and the *re-planning* module. Together they form the iterative learning algorithm, which results, depending on the setup, either in a (approximately) Nash equilibrium (NE approach) or in a (approximately) system optimum (MSCB approach). Regardless of whether a Nash equilibrium or the system optimum is desired, for both approaches the iterative learning procedure begins with shortest path routing (SP solution) as the starting solution.

There is a lot of input data that is needed in order to develop a large-scale evacuation simulation for pedestrians. The needed input data has been discussed based on a tsunami related evacuation scenario. The basic input data comprises geographic information and socio-economic data. The geographic information describes the evacuation area and the street network of the city. In this chapter the evacuation area includes all the area with an elevation of less than $10\,m$. This area has been derived from a simple topographic model. The street network has been semi-automatically extracted from satellite imagery. The socio-economic data has been provided as a buildings mask comprising almost 60 000 buildings. For each building the number of persons that are inside the building in the morning hours, in the afternoon, and during the night time are given. This leads to three different scenarios. The morning scenario is studied in detail by different simulation approaches.

The validity of the simulation approaches is demonstrated through comparison of their respective performance with the results of a combinatorial optimization solution. The simulation approaches comprise an NE approach and an MSCB approach. Furthermore, the SP solution has also been tested on the Padang scenario. The SP solution, the NE approach, and the MSCB approach are studied in detail regarding average evacuation time and evacuation curves (evacuated person vs. time). It turned out that the MSCB approach leads to the shortest overall and average evacuation time. This is not surprising but, as discussed in Chapter 3.1.3, a basal attribute of the MSCB approach. Also worth mentioning is that the SP solution significantly underestimates the travel time. This is because of the negligence of congestion. However, the most important finding that with the basic setup many evacuees are fleeing in parallel to the shoreline to reach safe locations. Even if this leads to a faster evacuation such behavior is not desirable. Instead, the evacuees should move away from the approaching tsunami. This undesired behavior is a result of the agents' unawareness about when and where the tsunami hits the land. An integration of time- and location-dependent

flooding information could help to avoid such behavior. An implementation approach is discussed in the following chapter.

Chapter 5

Time-dependent networks

In the case of an evacuation simulation the network often has time-dependent attributes. For instance, large-scale inundations or conflagrations do not cover the entire hazard zone at once. Time-dependent aspects must not be neglected in evacuation planning otherwise inappropriate recommendations may be the result. This issue has been demonstrated based on the Padang scenario in Chapter 4, where many agents flee in parallel to the shore instead of moving away from the approaching tsunami. This chapter discusses how the spreading of inundations or conflagrations can be modeled by time-dependent networks where parameters like walking speed or flow capacity can vary over time. The remainder of this chapter is organized as follows. Implementation details are given in Chapter 5.1. A description of the test scenario is given in 5.2. Simulation results are discussed in Chapter 5.3. A discussion on the results can be found in Chapter 5.4 and a brief conclusion in Chapter 5.5. The material in this chapter was partially published in (Lämmel et al., 2010a)

5.1 Implementation

The disaster related time-varying aspects are modeled as network change events. Network change events modify link parameters while the mobility simulation is running. In the evacuation context, a network change events sets the free flow speed of a link to zero as soon as the link is no longer passable. The network change events are stored independent from the network

Listing 3 Sample network change event: valid from 03:06 AM, applied to links with id 12487, 12489, and 12491. The change event sets the free speed of the corresponding links to zero.

```
<networkChangeEvent startTime="03:06:00">
            <link refId="12487"/>
            <link refId="12489"/>
            <link refId="12491"/>
            <freespeed type="absolute" value="0.0"/>
 </networkChangeEvent>
```

in a separate XML file. Listing 3 shows a sample network change event in its XML notation. The event manipulates links 12487, 12489, and 12491 at 03:06 AM by setting the free speed to 0 m/s. The change values are in SI based units.

The network change events file is loaded at simulation start up during the run of the mobility simulation (*mobsim* module). The network change events will be applied to the corresponding links at the given times. The time-dependent attributes of the links are accessed by time-dependent getter methods. If there exists a network change event for the given query time, then the value of this network change event will be returned. But, as discussed in Chapter 3.2, it is not expected that a network change event exists for every link and every time step. If for a given query time no network change event exists, then the value of the network change event with the next smaller event time will be returned. If there exists no network change event at all, then the link's default value will be returned. The time-dependent getter methods give the flexibility to query the attributes in arbitrary chronological order. A discussion about the implementation of the retrieval mechanism is given in Appendix C.

5.2 Inundation scenario

The inundation data is based on simulations and has been provided by "Last-Mile" project partners. Details on the simulation approach and the resulting inundation scenario are given in (Goseberg et al., 2009; Goseberg and Schlurmann, 2009). The data has been delivered as a set of 8 netcdf[1] files and comprises highly resolved time-dependent flooding information. The spatial resolution is $3\,m$ on average and the temporal resolution is $60\,s$. From this data 108 network change events affecting 7690 links have been created. A map with time-dependent flooding information is shown in Figure 5.1. The flooding covers an area of about $25\,km^2$. Besides the time-dependent flooding information this map also shows a spatial buffer of $500\,m$ around the maximum flooding expansion. This buffer is of particular importance because people outside the directly affected area should keep some distance to the danger. Otherwise, those people could block evacuees from leaving the hazard zone. Therefore, an additional buffer around the hazard zone is needed. This buffer has also to be evacuated. A side effect of this inundations evacuation area approach is that a considerable smaller area has to be evacuated compared to the basic "$10\,m$" approach, introduced in Chapter 4. As a consequence of the smaller

[1]For more information about netcdf see, e.g., http://en.wikipedia.org/wiki/NetCDF (accessed July 2010)

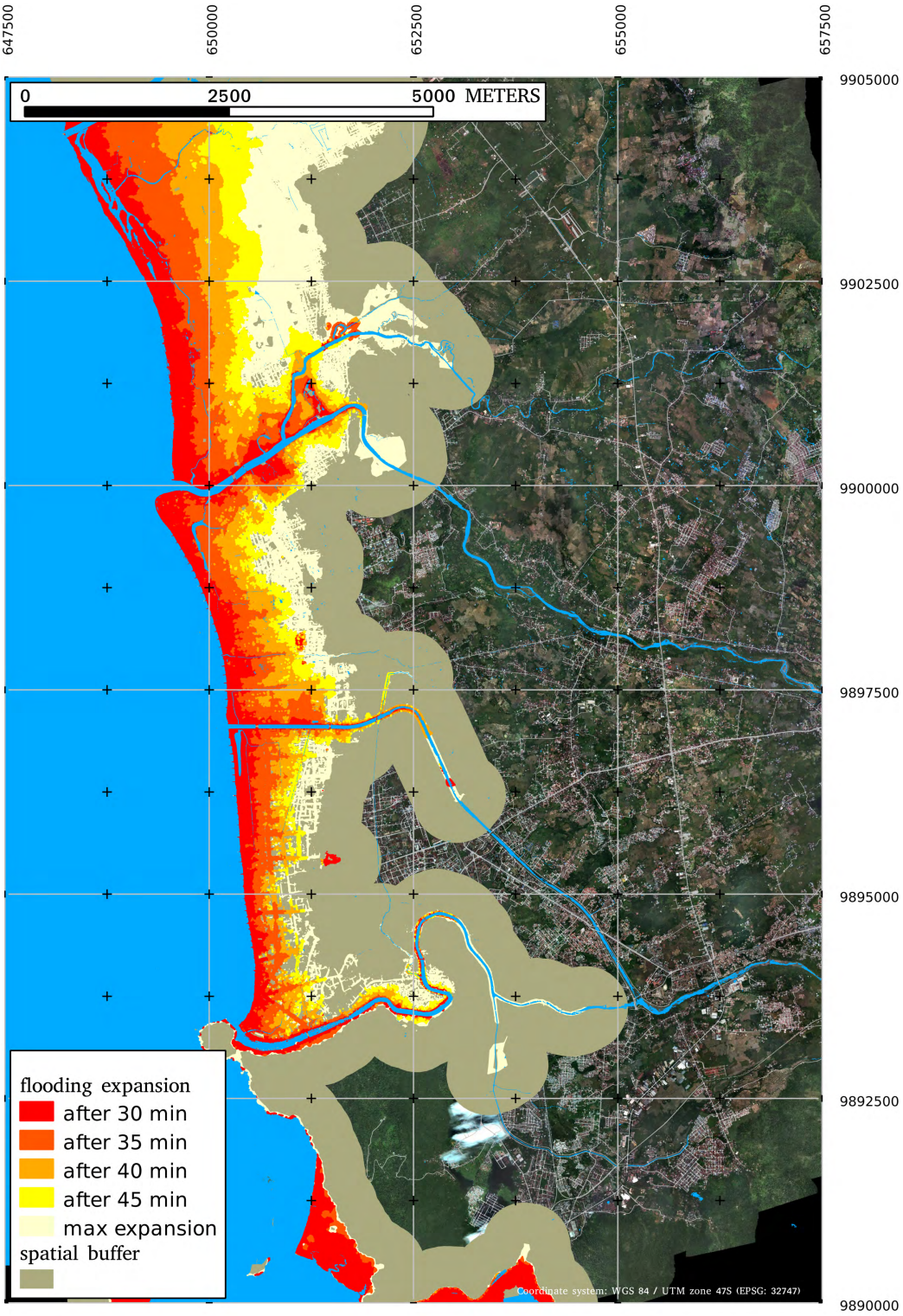


Figure 5.1: Map of Padang with time-dependent flooding information and spatial buffer for evacuation.

time of day	number of persons
night	218 038
morning	278 411
afternoon	245 388

Table 5.1: Number of persons inside the evacuation area for the night, morning and afternoon scenario.

area the number of affected people is also smaller. The actual numbers for the morning, afternoon, and night scenario are given in table 5.1. This chapter, again, only considers the morning scenario in experiments section.

5.3 Experiments

The evacuation setup for the flooding scenario has been investigated through three different runs. *Run 5.1* implements the shortest path solution (SP solution). Strictly speaking, the shortest path solution should better be called free flow solution. This is because the SP solution in this chapter takes the time-dependent flooding of the links into consideration but assumes that there is never congestion on any link and so free speed is always possible. However, to keep consistency with the strategy names it is referred here to as shortest path solution (SP solution). The other two runs are implement the Nash equilibrium approach (NE approach) (*Run 5.2*)and the marginal social cost based approach (MSCB approach) (*Run 5.3*).

The evacuation area encloses the assumed flooding area extended by a spatial buffer. The synthetic population consist of 278 411 agents. This number corresponds to the morning population for the evacuation area. *Run 5.1* corresponds to the first iteration of *Run 5.2* and *Run 5.3*. This means, no extra run had to be performed in order to get the results for the shortest path solution. However, for consistency reasons the shortest path solution is called *Run 5.1*. *Run 5.2* and *Run 5.3* have been performed with the following setup:

- For the first 750 iterations 10% of the population produce new routes using the `ReRoute` module during *re-planning*. For the remainder (90% of the population) an existing plan is chosen out of their memory by the `ChangeExpBeta` module.
- From iteration 751 `ChangeExpBeta` is performed for all agents, meaning no new routes are produced from iteration 751 on.
- The simulation stops after iteration 1000 finishes.

parameter	reading *Run 5.2*	reading *Run 5.3*
wall time	21:45:33	1:12:16:25
CPU time	1:01:49:42	1:17:31:26
max memory	8.296 GiB	8.260 GiB

Table 5.2: Performance measurement of *Run 5.2* and *Run 5.3*.

The simulation runs have been performed on *cluster 7* hosted at the math institute of TU Berlin[2]. Each node of the cluster is equipped with 2 QuadCore-Xeon X5550 (2.67 GHz) CPU and 24 GiB or 48 GiB RAM. To get an idea of the complexity of the underlying large-scale scenario measurements of the computing time and memory consumption are important. Table 5.2 give basic performance measurements for *Run 5.2* and *Run 5.3*. The wall time describes the "real" time the simulations took, i.e. the result depends on other processes running at the node. For that reason, the wall time gives only an estimate how long one has to wait for a simulation run to complete. The run time and memory consumption for both runs is reasonable. However, with the social cost optimization enabled (*Run 5.3*) the CPU time increases. A comparison with the run times of the basic setup in Chapter 4.3 shows that the run times have nearly halved. This is not a positive side effect of the network change events, but it is due to the considerably smaller evacuation area compared to the "10 m" approach.

As before, the simulation is stopped after the found solution does not longer (significantly) improve. The learning curves for *Run 5.2* and *Run 5.3* in Figure 5.2 indicate that 1 000 iterations of learning are sufficient. The figure shows that, as expected, the average evacuation time for the MSCB approach (*Run 5.3*) is less than for the NE approach (*Run 5.2*). The difference in the average evacuation time between NE approach and MSCB approach is 55 seconds.

An important indicator for the quality of an evacuation solution is the evacuation curve. The evacuation curve describes the number of already evacuated agents over time. Figure 5.3 shows the evacuation curves for *Run 5.1* (SP solution), *Run 5.2* (NE approach), and *Run 5.3* (MSCB approach). In Chapter 4 it has been discussed that the SP does not take congestion into consideration and therefore SP underestimates the resulting travel times (cf. Figure 4.11). When it comes to the integration of the tsunami waves, then SP performs even worse. There are a lot of agents that do not manage to escape

[2]For more information about the cluster look at http://www.naturwissenschaften.tu-berlin.de/iuk/computeserver/hardware/hardwareuebersicht (in German, accessed July 2010)

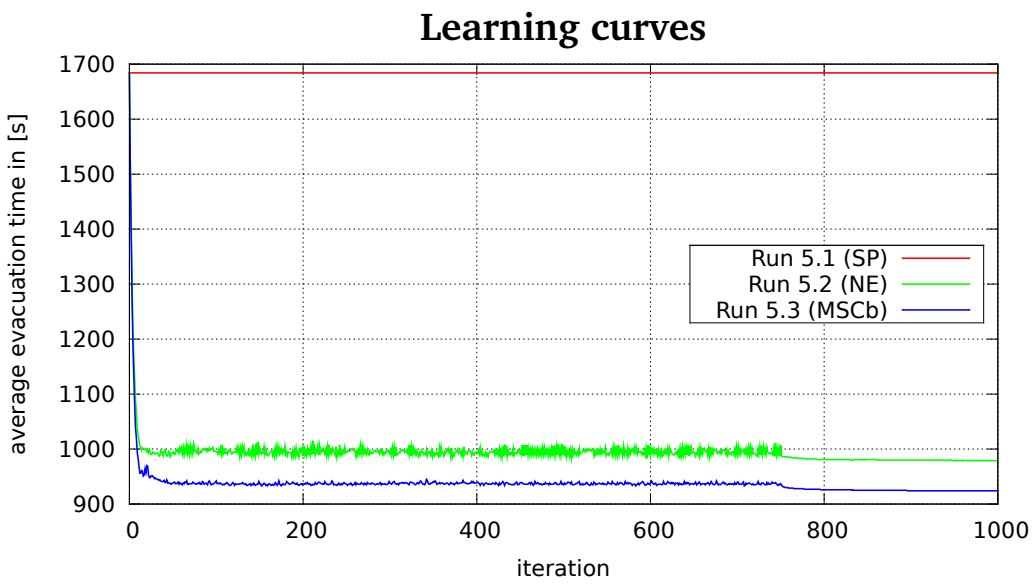


Figure 5.2: Average evacuation time over iteration number for *Run 5.1* (SP), *Run 5.2* (NE), and *Run 5.3* (MSCB approach). Note, no learning iterations have been performed for *Run 5.1*. Therefore, its average evacuation time remains constant over the iterations. For the SP solution the average evacuation time is 1684 seconds, for the NE approach it ends up with 979 seconds, and for the MSCB approach it ends up with 924 seconds.

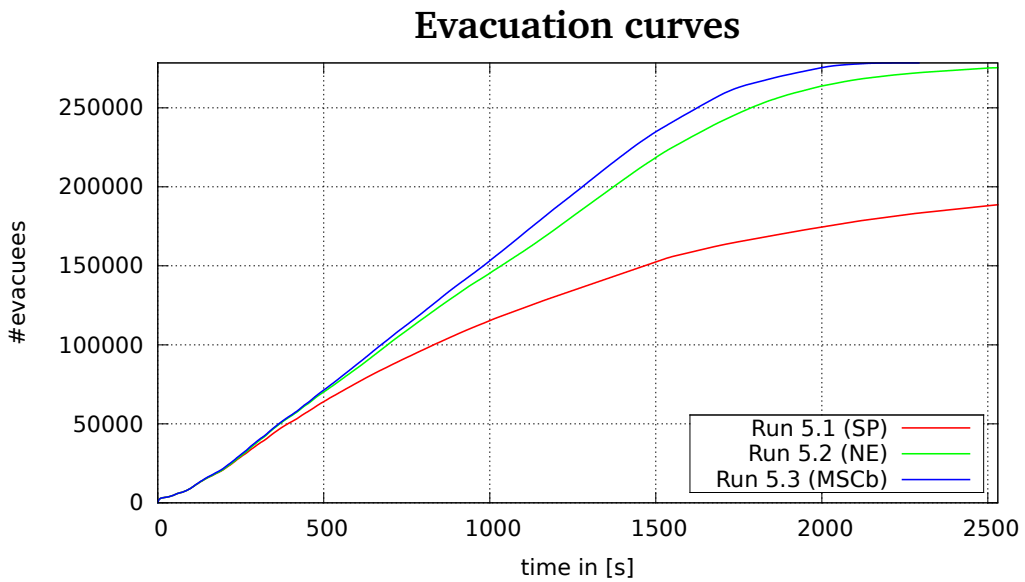


Figure 5.3: Evacuation curves of *Run 5.1* (SP solution), *Run 5.2* (NE approach), and *Run 5.3* (MSCB approach). For many agents the SP solution run does not leave enough time. On this account, the corresponding evacuation curve stagnates at about 200 000 agents. The curves of NE approach and MSCB approach are more similar to each other, where MSCB approach performs slightly better (steeper gradient). The total egress time in the MSCB approach case is 2 292 seconds and for the NE approach 2 530 seconds.

at all. This fact is indicated by the evacuation curve for the SP solution (*Run 5.1*) in Figure 5.3 and by the visualizer snapshots presented in Figure 5.4.

But also in the NE approach (*Run 5.2*) 3 038 of agents did not manage to escape (red agents in Figure 5.4), where for the MSCB approach only one out of the 278 411 agents did not manage to escape. Visualizer snapshots of *Run 5.1*, *Run 5.2*, and *Run 5.3* are given in Figure 5.4. There are alternative evacuation routes for at least some agents in the southern part of the city. It seems to be that if the red agents in *Run 5.2* would evacuate to the northeast, instead of evacuating to the south, they would be safe. The question raises why does the iterative learning procedure do not find those alternative evacuation routes, and what are the solutions to this problem. This will be discussed in the following section.

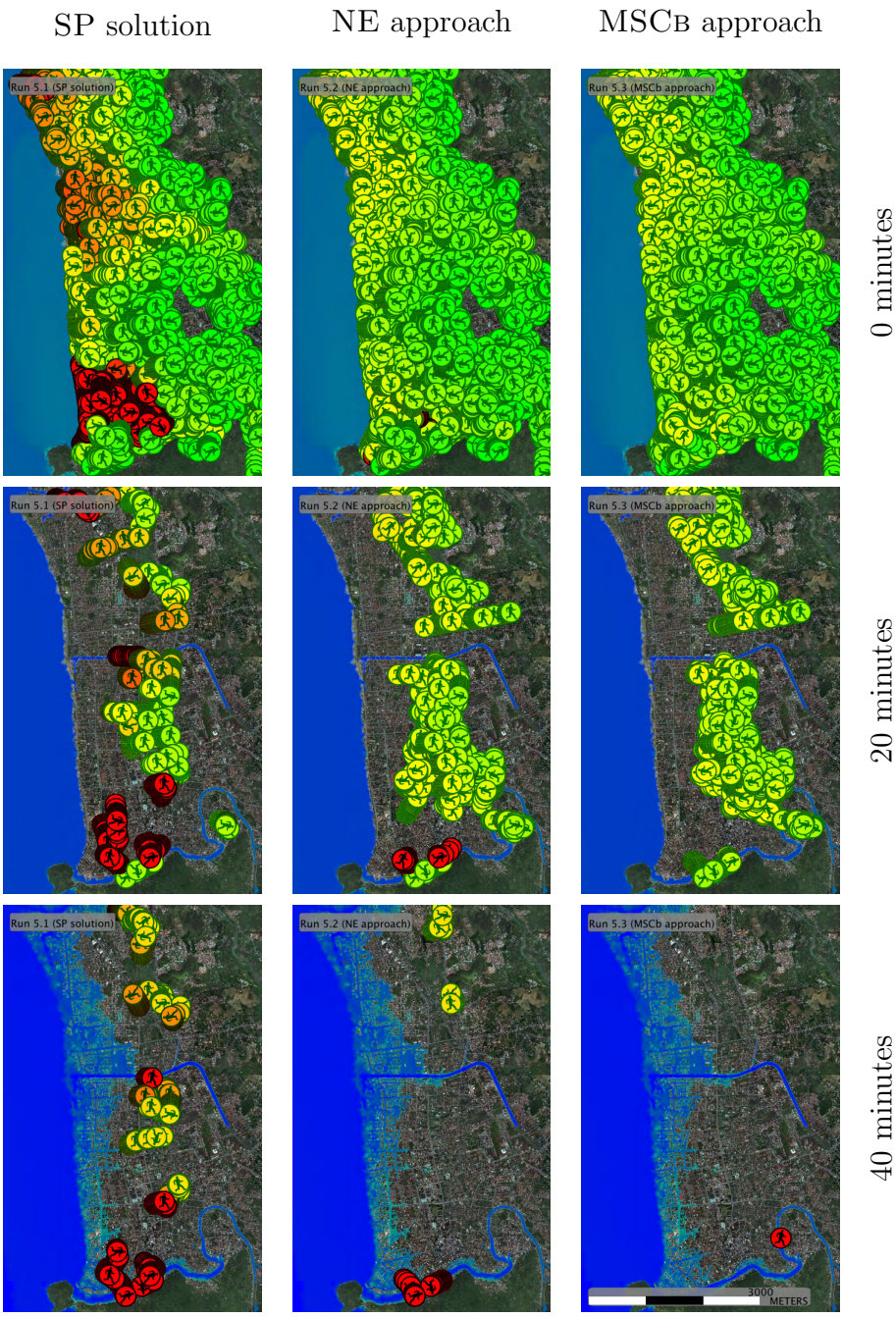


Figure 5.4: Visualizer snapshots of *Run 5.1*, *Run 5.2*, *Run 5.3* at the beginning of the evacuation (top), after 20 minutes of evacuation (center), and after 40 minutes of evacuation (bottom). The agents are colored depending on the time they need for the evacuation. A green color indicates that they escape quickly and as the color moves from green to yellow to red the evacuation time increases. A red color stands for agents that do not manage to escape at all. There are many red agents in the SP solution and NE approach case and one single red agent in the MSCb approach case.

5.4 Discussion

The integration of time-dependent flooding data considerable improves the evacuation results compared to the basic setup discussed in Chapter 4. However, there are still agents that do not manage to escape the flooding even in the optimized NE approach and this despite that at least for some agents a better solution exists. A reason why those agents are not escaping from the flooding is because they "think" they will make it just in time. This miscalculation is a consequence of the stochastic characteristics of the model. A simple approach to avoid this "miscalculation" would be to let the tsunami arrive earlier, say 5 minutes, while learning and after that for one additional iteration with the actual time. In general, the arrival time of the tsunami, or to be more precise the advance warning time, can be seen as an uncertain aspect in the evacuation planning. Any evacuation strategy exposes evacuees to risk of being caught by the tsunami conditional to the advance warning time. Furthermore, the advance warning time follows a probability distribution that is hard to estimate. This calls for an evacuation strategy that is risk reducing unconditional of the advance warning time. How such an evacuation strategy can be developed is discussed in Chapter 6.

5.5 Conclusion

The time and location dependent flooding has been modeled as a time-dependent network in a way that as soon as a link gets flooded its free speed will be set to zero. It is common to model time-dependent networks as time-expanded graphs. However, this leads to a lot of overhead regarding memory consumption. Therefore, the network change events have been introduced. A network change events changes the parameters of the links in the network (e.g. free speed). The advantage over time-expanded graphs is that network change events are only be triggered if an actual change to the network happens.

The change events are stored in an XML file. The efficiency of its implementation is shown in Appendix C through benchmarks.

The explicit modeling of the inundation helps to overcome the problem with evacuation routes that are running in parallel to the shoreline instead of away from the approaching tsunami. It is obvious that introducing inundation reduces the diversity of the valid evacuation routes. As a result the NE approach becomes more similar to the MSCB approach compared to the results in Chapter 4. This means, through the introduced restrictions the NE approach is closer to an optimal solution as before. Furthermore, it has become even more clear that the shortest path solution is not qualified for

evacuation planning, since many evacuees would not manage to escape from the tsunami. However, even when the NE approach is applied some agents do not manage to escape. This is because of the stochastic nature of the simulation that leads to an over or under estimation of the evacuation times. It has been discussed that the stochastic nature of the simulation can be seen as an uncertainty in the advance warning time. The higher this uncertainty is, the higher is the risk that not all evacuees manage to escape. This calls for a risk reducing evacuation strategy, which is discussed in the following chapter.

Chapter 6

Risk reducing evacuation strategies

The overall egress time is an important aspect in most evacuation situations. There are many models that find feasible routing strategies for evacuation scenarios. Sometimes the hazard zone is time-dependent. For instance, large-scale inundations or conflagrations do not cover the entire hazard zone at once. When modeling evacuation processes, it is important also to model those time-dependent aspects. In the time-dependent case the objective for the evacuation is to move all the evacuees to safety as fast as possible but never via an already inaccessible area. In the last chapter it has been discussed how the time-dependent aspects can be modeled using time-dependent network. In the time-dependent network a link is only accessible as long as the inundation or conflagration has not reached it. However, that approach works only if the advance warning time is known beforehand. When it comes to uncertain advance warning times that approach does not necessarily result in a routing strategy that is risk reducing. The issue has been demonstrated through the consideration of the expected value of an evacuation strategy's utility in Chapter 3.3. This chapter discusses the application of a risk reducing strategy for a flooding scenario within the MATSim framework.

Implementation details are given in Chapter 6.1. Experimental results are discussed in Chapter 6.2, followed by a discussion in Chapter 6.3, and a conclusion in Chapter 6.4. The material in this chapter is partially published in (Lämmel et al., in press).

6.1 Implementation

This section introduces a strategy that allows risk-decreasing moves only as long as such moves exist. A move is defined as risk-decreasing if after that move the evacuee's distance to the danger is higher as before that move. As discussed in Chapter 3.3, the aim of the risk reducing strategy is to increase the distance to the danger with every move. Inside the flooding area the distance describes the temporal distance.

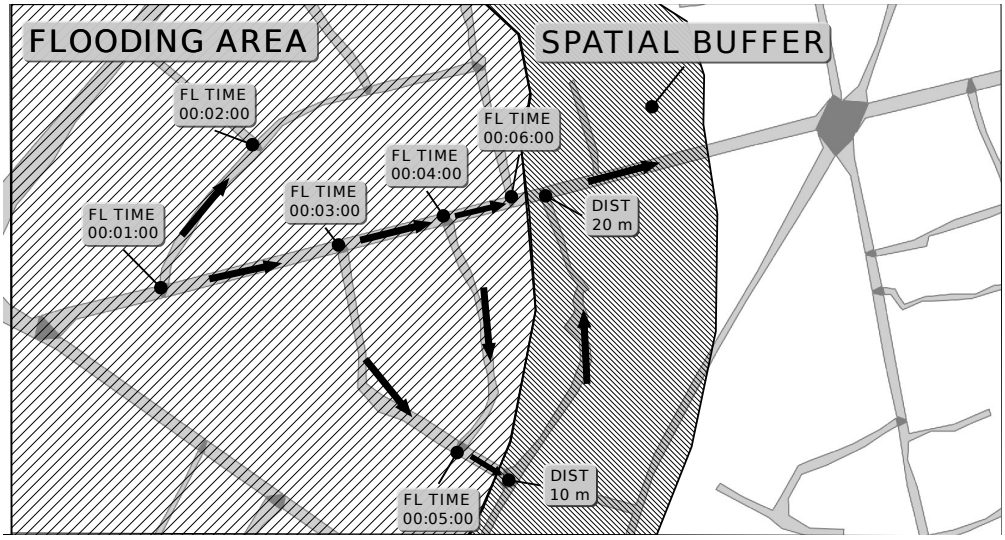


Figure 6.1: Illustration of the risk reducing strategy: The boxes denoted with "FL TIME" showing the flooding time for the corresponding crossings (nodes) and the boxes denoted with "DIST" showing the distance of the corresponding crossings to the flooding area. The black arrows pointing towards lower risk.

In inundation scenarios the temporal distance corresponds to the time span between the moment of the earthquake and the time the water reaches a given location. Thus, a move is seen as to be risk decreasing if the evacuee's location before the move has a smaller temporal distance to the water than the evacuee's location after the move. Furthermore, even people outside the directly affected area should keep some distance to the danger. This is important because those people could block evacuees from leaving the flooding area otherwise. Therefore, an additional buffer around the flooding area is needed. This buffer has also to be evacuated. Within this buffer a move is defined as risk-decreasing if it increases the evacuee's spatial distance to the flooding area. An illustration of this risk reducing strategy is given in Figure 6.1.

In general, there may be a number of different evacuation paths. Some of them may only have risk decreasing moves, some of them not. Let P denote the set of all possible paths in the network, $P^{risky} \subseteq P$ denotes the set of all paths with at least one risk increasing move and $P^{safe} = P \setminus P^{risky}$ the set of all risk decreasing paths, respectively. In the underlying network based traffic simulation, the decision points are at nodes. As soon as an agent has entered a particular link she has to travel along that link until the next node. On this account, the risk levels are calculated for nodes. The risk level of a node $n \in \mathcal{N}$ is denoted by r_n. The time when the node gets impassable is denoted by t_n^{imp}; for nodes outside the flooding area $t_n^{imp} = \infty$. The distance of a node

to the flooding area is denoted by $d_n^{flooding}$; for nodes inside the flooding area $d_n^{flooding} = 0$. For simplicity it is assumed that the spatio-temporal expansion of the flooding is monotone increasing function of location and time. In other word as soon as a node has been flooded, it stays flooded to the end of the evacuation. If $t_{max}^{flooding}$ is the time when the flooding has reached its maximum expansion, the risk level of a node is defined as follows:

$$r_n = \begin{cases} t_{max}^{flooding} - t_n^{imp}, & \text{if } t_n^{imp} \neq \infty \\ -d_n^{flooding}, & \text{otherwise.} \end{cases} \tag{6.1}$$

In order to compare the risk level of nodes that are inside the flooding area with those inside the spatial buffer the risk level of a node is a number without an unit. Since the risk level of nodes inside the flooding area are positive numbers and negative numbers for nodes in the spatial buffer, it is guaranteed that nodes inside the spatial buffer always have a lower risk level than nodes inside the flooding area. If a link leads from a lower risk level node to a higher risk level node then that link will be charged by adding an additional penalty to the cost function of the respective routing algorithms. The penalty will be chosen proportional to the length of the link. Since the routing strategy has to avoid risky paths as long as a non-risky path exist the minimal penalty has to be at least as high as the cost of the worst non-risky path. This is achieved by adding additional risk penalty costs C^r to the cost function C. For a link $a = (i, j)$ with length l_a the risk penalty costs C_a^r are defined as follows:

$$C_a^r = \begin{cases} l_a * penalty, & \text{if } r_i < r_j \\ 0, & \text{otherwise,} \end{cases} \tag{6.2}$$

where *penalty* is a constant that has to be chosen that the following inequality holds:

$$\min_{p \varepsilon P^{risky}} C(p) > \max_{p' \varepsilon P^{safe}} C(p') \tag{6.3}$$

The risk costs are static cost offsets that are applied to the links in the network. As static cost offsets, the risk costs are straight forward applicable to the cost functions of Algorithms 1, 2 and 3. Recall: $C_a(k)$ denotes the link costs for link a if entered during time bin k, $\tau_a^{free}(k)$ denotes the free speed travel time for link a if entered during time bin k, $\tau_a(k)$ the travel time for link a if entered during time bin k, and $C_a^s(k)$ denotes the external costs that an additional agent imposes to the system if entered link a during time bin k (for a detailed explanation see, Chapter 3.1). The modified costs function for the SP solution is:

$$C_a(k) = \tau_a^{free}(k) + C_a^r. \tag{6.4}$$

Note, originally the cost function for the SP solution is not time-dependent. However, as discussed in Chapter 5.3, the SP solution takes the time-dependent link flooding into consideration but assumes that there is never congestion on any link and so free speed is always possible. For that reason, the time information k is needed. For the NE approach the costs function is:

$$C_a(k) = \tau_a(k) + C_a^r \tag{6.5}$$

And for the MSCB approach it is:

$$C_a(k) = \tau_a(k) + C_a^s(k) + C_a^r \tag{6.6}$$

The results of the experiments are given in the following section.

6.2 Experiments

Three runs have been performed to investigate the performance of the risk reducing strategy. *Run 6.1* implements the risk reducing shortest path solution (SP solution), *Run 6.2* the risk reducing Nash equilibrium approach (NE approach), and *Run 6.3* the risk reducing marginal social cost based approach (MSCB approach). For all three runs the links' free flow travel times are time-dependent as discussed in Chapter 5. The evacuation area encloses the area that will be flooded according to flooding simulations (Goseberg et al., 2009; Goseberg and Schlurmann, 2009) extended by an additional spatial buffer of $500\,m$. The scenario is the same as in previous chapters. The synthetic population consist of 307 413 agents. Details of the underlying scenario are given in Chapter 4 and Chapter 5.

In order to achieve the risk reducing behavior the cost function of Algorithm 1 (SP solution) has been replaced with the cost function given in Equation (6.4), for Algorithm 2 (NE approach) the cost function has been replaced with the cost function given in Equation (6.5), and for Algorithm 3 (MSCB approach) with the costs function given in Equation (6.4) accordingly. The constant *penalty* in Equation (6.2) has been set based on a heuristic estimate to 30 hours per 100 meters[1].

For the first iteration $\tau_a(k)$ corresponds to the free flow travel time and $C_a^s(k) = 0$ for all $a \in A$ and $k = 0...K-1$. Therefore *Run 6.1* corresponds to the first iteration of *Run 6.2* and *Run 6.3*. Every agent follows the shortest path according to the cost function given in Equation (6.4). *Run 6.2* and *Run 6.3* have been performed with the following setup:

[1]In the underlying scenario there is no (feasible) evacuation path that takes so long. Therefore, Inequality (6.3) holds.

parameter	reading *Run 6.2*	reading *Run 6.3*
wall time	23:42:47	2:07:18:16
CPU time	22:52:19	2:08:58:05
max memory	8.354 GiB	8.279 GiB

Table 6.1: Performance measurement of *Run 6.2* and *Run 6.3*.

- For the first 750 iterations 10% of the population produce new routes using the `ReRoute` module during *re-planning*. For the remainder (90% of the population) an existing plan is chosen out of their memory by the `ChangeExpBeta` module.
- From iteration 751 `ChangeExpBeta` is performed for all agents, meaning no new routes will be produced from iteration 751 on.
- The simulation stops after iteration 1000 finishes.

Information about run time and memory consumption of *Run 6.2* and *Run 6.3* is shown in Table 6.1. A visual comparison of the three runs is given in Figure 6.2. The figure shows visualizer snapshots for the first 40 *min* of the evacuation. The figure compares the results of *Run 6.1*, *Run 6.2*, and *Run 6.3*. The agents are colorized depending on the time they need for the evacuation. A green color indicates that they escape quickly while a red color means they need rather long time or even fail to evacuate. In *Run 6.1* (SP solution) there are many red agents in the northern and southern part of the city, indicating that they do not have enough time. The colorization of the agents in *Run 6.2* (NE approach) and *Run 6.3* (MSCB approach) do look very similar. There are also some agents who need a very long time to evacuate. In general the evacuation times are higher compared to the run without risk costs in Chapter 5.3 (c.f. Figure 5.4). These longer evacuation times are not unexpected since the agents have to deal with another constraint when finding evacuation routes (i.e. risk avoidance). The longer evacuation times are also shown in the evacuation curves in Figure 6.3. The total evacuation time for the NE approach is about 9 900 seconds and for MSCB approach about 8 400 seconds. However, the fact that evacuees are reaching the safe area long after the tsunami inundated the city shows that the congestion leading to this long evacuation time must be outside the inundation area. This is because the inundation has reached its maximum expansion after about one hour and if the congestion would have occurred within the flooding area none of the evacuees who got stuck there would be able to reach the safe area. At the end all agent manage to escape for both the NE approach and the MSCB

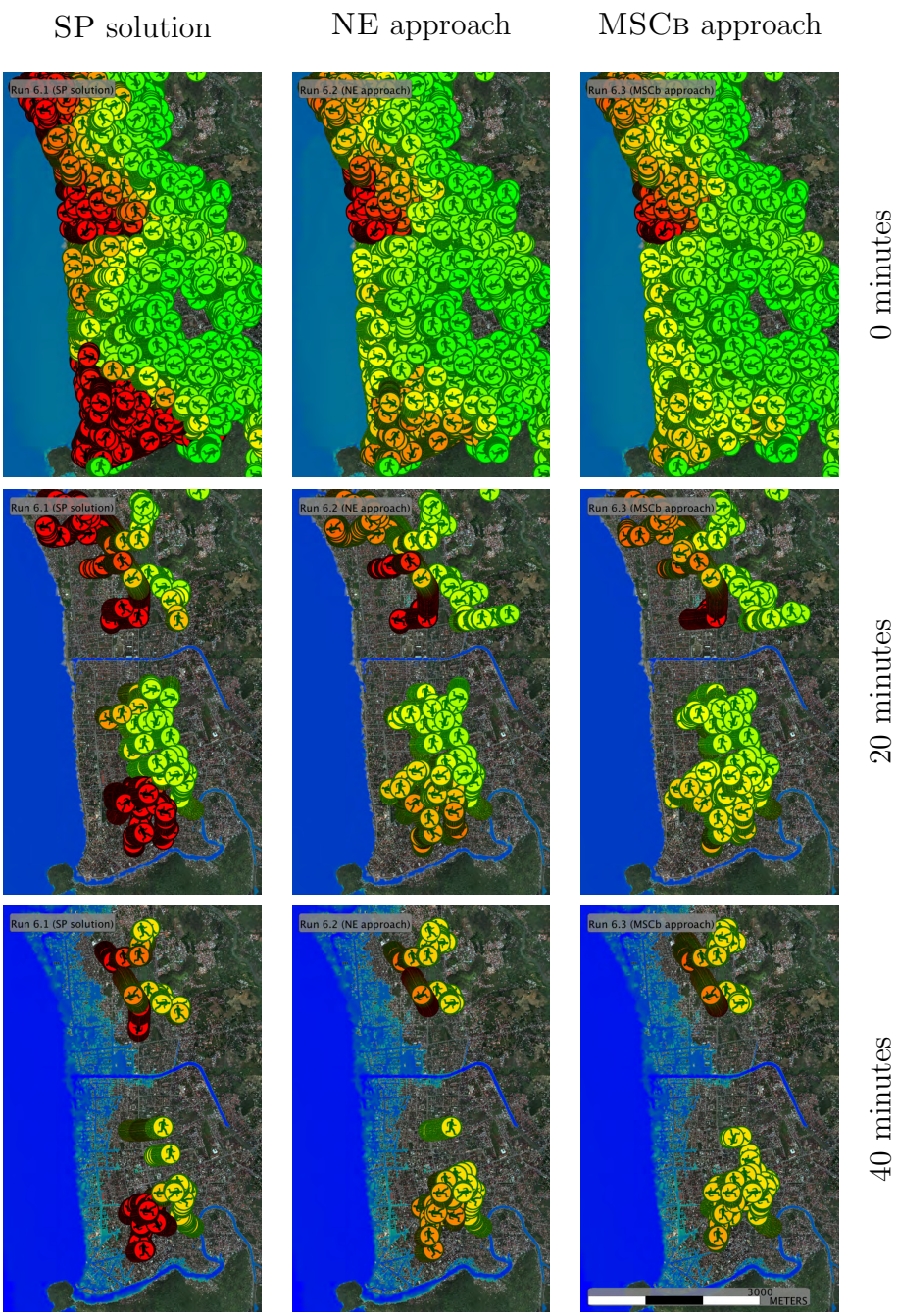


Figure 6.2: Visualizer snapshots of *Run 6.1*, *Run 6.2*, *Run 6.3* at the beginning of the evacuation (top), after 20 minutes of evacuation (center), and after 40 minutes of evacuation (bottom). The agents are colored depending on the time they need for the evacuation. A green color indicates that they escape quickly while a red color means they need rather long time for the evacuation.

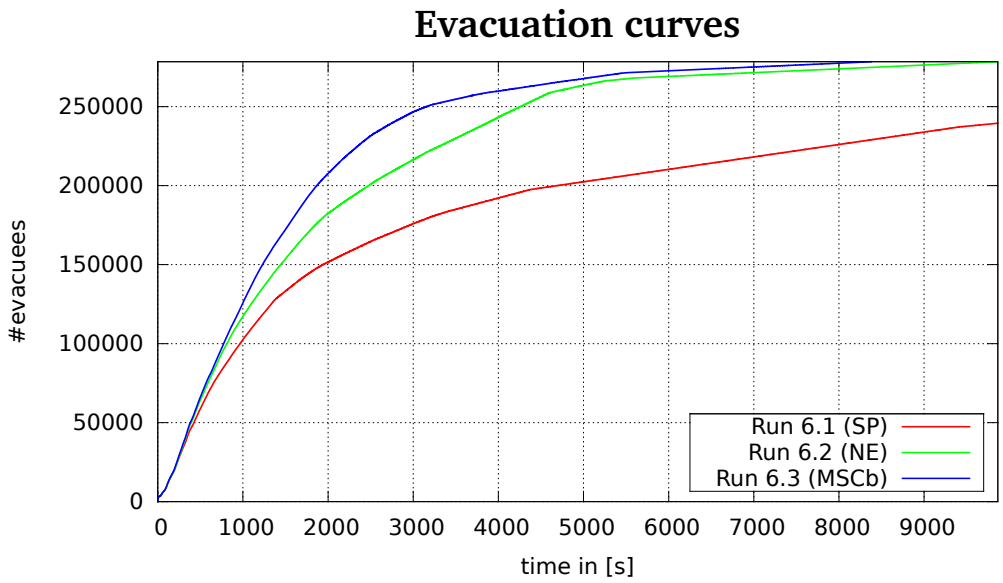


Figure 6.3: Evacuation curves of *Run 6.1* (SP solution), *Run 6.2* (NE approach), and *Run 6.3* (MSCB approach). The risk reducing evacuation strategy leads to longer evacuation times compared to an evacuation without additional risk penalty (c.f. Figure 5.3).

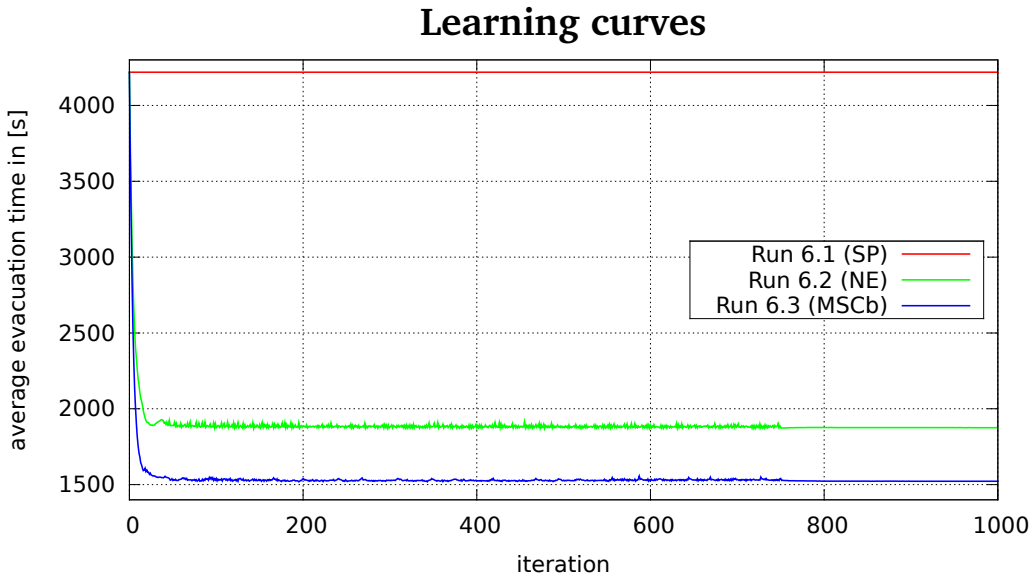


Figure 6.4: Average evacuation time over iteration number for *Run 6.1* (SP solution), *Run 6.2* (NE approach), and *Run 6.3* (MSCB approach). Note, that no learning iterations have been performed for *Run 6.1*, therefore, its average evacuation time remains constant over the iterations. For the SP solution the average evacuation time is 4 219 seconds, for the NE approach it goes down to 1 875 seconds and for the MSCB approach ends up with an average evacuation time of 1 522 seconds.

approach. The times are also worse compared to an evacuation simulation without risk costs (cf. Figure 5.2). The average evacuation times are shown as learning curves in Figure 6.4.

One learns from these results that the introduced approach leaves enough time to evacuate all agents for both the NE approach and the MSCB approach. However, in certain areas, in particular in the northern part of the city, many agents need rather long time to evacuate. Those areas have to be seen as to be highly endangered and consequentially it is strongly recommended to find solutions for a faster evacuation.

As a first step to improve the situation a detailed spatial analysis of the evacuation time is needed. Such an analysis helps to identify areas where the evacuation takes too long. Results of a grid based GIS analysis for the average evacuation times are given in Figure 6.5. The figure shows the GIS analysis for *Run 6.2* (NE approach) and *Run 6.3* (MSCB approach). The GIS analysis is performed on a 500 meter grid. For each cell in the grid the number of departing agents is recorded. The cell colors describe the average evacuation time over all agents that depart from within the corresponding cell. The numbers in the cells describe the total number of departing agents. There are no big difference regarding the average evacuation times between the NE approach and the MSCB approach. This is an indicator that the additional static risk costs push both approach closer to the same solution, which is not unexpected since the risk costs reduce the number of feasible evacuation paths. In general, it can be stated that there are a lot of agents that need long evacuation times especially in the costal area (red cells in Figure 6.5).

6.3 Discussion

The risk cost approach increases the evacuation time independent of the routing strategy considerably. However, this behavior is not unexpected since the risk cost approach "forbids" several routes that would lead to shorter evacuation times. A good example documenting this fact is the Siti Nurbaya Bridge in the southern part of the city. The visualizer screen shots in Figure. 6.6 are taken after 5, 20 and 35 minutes of evacuation. The screen shots on the top show the evacuation behavior for run *Run 5.2* (i.e. evacuation without risk costs). As there is no risk cost penalty for agents moving towards the danger many of them take the Siti Nurbaya Bridge to get to safety (green agents). However, there are a lot of agents that "think" they would make it over the bridge but get caught by the inundation (red agents). The screen shots at the bottom demonstrate the risk reducing strategy for *Run 6.2*. It is clearly shown

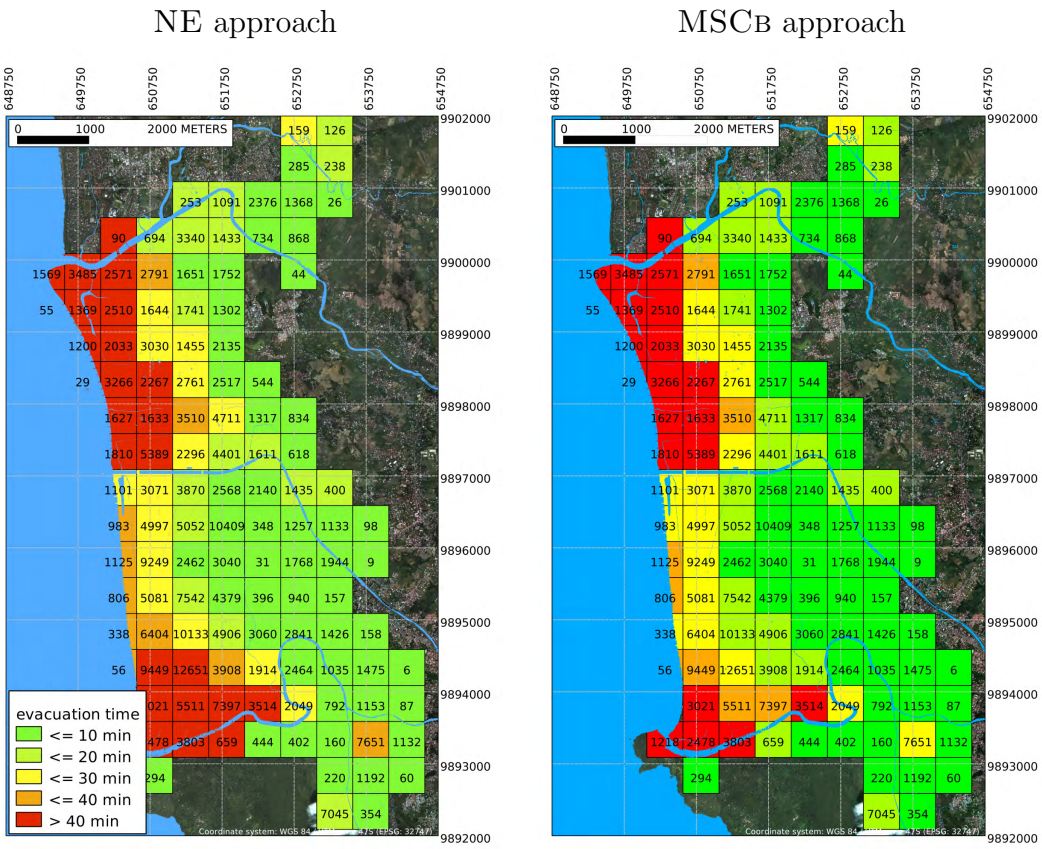


Figure 6.5: GIS analysis of the average evacuation times on a 500 meter grid. The figure on the left corresponds to *Run 6.2* (NE approach) and the figure on the right to *Run 6.3* (MSCB approach). The cell colors describe the average evacuation time over all agents per cell. Details can be found in the legend. The numbers in the cells describe the total number of departing agents per cell.

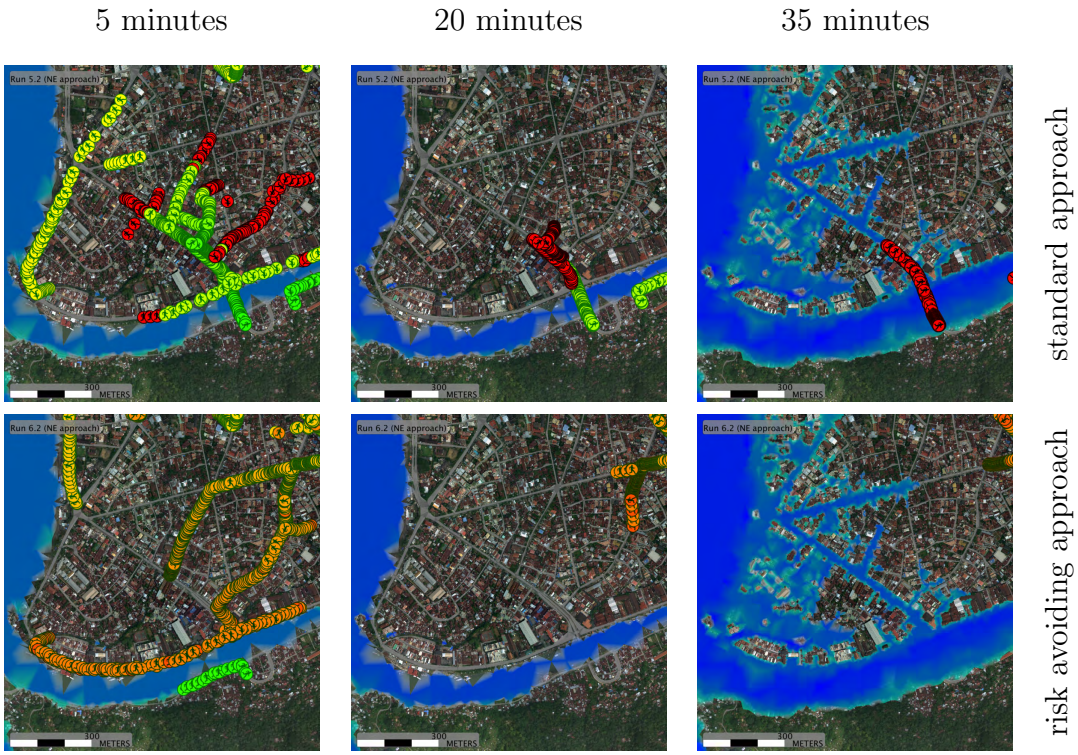


Figure 6.6: Screen shots for *Run 5.2* (top) and *Run6.2* (bottom) after 5, 20, and 35 minutes (from left to right) of evacuation. It is clearly shown that for *Run 5.2* the agents do not avoid the bridge and thus moving towards the danger first before getting to safety. In *Run 6.2* the agents avoiding this bridge because of the risk cost penalty.

that in this run agents avoid that bridge at the cost of a longer evacuation time (indicated by a more orange agents).

6.4 Conclusion

When it comes to uncertain aspects in evacuation situations one has no longer to be prepared for a single known scenario but for a range of different situations. In the case of a tsunami related evacuation one uncertain aspect is the advance warning time. There may be evacuation routes that are fast if the advance warning time is long enough but would not work otherwise. If the advance warning time is not exactly known beforehand it is risky to take such evacuation routes. Therefore, a risk reducing evacuation strategy is proposed. The risk reducing evacuation strategy allows risk reducing moves only as long as such moves exist. This leads to a risk reducing behavior, where the evacuees are moving away from the danger. The risk reducing behavior is reached

through a static cost offset on links that are leading towards the danger. Since this approach "forbids" a lot of fast evacuation routes the evacuation time increases and there is a considerable number of evacuees in distinct areas that need too long to evacuate even in the MSCB approach case. Those areas can be seen as highly endangered, where a vertical evacuation into shelter buildings seems to be appropriate. Shelters are sinks with limited capacities and, therefore, the common evacuation optimization approaches are not longer applicable. An extended version of the simulation framework that can deal with shelter buildings is discussed in the following chapter.

Chapter 7
Shelters

In the previous chapters several evacuation strategies have been discussed. If the advance warning time is long enough those evacuation strategies lead to plausible results, but in general there may be situations where also the best evacuation strategy does not leave enough time to evacuate all of the affected people. In the previous chapter the risk minimizing evacuation strategy has been introduced and it turned out that this strategy is plausible for scenarios with uncertain advance warning time, but in the underlying scenario this strategy leaves barely enough time for all evacuees. One possible solution to improve the situation would be the construction of shelters inside the hazard zone. For Padang one could build tsunami proof concrete towers or platforms that are high enough and strong enough. In addition existing buildings in the city could also serve as shelter. There are questions that arise when it comes to shelters.

- Where to build shelters?
- Which (storage) capacity do the shelters need to have?
- How to assign the people to the shelters?

The best solution for the evacuees would be to make every building in the city quake and tsunami proof. However, this would exceed the financial and technical capabilities for a city like Padang. The question is how to get a good benefit with existing means. This chapter discusses an evolutionary approach to find feasible solutions to the assignment problem. In order to simulate evacuation scenarios with shelters the simulation framework has to be extended. This is discussed based on the shortest path solution in Chapter 7.1. Chapter 7.2 gives implementation details for the NE approach and MSCb approach. The additional input data, which is needed to simulate shelters, is introduced in Chapter 7.3. Simulation results are discussed in Chapter 7.4. A discussion of the results is given in Chapter 7.5. A brief summery of the findings is given in Chapter 7.6. The material in this chapter is partially published in (Flötteröd and Lämmel, 2010).

7.1 Limited capacity sinks

The consideration of shelters changes the evacuation scenario. So far the multi-destination evacuation problem has been transformed into a single-destination evacuation problem. To be more precise the problem so far is a multi-source single-sink with unlimited capacity problem (see e.g., (Ford and Fulkerson, 1962; Lu et al., 2005)). As discussed in Chapter 3.4, this approach is not applicable to an evacuation scenario with shelters, since shelters are sinks with limited capacity. Therefore the evacuation network has been extended by a set $T^{shelter}$ of shelter nodes as shown in Figure 3.4. An evacuating agent can either chose the super sink t^{super} or a shelter $t^{shelter} \in T^{shelter}$ as her destination.

A shelter as a sink with limited capacity is a dynamic entity of the simulation like the time-dependent inundation expansion discussed in Chapter 5. The time-dependent inundation expansion has been modeled as network change events. Network change events are pre-calculated triggers for free speed changes on links. As soon as a link becomes impassable its free speed will be set to zero. Correspondingly, the entry to a shelter could be blocked as soon as it is filled up. However, the time when a shelter is filled up depends on the behavior of the agents and thus can not be pre-calculated. This means, the network change events based time-dependent network approach cannot be used to solve the shelter assignment problem. In this thesis an evolutionary learning algorithm is proposed which lets each agent find an appropriate shelter according to the underlying objective (NE approach vs. MSCB approach). The shortest path solution again serves as a benchmark to appraise the performance of the NE approach and the MSCB approach. However, the shortest path routing discussed in Chapter 3.1.1 needs to be adapted in order not to overload shelters. One way to deal with this problem is to assign the agents in a first come, first serve order (in terms of distances) to shelters and then calculate shortest path routes for each agent to her respective shelter. An algorithm that meets this requirement is easier to construct from the source nodes' perspective than from the agents' perspective. The proposed algorithm does not search shelter assignments and evacuation routes for agents but the algorithm assigns shelters and routes to the demand of source nodes $s \in S$. Algorithm 7 demonstrates the shortest path agent shelter assignment, where $c(t)$ denotes the storage capacity of sink node t with $c(t^{super}) = \infty$.

The algorithm works as follows: In step 1. the variables and storage structures are initialized. Step 2. essentially calculates a shortest path tree for each source node and stores all pairs of source node/ shelter node in ascending order based on the distances. In step 3. the algorithm iterates over the ordered source node/ shelter node pairs. For each pair the demand of the source node

Algorithm 7 First come, first served agents shelter assignment algorithm

1. initialize priority queue $OPEN = \{\}$, list $PLANS = \{\}$, $o_t = 0$ for all $t\varepsilon(T^{shelter} \cup t^{super})$and d_s with the number if departing agents for all $s \in S$
2. **for** all nodes $t \in (T^{shelter} \cup t^{super})$ **do**

   for all nodes $s \in S$ **do**

   (a) calculate shortest path length l from s to t

   (b) add tuple (s, t) with weight l to priority queue $OPEN$
3. **while** $OPEN \neq \{\}$ **do**

   (a) remove first tuple (s, t) from $OPEN$

   (b) **if** $d_s == 0$ **then** continue

   (c) $d = d_s$

   (d) **for** $i = 1$ **to** d **do**

   - **if** $o_t < c(t)$ **then**

     i. $o_t + +$

     ii. create plan p with origin s and destination t

     iii. add p to $PLANS$

     iv. $d_s - -$
   - **else** break
4. **return** $PLANS$

is assigned to the shelter node as long as there is demand left and the shelter capacity is not used up. The algorithm find a shelter assignment for every agent since the super node t^{super} is treated as a shelter node with unlimited space capacity. At the end the agents are assigned to the shelters according to a first come, first served order (in terms of travel distances), and since no agent can find a closer shelter by unilateral deviation Algorithm 7 produces a Nash equilibrium with respect to travel distances. Since this algorithm relies on the shortest path solution it does not take congestion into account. In general the algorithm could also be used with the generalized time-dependent cost function as it has been discussed for the NE approach and the MSCB approach. The *re-planning* in a MATSim iteration would work as follows:

1. Select a fraction of all agents for shelter assignment re-planning.
2. Remove those agents from their previously assigned shelters.
3. Perform Algorithm 7 for the re-planning agents with the now free shelter space and the corresponding cost function for links (i.e. according to the NE approach or the MSCB approach).

The complexity of the Algorithm 7 is dominated by the number of source nodes times number of shelters, since in step 2 of the algorithm a shortest path is calculated from every source node to every shelter node, meaning in the worst case a shortest path tree (Dijkstra, 1959) is calculated for every source node. For this reason the re-planning algorithm would have a rather long computation time. Therefore the iterative learning Algorithm 6 has been proposed in Chapter 3.4, which follows the "standard" MATSim *re-planning* procedure. In the following this algorithm will be tested on the Padang scenario for the NE approach and the MSCB approach.

7.2 Implementation

In MATSim shelters are represented by additional links where agents can perform their *post-evacuation* activity. However, since agents perform activities on links (cf. Chapter 4.1.1.2) and entrances to shelter buildings are bottleneck with a specific flow capacity at least two links have to be inserted for each shelter. The first shelter link (connector link) represents the bottleneck that regulates the inflow and the second link represents the shelter area. The connector link connects shelters to their respective nearest nodes in the simulation network. In reality evacuees have to walk a certain distance from the street into the shelter building and then upstairs to the safe floor. However, this distance is neglected here and the length of the connector links is set to

$1.66\,m$ arbitrarily. As a result, it takes one simulation time step to traverse a connector link if there is no congestion (a simulation time step is $1\,s$ and the free flow speed is $1.66\,m/s$ (cf. Chapter 4.1)). It is expected that the inflow rate of a shelter is dominated by the flow capacity of the bottleneck and not by the distance to walk, consequently this simplification is justified.

The flow capacity $q_a^{shelter}$ for a shelter link $a^{shelter}$ with minimum width w_a is set to:

$$q_a^{shelter} = w_a * 0.85 \frac{1}{m * s}. \tag{7.1}$$

The value of $0.85 \frac{1}{m*s}$ corresponds to the flow capacity on staircases when moving upstairs as it is proposed by Weidmann (1993). Since the *re-planning* module considers the space capacity of the shelters, no further mechanism is needed to guarantee that no shelter will become overcrowded (cf. Algorithm 6). In particular there is no mechanism in the mobility simulation that controls the number of shelter entering agents.

7.3 Shelters in the Padang scenario

Every building that is stable enough to withstand an earthquake and the subsequent tsunami and has at least one floor that is high enough could shelter evacuees. The challenge is to identify those buildings. Taubenböck et al. (2009b) introduced an approach to classify the stability of buildings by remote sensing technology. A sample of 500 of those buildings has been further appraised in a field survey by civil engineers. Some 40 of these buildings have been surveyed by the author of this work in July 2009. The survey of these buildings comprised among other things the measurement of the doors' and staircases' widths since doors and staircases are very likely bottlenecks for evacuees trying to get to the higher floors inside. However, many of these buildings have been severely damaged or even collapsed in the September 30, 2009 earthquake. Since the classification of buildings' stability is out of the scope of this work, it was decided to stick with the surveyed buildings. Consequently, the shelter buildings introduced in this chapter have to be considered as hypothetical. An overview of the hypothetical shelter buildings is given in Figure 7.1. The figure shows 42 buildings with a total shelter space for 31 536 evacuees. Since the number of evacuees in this scenario is 278 411 most of them still have to evacuate to the safe hinterland.

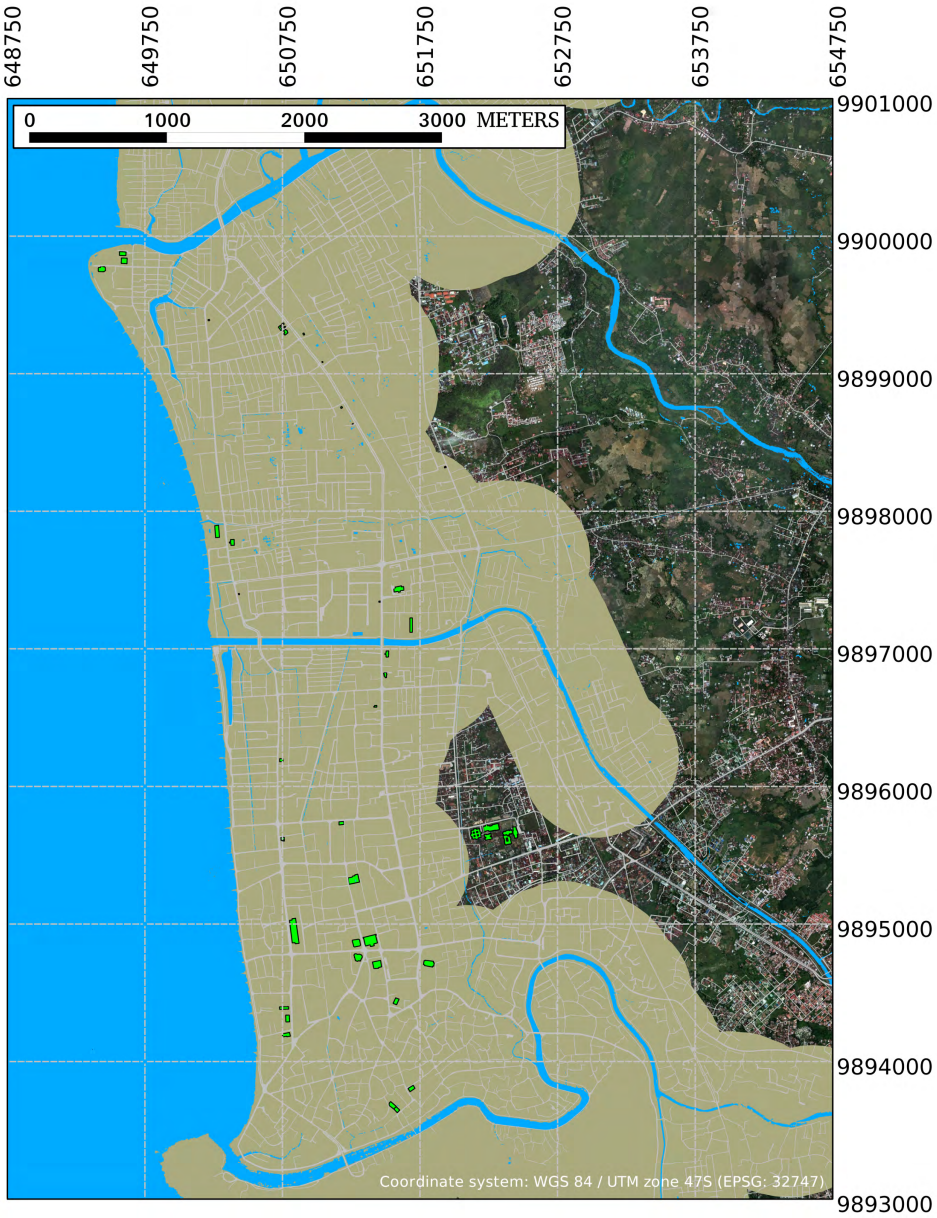


Figure 7.1: Map of the evacuation area in Padang with shelter buildings (green color).

7.4 Experiments

Three runs have been performed to investigate the performance of the shelter assignment algorithm. *Run 7.1* implements the shelter assignment based on the shortest path solution (SP solution), *Run 7.2* the shelter assignment based on the Nash equilibrium approach (NE approach), and *Run 7.3* the shelter assignment based on the marginal social cost based approach (MSCB approach). The basic assignment algorithms are discussed in Chapter 7.1 for the SP solution and in Chapter 3.4 for the NE approach and MSCB approach. For all three runs, the links' free flow travel times are time-dependent as discussed in Chapter 5. Furthermore, risk costs are also enabled as discussed in Chapter 6. The combination of shelters with risk costs lead to a setup where agents departing further inland will not occupy shelter space in shelters that are closer to the coastline. This can be interpreted as a policy to reserve shelters for the most vulnerable evacuees. The evacuation area encloses the area that will be flooded according to flooding simulations (Goseberg et al., 2009; Goseberg and Schlurmann, 2009) and an additional spatial buffer of $500\,m$. The scenario is the same as in previous chapters. The synthetic population consist of 278 411 agents. Details of the underlying scenario are given in Chapter 4, Chapter 5, and Chapter 6.

As mentioned, the SP solution (*Run 7.1*) has been calculated by Algorithm 7. The NE approach (*Run 7.2*) and the MSCB approach (*Run 7.3*) are the results of simulations according to Algorithm 6. The simulation is performed for 1 000 iterations with the following setup:

- The probability for an agent of being chosen for `ReRoute` during replanning is 5%.
- The probability of being chosen for the **shift** operation is 2.5%.
- The probability of being chosen for the **switch** operation is 2.5%.
- The probability to stick with the current plan 90%.

With this setup, 10% of the agents (try to) produce new plans in each iteration.

Information about run time and memory consumption is shown in Table 7.1. Note, that there are no measurements of the CPU time for *Run 7.2* and *Run 7.3*. However, since parallelization plays only a minor part, the CPU time should be, as for the runs in previous chapters, almost identical to the wall time[1]. The run time of *Run 7.1* is about $10\,min$, which is much shorter compared to both *Run 7.2* and *Run 7.3*. However, Algorithm 7 runs for only

[1]The *re-route* part of Algorithm 6 is executed in parallel (2 threads). However, the re-routing makes only a minor contribution to the overall run time.

parameter	reading *Run 7.1*	reading *Run 7.2*	reading *Run 7.3*
wall time	0:00:10:21	1:11:58:52	2:13:11:09
CPU time	0:00:10:25	—	—
max memory	8.205 GiB	8.264 GiB	8.258 GiB

Table 7.1: Performance measurements for the shelter runs.

one iteration, meaning the expected run time for 1000 iteration is some 7 days, which is still practical but considerably slower than Algorithm 6. A visual comparison of the three runs is given in Figure 7.2. The figure shows visualizer snapshots for the first 40 *min* of the evacuation. The figure compares the results of *Run 7.1*, *Run 7.2* and *Run 7.3*. The agents are colorized depending on the time they need for the evacuation. A green color indicates that they escape very quickly while a red color means they need rather long time or even fail to evacuate. In *Run 7.1* (SP solution) there are, despite of the availability of shelters, many red agents in the northern and southern part of the city, indicating that they do not have enough time. The colorization of the agents in *Run 7.2* (NE approach) and *Run 7.3* (MSCB approach) do look very similar. There are also some agents that need a very long time to evacuate and—in contrast to *Run 6.2* and *Run 6.3*—some agents on the northern tip of the evacuation area even fail to evacuate.

The learning performance of the shelter assignment algorithm is shown in Figure 7.3. The average evacuation time for *Run 7.1* (SP solution) is 3 142 seconds, which compares to 4 219 seconds in *Run 6.1* (SP solution without shelters). For *Run 7.2* (NE approach) the average evacuation time goes down to 1 535 seconds, which compares to 1 875 seconds in *Run 6.2* (NE approach without shelters, see Figure 6.4). And finally the resulting average evacuation time of *Run 7.3* (MSCB approach) is 1 317 seconds, which compares to 1 522 seconds in *Run 6.3* (MSCB approach without shelters). The average evacuation times are considerably less compared to an evacuation without shelters. This fact is also reflected in the evacuation curves depicted in Figure 7.4. A comparison between NE approach and MSCB approach runs with and without shelters based on detailed spatial analysis is shown in Figure 7.5.

It is shown that mainly agents that are departing near the coast gain from shelters. This is not only because many shelters are located near the coast (see Figure 7.1) but also because the risk costs prevent agents that are departing further inland to evacuate towards the coast (i.e. towards the approaching tsunami). Overall there are significant reductions in evacuation times. However, the setup in Chapter 6 (risk reducing evacuation strategy without shelters) leads to a situation where all agents manage to escape the

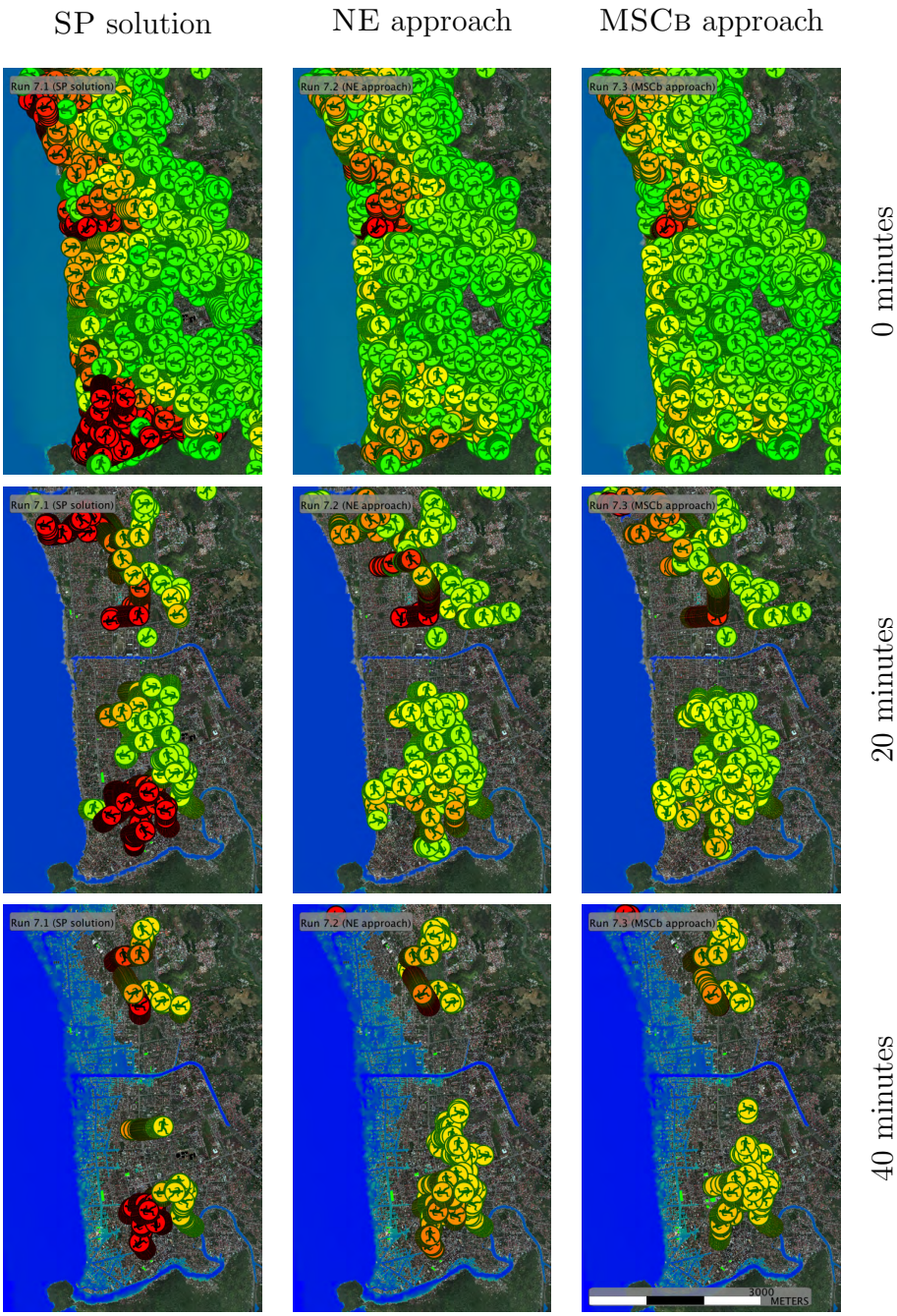


Figure 7.2: Visualizer snapshots of *Run 7.1*, *Run 7.2*, *Run 7.3* at the beginning of the evacuation (top), after 20 minutes of evacuation (center), and after 40 minutes of evacuation (bottom). The agents are colored depending on the time they need for the evacuation. A green color indicates that they escape very quickly while a red color means they need rather long time for the evacuation.

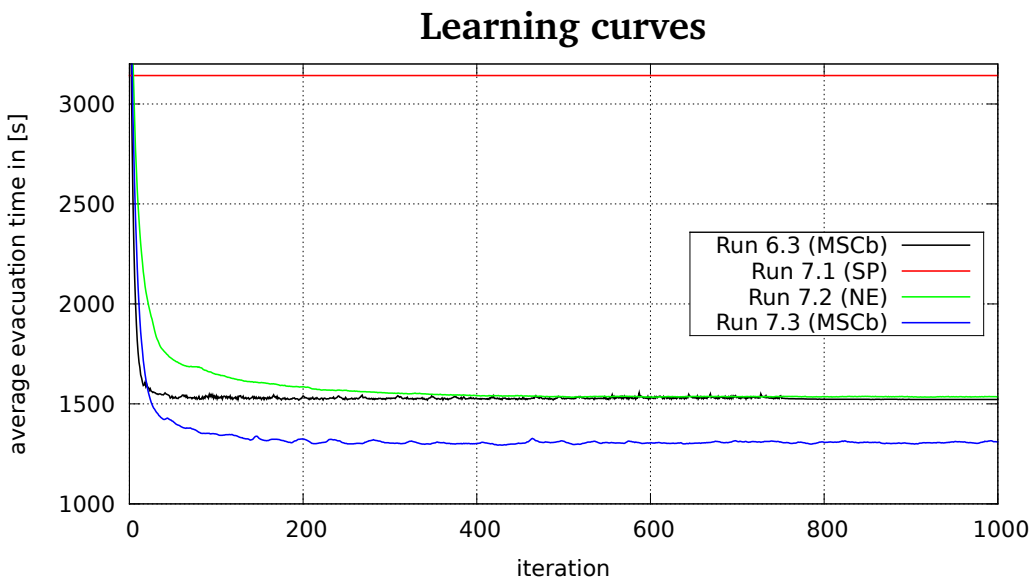


Figure 7.3: Average evacuation time over iteration number for *Run 7.1* (SP solution), *Run 7.2* (NE approach), and *Run 7.3* (MSCB approach) and for comparison *Run 6.3* (MSCB approach w/o shelters). Note that no learning iterations have been performed for *Run 7.1* therefore its average evacuation time remains constant over the iterations. For the SP solution the average evacuation time is 3 142 seconds, for the NE approach it goes down to 1 535 seconds and for the MSCB approach ends up with an average evacuation time of 1 317 seconds. It is worth mentioning that the average evacuation time for the NE approach w/ shelters is almost the same as the average evacuation time for the MSCB approach w/o shelters. Agents that do not manage to escape are ignored.

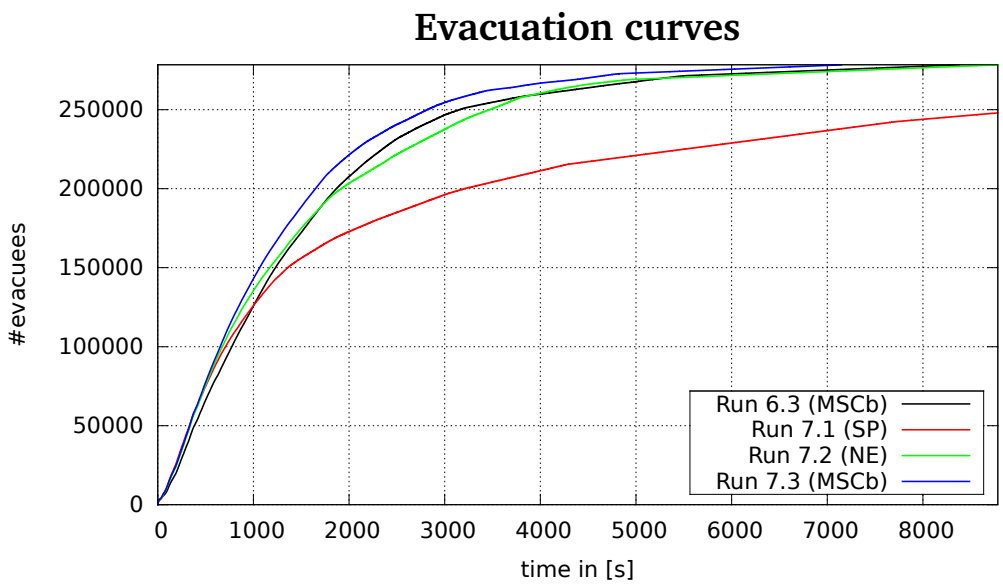


Figure 7.4: Evacuation curves of *Run 7.1* (SP solution), *Run 7.2* (NE approach), and *Run 7.3* (MSCB approach) and for comparison *Run 6.3* (MSCB approach w/o shelters). As before, the SP solution does not solve the evacuation problem. For the NE approach the total evacuation time is 8 783 seconds and for the MSCB approach it is 7 147 seconds. The MSCB approach w/ shelters (*Run 7.3*) leads to considerably better results compared to the MSCB approach w/o shelters (*Run 6.3*), where the total evacuation time is 8 381 seconds.

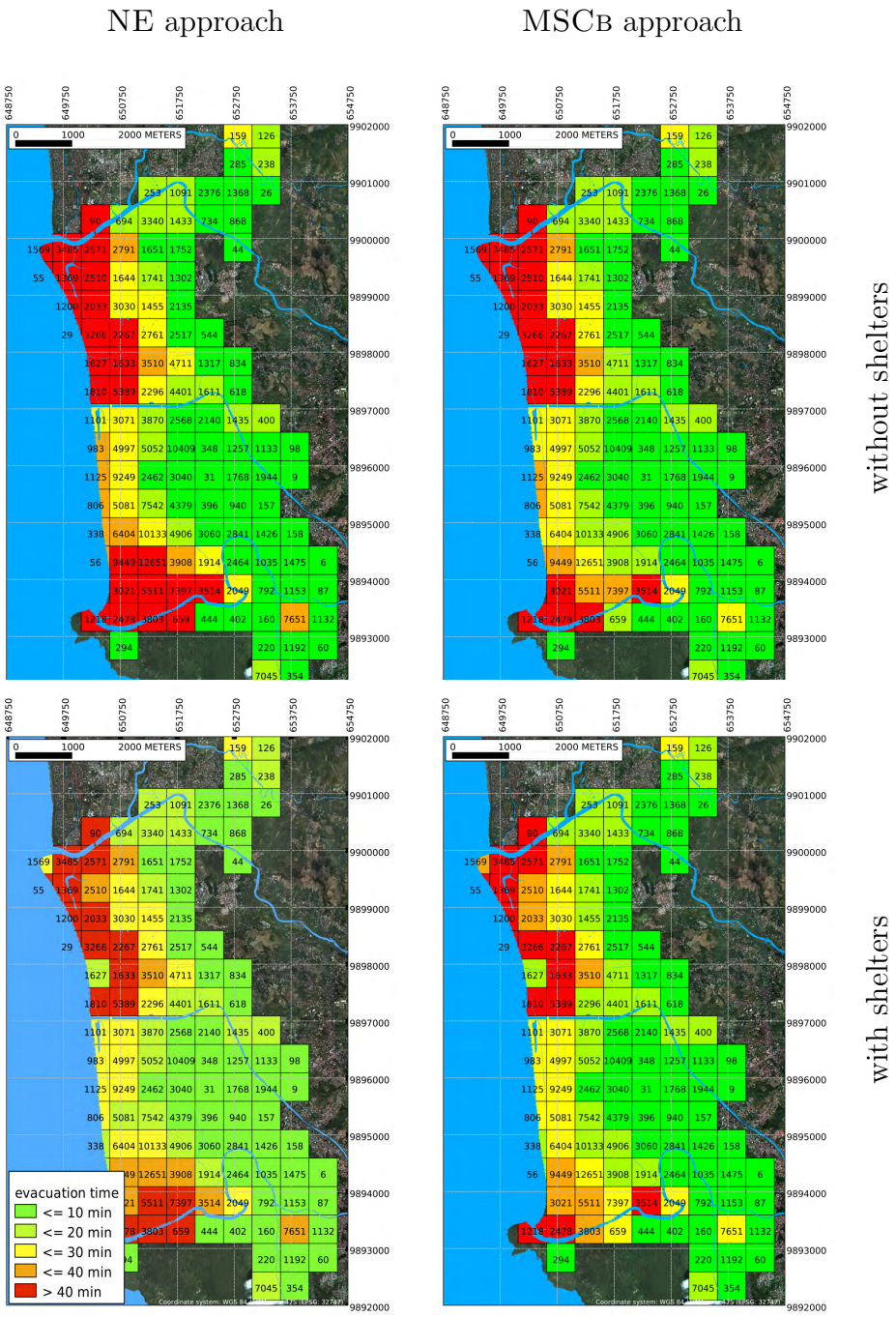


Figure 7.5: GIS analysis of the average evacuation times on a 500 meter grid. From top left to top right to bottom left to bottom right: *Run 6.2* (NE approach without shelters), *Run 6.3* (MSCB approach without shelters), *Run 7.2* (NE approach with shelters), and *Run 7.3* (MSCB approach with shelters). The cell colors describing the average evacuation time over all agents per cell. Details can be found in the legend. It is clearly shown that the evacuation with shelters significantly reduces the average evacuation time in several grid cells. The numbers in the cells describe the total number of departing agents per cell.

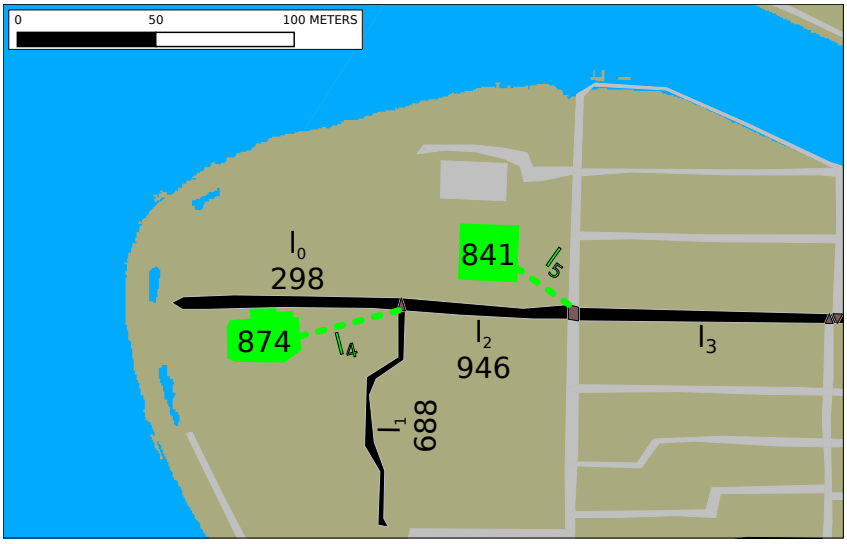


Figure 7.6: Cutout of the evacuation network with shelters to illustrate the structural problem when optimizing evacuation routes in the shelter scenario.

tsunami despite the longer evacuation times, for both the NE approach and the MSCB approach. In contrast to this, a few agents do not manage to escape in the current setup, for both the NE approach and the MSCB approach. This is a structural problem in evacuation scenarios with limited capacity sinks, which is discussed in detail in the next section.

7.5 Discussion

The introduction of shelters to the evacuation scenario significantly reduces the (average) evacuation time. However, it has been shown that not every one gains from the introduction of shelters. Chapter 6 introduces the risk reducing evacuation strategy approach. For the underlying scenario the risk reducing evacuation strategy approach finds a feasible solution, for both the NE approach and the MSCB approach, meaning every agent manages to escape. In contrast, as shown by experiments (Chapter 7.4), with shelters neither the NE approach nor the MSCB approach finds a solution where every agent manages to escape. The reason for this lies in the characteristic of shelters itself (i.e. sinks with limited capacity). This can be illustrated by small example. Consider a situation as shown in Figure 7.6. In this situation, there are four regular links (l_0, l_1, l_2 and l_3) and two shelter links (l_4 and l_5) connecting two shelters with their respectively nearest nodes. The shelter connected by link l_4 has a space capacity of 874 agents and the shelter connected by link

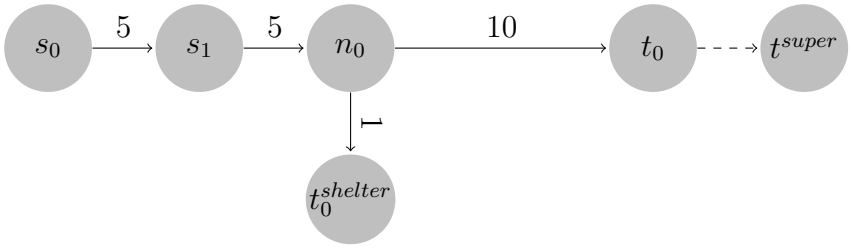


Figure 7.7: Sample evacuation network with shelter nodes

l_5 has a space capacity of 841 agents. The links l_0, l_1 and l_2 have a demand of 298, 688 and, 946 agents, meaning there is a total demand of 1 932 agents on the given three links. The demand exceeds the space capacity of the two shelters which is 1 715 in total. Furthermore, with the queue model it is not possible to overtake each other and given a relatively small flow capacity at the entrance of the shelters (usually entrance doors have a smaller width than a street) there will be spill back to the upstream link. For example congestion on link l_5 will cause spill back onto link l_2. If the 841 agents that are assigned to the shelter stand at the head of the queue, then those agents for which there is not enough space in the shelters need to wait until the last one of 841 agents has left link l_2 before they can move onto link l_3, which might be too late to reach the safe hinterland. It is obvious that for the NE approach an agent that is assigned to the shelter and stands at the head of the queue would never switch the shelter assignment with an agent that queues behind her. In the MSCB approach case an agent that stands in front of the queue would indeed switch the shelter assignment with one that stands behind her, if the agent that stands behind her would otherwise not survive. However, the probability that such two agents are chosen during the *re-planning* to switch their shelter assignment is very small, so that more than 1 000 iterations of learning are needed to find the final solution. In general, when it comes to sinks with limited capacity, there is no unique optimal solution to the evacuation problem. This illustrated by the example given in Figure 7.7. In the example the network consists of two source nodes s_0 and s_1, one intermediate node n_0, one shelter node $t_0^{shelter}$ and one general sink node t_0. The node t_0 is connected to the super sink t^{super}. The nodes are connected by links as shown in the figure. The numbers on top of the links denote the link travel times. Assume a demand of one at each source node, meaning one evacuee departs from each source node and has to evacuate either to the shelter node $t_0^{shelter}$ or to the safe node t_0. Furthermore, assume shelter node $t_0^{shelter}$ has a storage capacity of one, meaning that the shelter node can only host one of the evacuees. Taking this situation there are three different assignments

no.	assignment	average time	end time
1	(s_0, t_0), (s_1, t_0)	17.5	20
2	$(s_0, t_0^{shelter})$, (s_1, t_0)	15.5	16
3	(s_0, t_0), $(s_1, t_0^{shelter})$	15.5	20

Table 7.2: Example for different shelter assignments.

possible. An assignment is denoted by (s, t), where s is the source node and t is the target node. Table 7.2 shows the assignments and the resulting average evacuation times and also the evacuation end times (i.e. the time needed until the last one reaches the sink). Assignments 2 and 3 have the same average evacuation time of 15.5 time units, which is the optimum. This means that the system optimum (minimal average evacuation time) is not unique. However, the evacuation end time for assignment 2 is at time 16 and for assignment 3 it is only at time 20. Consequently, when it comes to limited capacity sinks then minimizing the average evacuation time does not necessarily minimize the evacuation end time nor does it maximize the number of evacuated agents for all points in time. This is because assignment 2 would achieve to evacuate the first evacuee at time 11 and second at time 16 and assignment 3 would achieve to evacuate the first evacuee already at time 6 but the second one would be evacuated only at time 20. In Chapter 3.1.3 it has been discussed that minimizing the average evacuation time maximizes the total amount of flow that has reached the sink for all points in time. A flow that realizes this property is called Earliest Arrival Flow. The example discussed clearly violates the earliest arrival property. This fact means that there exists no earliest arrival flow for limited capacity sinks. The non-existence of earliest arrival flows for this kind of problems is a well known mathematical problem (see e.g., Köhler et al. (2009)).

In situations similar to the example discussed above the evacuation end time could be reduced by charging agents for entering a shelter with the amount of external costs they cause by entering the shelter. The external costs in this case are the costs (in terms of evacuation time) that the agent queuing in front of the entry to the shelter causes to everybody in the queue behind her. In other words, this would be an extension of the marginal social cost based approach to shelters. Such an approach might solve the discussed problem but is very likely to cause ethical problems when implementing this approach in reality.

7.6 Conclusion

The most obvious evacuation direction in tsunami situations is the safe hinterland. However, if the distances are too long or there are more evacuees than the street network can cope with another evacuation strategy is needed. One way to cope with this problem is to create safe places—so called shelters—within the evacuation area. In most cases it is not possible to create a shelter space for every evacuee. In those situations one has to decide who is assigned to a shelter and who has to evacuate to the safe hinterland. This chapter introduces an approach to solve the assignment problem depending on the chosen routing strategy. Three different shelter assignments with their corresponding routing strategy are discussed. The first assignment approach is the shortest path routing with shelter assignment (SP solution). The SP solution assigns evacuees in a greedy way to shelters (nearest evacuee first); those evacuees for whom there is no space have to evacuate to the safe hinterland. All routing is done according to a shortest path routing algorithm. In the second assignment approach, evacuees are assigned and routed according to Nash constraints (NE approach). The third assignment approach assigns and routes evacuees to shelters according to the marginal social costs (MSCB approach). While the SP solution, again, only serves as a benchmark, the NE approach and MSCB approach show significant improvements for both the average evacuation time and the evacuation end time compared to an evacuation without shelters. However, with the presented approach not all problems can be solved. It is shown that not every one gains from the introduction of shelters. A promising approach to overcome the discussed problems (see above) would be to extend the marginal social cost based approach to shelters, meaning agents would be charged according to the caused external costs for entering a shelter. Such an approach might lead to a better solution, but it remains unclear how the resulting shelter assignment might be stabilized in practice since individual agents would have an incentive to deviate. However, the results from such an approach still can be used as benchmarks. Developing a corresponding model is the topic of future work. For the time being the presented approach can nevertheless help to give appropriate recommendations and estimate evacuation times. Furthermore, in the reality it would not be a real problem if only a few evacuees would not have enough space in a reachable shelter, because in reality there are no hard limits for shelter capacities. One would simply allow to slightly overload one of the reachable shelters.

Chapter 8

Conclusion and outlook

Evacuation planning, as a special case of transport planning, has been a hot topic in research for many years. Mathematical foundations for evacuation modeling were developed as early as the 1960's (Ford and Fulkerson, 1962). With the advance of computer technology, it has become manageable to solve evacuation problems of non-trivial size. Since the early 1980's computers have been sophisticated enough to solve evacuation problems for areas of the size of office buildings (see, e.g., (Chalmet et al., 1982)). Over the years, mathematical models have been applied to more and more complex evacuation problems. Nowadays, computers and mathematical models are advanced enough to solve even large-scale evacuation problems with hundreds of thousands evacuees (see, e.g., (Dressler et al., 2011)). However, these mathematical models optimize evacuation problems from a global perspective, which means that the evacuation performance of a single individual does not matter as long as the global evacuation performance is optimal. The system optimum, which minimizes the average evacuation time, is one example for such an objective.

Another approach to solve the evacuation problem is the optimization from the single evacuee's perspective, where every evacuee chooses an evacuation strategy that is optimal for herself under the given circumstances. Such a state is called Nash equilibrium (Nash, 1951). By now, there is no mathematical model to calculate a Nash equilibrium for evacuation problems of non-trivial size (Köhler et al., 2009). In principle, a Nash equilibrium can be reached through a simulation based approach. Cascetta (1989) has shown how a Nash equilibrium for general transport problems can be reached through an iterative learning algorithm. Although the convergence can only be proved under certain conditions, in general, the iterative approach finds solutions that are close enough to a Nash equilibrium at least from the practical point of view. Moreover, a simulation based approach gives more flexibility to test and apply various aspects that are important to find feasible and realistic solutions for evacuation problems.

The main aim of this thesis is to solve the evacuation problem for a tsunami threatened city by a simulation based approach. In the introduction it is stated that the evacuation problem for such a scenario includes the following sub-problems:

- *Evacuation routing* – The routing problem is to find an appropriate evacuation route for every evacuee. In general, evacuation routes are assigned to reduce individual travel times (Nash routing) or to reduce the system travel time (system optimum).

- *Time-dependent aspects of the danger* – The flooding will not cover all of the danger zone at once, meaning that while some districts of the city are already inundated other districts may still be passable. This aspect has to be considered when developing evacuation strategies.

- *Risk reduction* – The objective of a risk reducing evacuation strategy is to find routes that avoid unnecessary risk.

- *Shelter assignment* – Shelters are safe places with limited space capacity inside the evacuation area (i.e. buildings for vertical evacuation). Problems that arise when it comes to shelters are: where to place them, of which size they must be and who are allowed to use them.

The theoretical basis and approaches for solving these sub-problems in a simulation framework are discussed in Chapter 3. Three different routing strategies are introduced there. The most straightforward solution is the shortest path solution (SP solution), where everyone chose the shortest path to the safe area. However, the shortest path solution does not take congestion into consideration and, consequently, it is likely that the SP solution underestimates the expected travel times. For this reason, two other routing strategies are introduced as well, namely the Nash equilibrium approach (NE approach) and the marginal social cost based approach (MSCB approach).

The NE approach has the advantage that nobody can gain by unilateral deviation and, therefore, nobody has an incentive to deviate once a Nash equilibrium is reached. In a real world situation, a Nash equilibrium can be achieved by appropriate training.

The third routing strategy that is discussed is the MSCB approach. Where the system travel time is approximately minimized. However, this approach requires cooperative behavior (individual minimization of the social impact of ones behavior) and does, therefore, not follow an intrinsic motivation of the evacuees. For that reason, this kind of behavior has to be enforced external. As a result, the MSCB approach is a more efficient approach compared to NE approach but at the same time harder to implement.

Chapter 4 explains how the three routing strategies are implemented within the MATSim framework[1]. MATSim is an agent based iterative learning framework for transport problems. In principle, MATSim works as follows:

[1]For more information see http://matsim.org (accessed December 2010)

A simulation starts with an initial demand as *input*. In the base case, as discussed in Chapter 4.1, the *input* consists of the simulation network and a set of agents, where every agent corresponds to a person in the real world. Each agent has an initial plan. A plan in its simplest form consists of an origin, a destination, and a route from the origin to the destination. The agents' plans are executed in the traffic flow simulator, also called mobility simulation (*mobsim*). Afterwards, the *analysis* module calculates the score of each agent's plan based on its performance in the *mobsim*. In addition, the *analysis* module also aggregates the experienced travel costs that are needed for *re-planning*. After the *analysis* of the performed mobility simulation has been finished, MATSim can either terminate if the predetermined number of learning iterations has been reached, or continue the learning cycle by running the *re-planning* module. During *re-planning* the agents adapt their plans based on the experienced travel costs or select already existing plans out of there memory for renewed execution. After a simulation run is finished, *output* data will be dumped so it can be used for further appraisal. This general work flow of MATSim also works for evacuation scenarios. However, several modules have to be adapted depending on the wished objective. The necessary adaptations are discussed in Chapter 4.1.

The implementation of the three different routing approaches in MATSim has been tested on a real world scenario. The scenario describes the situation in the city of Padang in the case of a tsunami warning. Padang is located at the West Coast of Sumatra Island in Indonesia. In the case of a tsunami, it is expected that about 300 000 people are affected, which means the evacuation problem is to find feasible evacuation plans for more than 300 000 people in an urban area. Details on the scenario generation are discussed in Chapter 4.2.

Several experiments have been conducted based on the Padang scenario. The results are discussed in Chapter 4.3. As mentioned above, the simulation based approaches, in general, only solve the objectives approximately. However, the experiments have shown that the NE approach and the MSCB approach generate similar results as a combinatorial optimization approach that has been proposed in Dressler et al. (2011). The combinatorial optimization model solves the evacuation problem on an abstraction of MATSim's physical model. For that reason, the combinatorial optimization model does not "know" the real physics of MATSim's mobility simulation, and so it is not surprising that the MSCB approach performs better than the combinatorial optimization model. These results have shown that from the practical or engineers point of view the simulation approaches are close enough to their respective objectives. However, while the simulation-based approaches do well when compared with a mathematical solution, it became apparent that the achieved results lack of practical relevance. The SP solution do not find a

feasible solution since this approach does not take congestion into consideration and, therefore, the SP solution pretends unrealistic evacuation times. A comparison of the pretended and the resulting evacuation times is shown in Figure 4.11. This means: **The shortest path solution is likely to underestimate the travel and therefore it is not qualified for evacuation planning.**

However, even the NE approach and the MSCB approach does not find a feasible solution, because with the basic setup many evacuees are fleeing in parallel to the shoreline to reach safe locations. Even if this leads to a faster evacuation, such behavior is not desirable. The evacuees should move away from the approaching tsunami instead. This undesired behavior is a result of the agents' unawareness about when and where the tsunami hits the land. This means: **In order to find feasible evacuation solutions the time-dependent aspects of the danger have to be explicitly modeled.**

Chapter 5 discusses an approach how the time-dependent aspects are modeled within MATSim. Experiments showing that with explicit consideration of the time-dependent flooding expansion, the resulting evacuation plans have much more practical relevance. Still, the explicit modeling of the time-dependent spreading of the danger only works if the advance warning time is known beforehand. Even small inaccuracies in the knowledge about the advance warning time can make a solution invalid despite the explicit modeling of the time-dependent aspects. Since MATSim is not entirely deterministic (e.g. the right of way in the simulation is handled stochastically), the expected travel times, even for the NE approach and the MSCB approach, are not always equal to the resulting ones. For that reason, some agents "think" they will make it just in time to safety along a certain road, but instead they get caught by the waves. The stochastic nature of the simulation can be seen as an uncertainty in the advance warning time. The higher this uncertainty is, the higher is the risk that not all evacuees will manage to escape. In order to reduce the impact of this uncertainty, **risk should be explicitly modeled, which calls for a risk reducing evacuation strategy.**

Chapter 6 introduces an evacuation strategy that avoids unnecessary risk in situations with uncertain advance warning time. The risk reducing strategy is demonstrated on the Padang example. It is assumed that the spreading of the inundation is known beforehand. The actual advance warning time, however, may be uncertain. That means, that in spite of the unknown advance warning time, locations in the city still can be sorted in a chronological order depending on which location becomes flooded first. The risk reducing strategy penalizes moves from a location that becomes flooded later to a location that becomes flooded earlier, since those moves are considered as risky. The same approach also works in a spatial buffer around the inundation area, where the

risk evaluation is performed depending on the distance to inundation's maximum expansion. The risk penalty is modeled as a static cost offset to the respective route cost. This approach is similar to the priority evacuation discussed by Hamacher and Tufekci (1987). As a result, the behavior changed in a way that, at least for the Padang scenario, everyone manage to escape. The risk cost approach significantly reduces the number of potential evacuation paths since many risky, but maybe fast, evacuation paths are banned. This leads to considerable longer evacuation times, compared to the approach that does not consider any risk. As a side effect the evacuation behavior resulting from the NE approach and the MSCB approach are similar with respect to the average evacuation time and evacuation end time, with a small advantage for the MSCB approach over the NE approach. At least for situations, where the evacuation times for a Nash equilibrium approach are not much worse compared to an optimal solution, the Nash equilibrium is to prefer. That is because, a Nash equilibrium is a more stable solution since no one has an incentive to deviate, while an optimal solution does not follow an intrinsic motivation of the evacuees, but has to be enforced externally.

The results of the risk reducing strategy for the Padang scenario makes clear that at some places the advance warning time is just long enough to get everyone out. However, in general, it might be often the case that even the best evacuation strategy would not leave enough time for an evacuation. In those situations safe place—so-called shelters—could be build inside the evacuation area. An approach that deals with shelters is discussed in Chapter 7. In many cases there may be buildings within the hazard zone that could serve as shelters. Shelters are safe places with limited space capacity inside the evacuation area. The existence of shelters changes the evacuation scenario, so that common approaches to solve the evacuation problem are not longer applicable, because the agents have to be assigned to the shelters (i.e. to the sinks) before they can be routed. The situation becomes even more complicated if there is not enough shelter space for all evacuees. In those situation some of the agents, depending on the available shelter space, still have to evacuate to the safe hinterland. In this work an iterative learning approach is proposed that extends the discussed routing solutions in order to to find feasible shelter assignments for the SP solution, the NE approach, and the MSCB approach.

It turned out that the integration of shelters, as expected, significantly reduces the evacuation time. There is, however, a structural problem when it comes to shelters, for which there seems to be no straightforward solution. For the Padang scenario it has been shown that not every one gains from the introduction of shelters. The risk reducing evacuation strategy approach, introduced in Chapter 6, finds a feasibly solutions for both the NE approach and the MSCB approach, meaning every agent manages to escape. When

it comes to shelters, however, neither the NE approach nor the MSCB approach find a solution where every agent manages to escape, despite of the shorter average evacuation times. The reason for this lies in the characteristic of shelters itself (i.e. sinks with limited capacity) and cannot easily be overcome. A promising approach would be to charge agents who enter a shelter early because they have more time to reach the safe hinterland and agents who arrive late at the shelter had to be charged less or not at all. Developing a corresponding model is a topic of future work. For the time being the presented approach can nevertheless help to give appropriate recommendations and estimate evacuation times. Furthermore, in reality it would not be a real problem if only a few evacuees would not have enough space in a reachable shelter, because in reality there are no hard limits for shelter capacities. One would simply allow overloading one of the reachable shelters slightly.

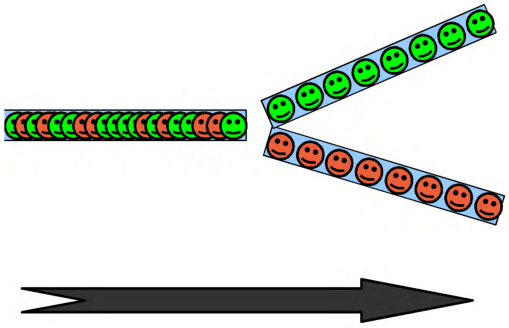
Some other open problems that have not been addressed is this thesis will be explained a little bit more in the following.

Open questions for the shelter problem are: Where to build new shelters, and of which capacity they must be? This problems belong to the classes of allocation problems for which a large body of work can be found in literature (see, e.g., (Lozano et al., 1998; Chowdhury et al., 2001; Hauskrecht and Singlair, 2003)). The main problem here is to solve the allocation problem on top of a dynamic network flow problem. In principle, this problem can also be solved be an iterative learning algorithm. A proposal for an implementation to solve the allocation problem is given in (Flötteröd and Lämmel, 2010).

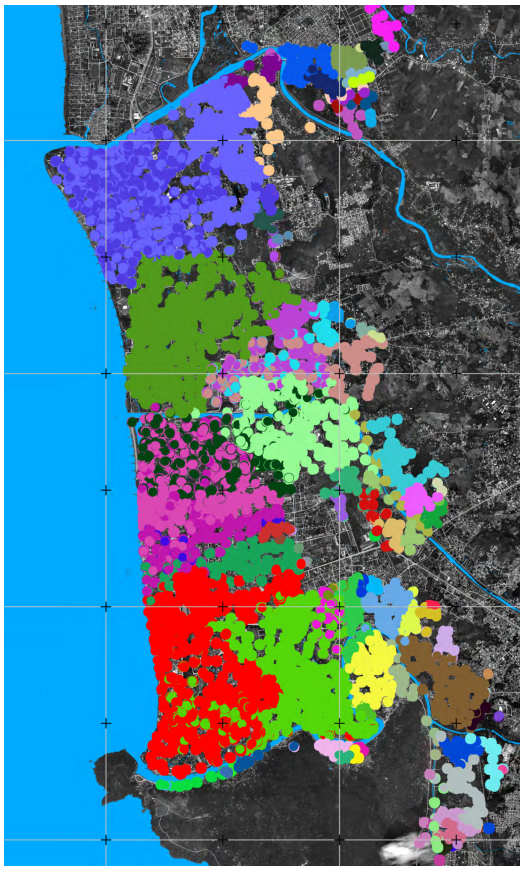
Another issue concerns the confluent flow problem. A solution would be confluent if at any node all the flow leaves along a single fixed link (see, e.g., (Chen et al., 2004)). From such a solution one could directly derive evacuation recommendations by applying a sign at every node (i.e. intersection) that points towards this link. In the case of an evacuation, every evacuee simply would have to follow these signs. In the results found in this work only the SP solution provides a confluent flow. This is because, in general, there is only one shortest path from any node to any other node in a network. For the NE approach and MSCB approach this, however, do not hold. In the corresponding results presented in this thesis, there are

- Links where evacuation flows diverge into multiple streams with possibly different destinations (Figure 8.1(a)).
- Regions where agents from neighboring starting locations evacuate to different safety locations (Figure 8.1(b)).

The former is a consequence of the situation that a high capacity link is followed by lower capacity links. In order to use the full capacity of the



(a) Illustration of a diverging evacuation flow



(b) Home locations of the agents colorized depending on their evacuation destination in *Run 6.2*

Figure 8.1: Illustrations why found solutions are not confluent

system, it is in this situation necessary that the stream diverges. Consequently confluent solutions would make no sense in most scenarios anyway. The latter is a consequence of the respective routing approach (NE approach or MSCB approach). Let us consider the example of the NE approach: If all exits are similarly congested, it does not matter which exit one uses. This is in part directly caused by the stream divergence problem: If there is a stream divergence, all locations upstream of the divergence point will, in equilibrium, have both options with the same cost. One approach that has been tried out in Lämmel et al. (2009) is to add the geographical distance to the respective cost function. However, those additional costs do not remove the effect, and for that reason this approach has not longer been applied. One solution to overcome the safe locations problem would be a follower-leader model (see, e.g.,(Murakami et al., 2003)), where leaders guide groups of evacuees to safe locations. In a multi-agent based evacuation simulation the agents would have to be divided into followers and leaders. The leaders would define evacuation groups and the followers would be assigned to these groups. The leaders decide which evacuation route to take. In MATSim one would let only leader agents re-plan, and their evacuation plans then would be injected to their respective group members. For the optimization of this procedure, various objectives are thinkable. For instance, one could try to minimize the average evacuation time of a group or the evacuation time of the group member who performs worst. The determination of the various evacuation groups and their respective leaders should be undertaken based on real-life data.

The last important topic of future research that should be discussed here is the post-disaster planning. It has been shown that the post-disaster management often has to cope with enormous difficulties to reach and supply the affected, possibly displaced, people. Recent examples are: the problems with the supply of the people affected by hurricane Katrina in 2005 and even more seriously the situation after the 9.0 quake off the coast of east Japan on March 11th, 2011. The earthquake was followed by a devastating tsunami, which not only wiped away entire villages in the coastal area in northeast Japan but also seriously damaged the Fukushima Daiichi nuclear plant. As result there are more then 300 000 displaced people living in temporary evacuation centers. In the wake of such large-scale disasters the first task is to find out where to help and how to deploy the usually limited resources to cope with the crises. A multi-agent based simulation, as a decision support system, could help to allocate the existing means in a responsible manner.

Appendix A
Visualization

Visualization is a key element of a simulation. Without visualization simulation results are hard to interpret. With OTFVis, MATSim provides a OpenGL based visualizer (Strippgen, 2009). OTFVis provides basic hardware accelerated visualization features, such as network visualization and the visualization of moving vehicles. In the evacuation context, it would be good to have some other evacuation related aspects visualized too. For that reason, some extensions have been developed for OTFVis. The most important extension that has been made, is the capability to visualize time-dependent flooding data. The underlying flooding data for Padang comprises water heights for 1 074 627 coordinates on a $1\,min$ time raster. For the visualization a time horizon $120\,min$ has been chosen. This results in 128 955 240 values of water heights. Even if each value were stored as a *float* each value would consume 32bit of memory, meaning that a total of $128\,955\,240 * 32bit \approx 500MiB$ would be needed for the flooding data alone. This calls for data compression. Since the visualization is independent of the simulation lossless data compression is not needed. However, one requirement is that the time when a location gets flooded in the visualization must not be later as for the simulation. This gives a hint for a first step of data reduction. Most parts of the city get flooded about 30 minutes after the quake. Lets consider $t^{flooding}$ as the time when the location at coordinate c gets flooded and T the time horizon of $120\,min$. Then the flooding series for coordinate c can be seen as a periodic signal with a period length of $N = T - t^{flooding}$. A periodic signal can be approximated by a series of Walsh functions (Walsh, 1923). The periodic signal is represented as the sum of Walsh functions that differ in amplitude and sequence. Lets consider the signal as time discrete (discretized into N segments of equal length), then the flooding height at time n can be approximated as follows:

$$f_n = \sum_{i=0}^{N-1} c_i \, \texttt{wal}(i, \frac{n}{N}), \tag{A.1}$$

where c_i denotes the ith Walsh coefficient and $\texttt{wal}(i, \frac{n}{N})$ the value of the ith Walsh function at n/N. A Walsh function is a periodic function on an interval from 0 to 1 and can have a value of either -1 or 1. The values of the first 16 Walsh functions are:

```
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
{1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1},
{1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1,1,1,1,1},
{1,1,1,1,-1,-1,-1,-1,1,1,1,1,-1,-1,-1,-1},
{1,1,-1,-1,-1,-1,1,1,1,1,-1,-1,-1,-1,1,1},
{1,1,-1,-1,-1,-1,1,1,-1,-1,1,1,1,1,-1,-1},
{1,1,-1,-1,1,1,-1,-1,-1,-1,1,1,-1,-1,1,1},
{1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1},
{1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1},
{1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1},
{1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,1,-1,-1,1},
{1,-1,-1,1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1},
{1,-1,1,-1,-1,1,-1,1,1,-1,1,-1,-1,1,-1,1},
{1,-1,1,-1,-1,1,-1,1,-1,1,-1,1,1,-1,1,-1},
{1,-1,1,-1,1,-1,1,-1,-1,1,-1,1,-1,1,-1,1},
{1,-1,1,-1,1,-1,1,-1,1,-1,1,-1,1,-1,1,-1}.
```

The coefficients c_i can be calculated as follows:

$$c_i = \frac{1}{N} \sum_{n=0}^{N-1} f_n \, \mathtt{wal}(i, \frac{n}{N}) \tag{A.2}$$

In general there is no fixed number of Walsh functions that is needed in order to approximate a given signal. However, it is obvious that as more Walsh functions are used as better the approximation becomes. On the other hand, as less Walsh functions are used as less coefficients needs to be stored and as higher the compression is. The relation between quality of the approximation and number of used Walsh functions is best seen on a comparison between the original signal and the corresponding Walsh approximations. Therefore two sample locations have been chosen arbitrary. The locations are shown in Figure A.1. Location $c1$ is located at the shore and location $c2$ is locate in town some $100\,m$ away from the shore. The original regime of water heights versus time for location $c1$ is depicted by the black curve in Figure A.2 (a) and analogous for $c2$ in Figure A.2 (c). The corresponding Walsh approximations with 4,8, and 16 Walsh functions are given by the red, green, and blue colored curve respective. As expected the approximation with 16 Walsh functions gives the best results. However, in the underlying case (visualization) the actual water heights play only a minor part. It is more important to see if and when a location becomes flooded and with that impassably. From this point of view, the approximation with only 8 Walsh functions seems still to be reasonable. However, Walsh functions are square functions and therefore the approximation results are also “squarish”. This would look odd in a

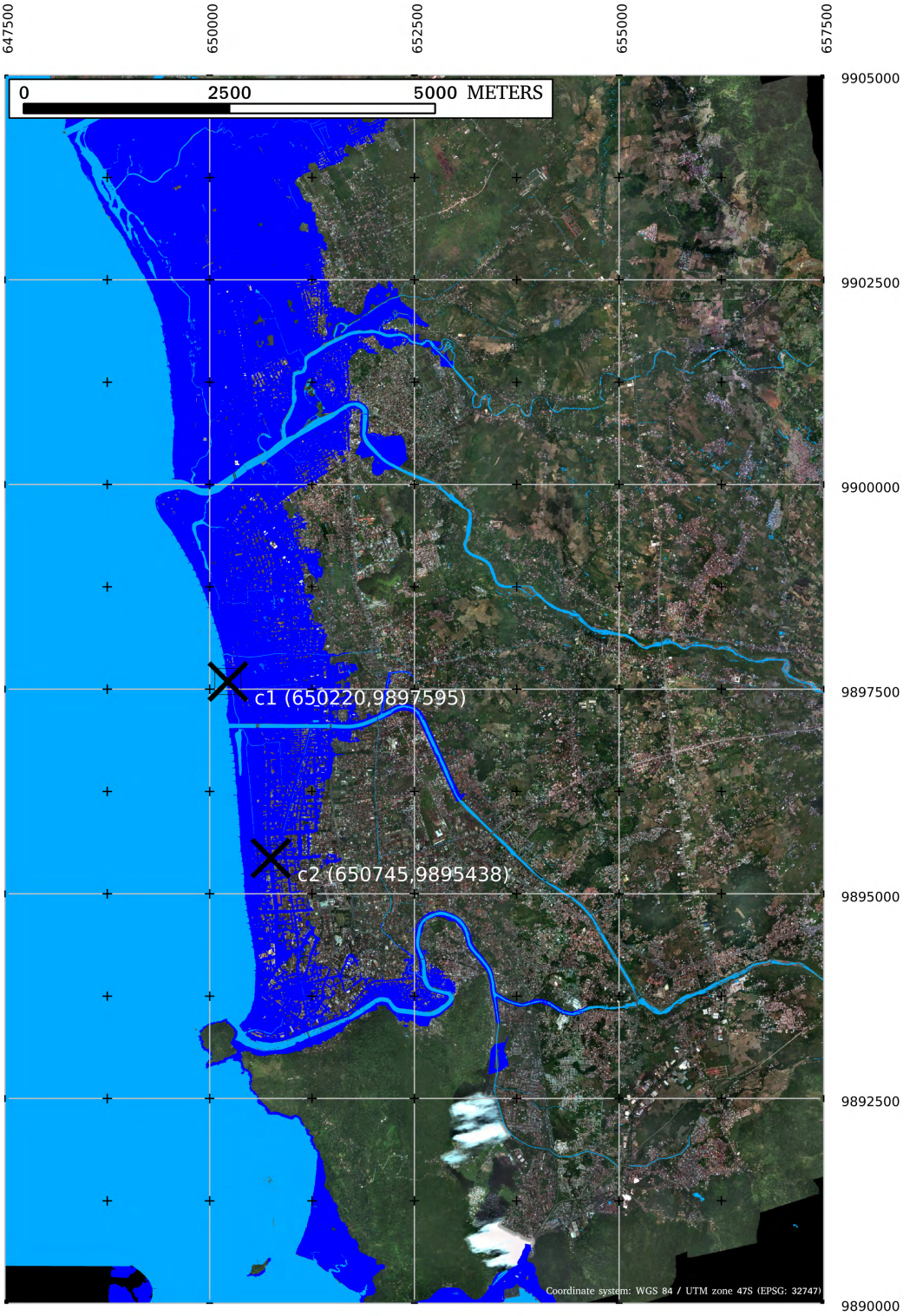
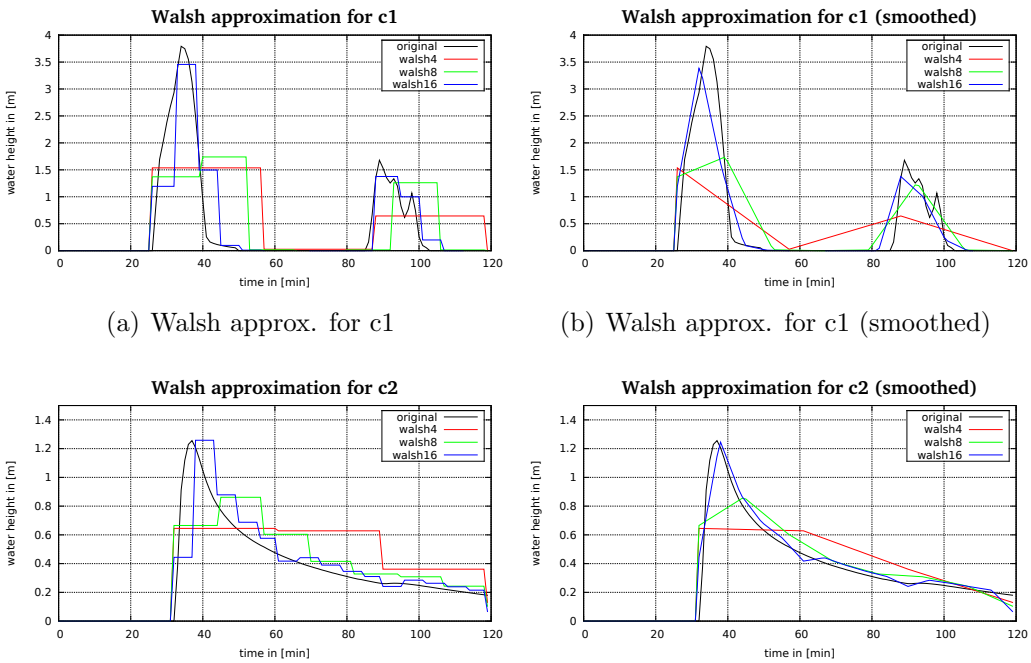


Figure A.1: Sample locations to demonstrate the data compression.



(a) Walsh approx. for c1

(b) Walsh approx. for c1 (smoothed)

(c) Walsh approx. for c2

(d) Walsh approx. for c2 (smoothed)

Figure A.2: Walsh approximation of the water heights at location c1 and c2 versus time. The black colored curves are showing the original regime and the red, green, and blue colored curve the approximation with 4,8, and 16 Walsh functions. Basic approximation results are shown in Subfigures (a) and (c) and Subfigures (b) and (d) are showing the resulting curves after linear smoothing.

visualization because the water would appear (and disappear) from one time step to the next. For that reason the curves have been smoothed through a linear interpolation. The results are shown in Figures A.2 (b) and (d). As discussed above, in the original data set the water heights are given on a $1\,min$ time raster for a period of $120\,min$, meaning that for each location 120 water height values have been stored. If the flooding series is approximated with 8 Walsh functions only 9 values ($t^{flooding}$ and $c_0, \ldots, c_7$) per location have to be stored. The original data set had a memory consumption of about $500\,MiB$ with the introduced compression this is reduced to $1\,074\,627 * 9 * 32bit \approx 40\,MiB$.

Appendix B

Storage and retrieval of link travel times

MATSim was originally developed for the simulation of vehicular traffic for large cities or even regions. For this kind of simulation a temporal resolution of 15 *min* for the link travel times is used: The link travel times are aggregated in 15 *min* time bins and the travel time values are stored in arrays (one array for each link). For a network consisting of 100 000 links, this means a memory consumption of about 75 MB[1] for the link travel time values.

For a pedestrian evacuation simulation with an expected evacuation time of one or two hours, a resolution of 15 *min* is too coarse. However, a finer resolution increases the amount of memory that is needed. To overcome this problem, the array-based implementation was replaced by an implementation using Java HashMap. There is a HashMap for each link that stores the link travel times. But only for those travel time bins where at least one agent has entered the link, a travel time value is stored. This means that if for a given travel time bin no agent enters the corresponding link, then nothing is stored (and the free speed travel time will be used). Depending on the scenario this can save a lot of memory.

An analytic appraisal of the memory usage or the needed execution time in Java is difficult and depends, besides others, on the virtual machine and its garbage collection mechanism, and on the specific scenario. Therefore it was decided to compare the HashMap based approach with the array-based approach through a benchmark scenario. In the benchmark scenario, all link travel times of a simulation run have been recorded and aggregated for different travel time bin sizes. Beginning with a travel time bin size of 15 *min* and ending with a travel time bin size of 1 *min*, the memory consumption, the time for storing and aggregating and the time for the retrieval of the link travel times have been measured.

The results of this benchmark test are shown in Fig. B.1. At the top there is a diagram comparing the memory usage of both approaches. Clearly, the HashMap based approach consumes considerably less memory than the

[1] For 24 *h* day there are $4 * 24$ time bins, each time bin holds one double value (64 bit) and there are 100 000 links ($4 * 24 * 100\,000 * 64$ bit ≈ 75 MB).

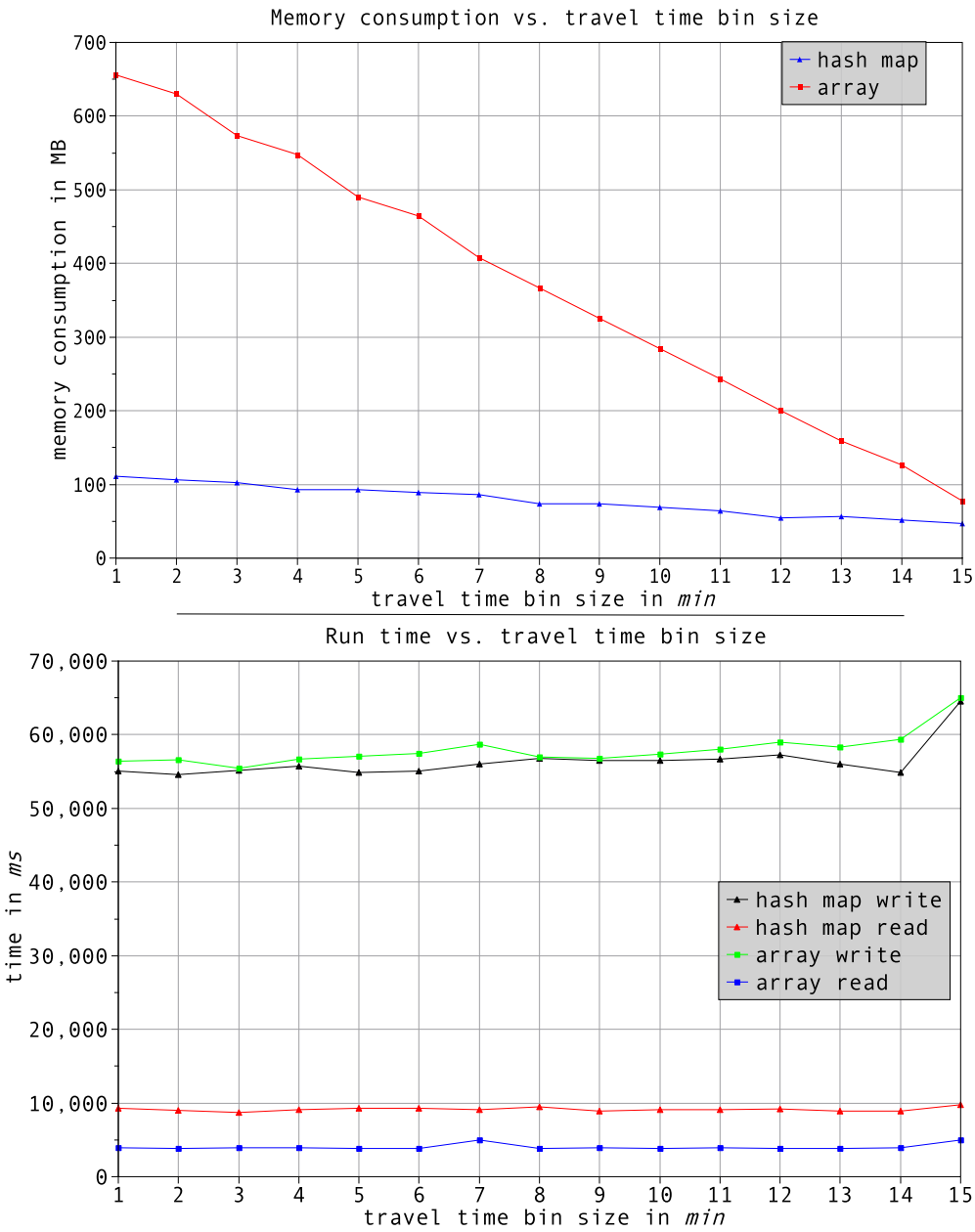


Figure B.1: Comparison of two different implementations of the storage and retrieval mechanism for the time-dependent link travel times. Top: comparison of the memory consumption. The Java HashMap based approach consumes considerable less memory than an array-based implementation. Bottom: comparison of the runtime performance. For write operations both approaches have almost equal execution times. The array-based approach is faster for read operations.

array-based implementation. In particular, at 1-min time resolution the array-based implementation consumes about 650 MB of memory, which, together with the memory requirements of the remainder of the package, would make simulations on today's ordinary desktop computers impossible. The diagram at the bottom compares the runtime for storage and aggregation (write) and for retrieval of the link travel times (read). The time needed for write operations is almost equal for both approaches. For read operations the array approach is faster. At the same time, the execution time for a read operation is much smaller than for a write operation. The underlying simulation run simulates the evacuation of about 320 000 agents. On average each agent has to traverse 23.5 links before she reaches the safe area. This means there are approximately 75 000 000 different link travel times that have to be aggregated and stored. For the benchmark scenario there were approximately 250 000 000 synthetic read operations[2]. Overall, for the time-dependent link travel times retrieval, the small disadvantage of the hash map approach with respect to the runtime can be tolerated for the sake of much lower memory consumption.

[2]The network for the scenario consists of 16 978 links. For each link the travel time was queried 20 times for 720 different time steps. ($16\,978 * 20 * 720 \approx 250\,000\,000$)

Appendix C

Retrieval mechanism for network change events

Network change events can be queried in arbitrary chronological order. As a consequence of this flexibility, the retrieval mechanism is not straightforward. Two different approaches have been tested. The first approach that has been tested is to store the network change events in a Java TreeMap. The Java TreeMap implements a red-black tree (Cormen et al., 2009) with a complexity of $O(log\ n)$ for get operations. The second approach that has been tested is to store the network change events in an array in chronological order and use a binary search to find the corresponding entry. The complexity for a binary search is also a $O(log\ n)$. However, the Java TreeMap cannot operate on primitives (double, int) but the primitives have to be converted into objects (Double, Integer). Since Java 5 this conversion is done automatically (*autoboxing*). The *autoboxing* mechanism gives the software developer more flexibility, but produces an overhead that increases the run time. A small benchmark scenario illustrates this issue. For both approaches the retrieval time for network change events was measured. The number of network change events was successively increased. To get a robust result, the time was taken over 10 000 000 queries. The test was performed on a 2.33 GHz CPU. Fig. C.1 shows that the TreeMap implementation is about 20% slower than the implementation using arrays and binary search.[1] In consequence, the array approach was chosen to implement the time dependent network.

[1] In fact, the author and others (Meyers, 2002) (Item 23) consistently find that in situations with few or no insertions or removals after initialization, the array-based approach is faster.

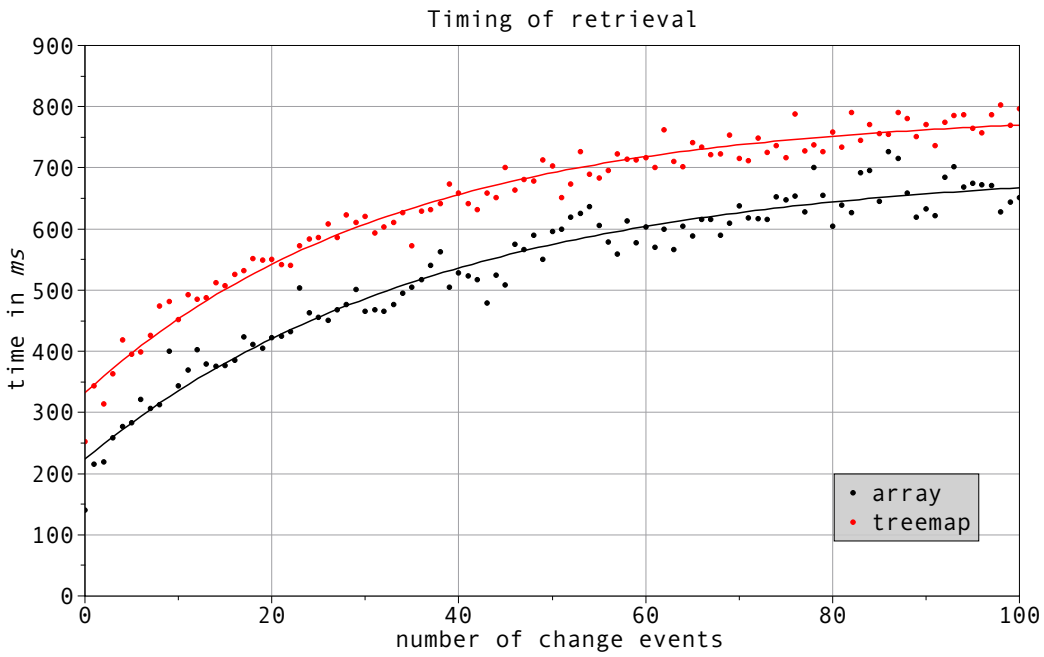


Figure C.1: Comparison of the runtime performance for two different implementations of the network change event retrieval. The array-based approach using binary search is about 20% faster then the approach using a Java TreeMap. The complexity of both approaches is $O(log\ n)$.

Bibliography

E. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines*. Wiley, 1989. (Cited on page 33.)

R. Alsnih and P. Stopher. A review of the procedures associated with devising emergency evacuation plans. Working Paper ITS-WP-04-04, Institute of Transport Studies, The University of Sydney and Monash University, 2004. See www.itls.usyd.edu.au/publications/working_papers/wp2004/its_wp_04-04.pdf. (Cited on page 10.)

M. Barbehenn. A note on the complexity of dijkstra's algorithm for graphs with weighted vertices. *IEEE Transactions on Computers*, 47:263, 1998. (Cited on page 18.)

B. Barrett, B. Ran, and R. Pillai. Developing a dynamic traffic management modeling framework for hurricane evacuation. Transportation Research Board Annual Meeting Preprint 00-1595, Washington D.C., 2000. (Cited on pages 3, 4 and 10.)

M. Beckmann, C.B. McGuire, and C.B. Winsten. *Studies in the Economics of Transportation*. Yale University Press, 1956. (Cited on pages 20 and 24.)

M. Ben-Akiva and S. R. Lerman. *Discrete choice analysis*. The MIT Press, Cambridge, MA, 1985. (Cited on page 51.)

E.M. Bernardino, A.M. Bernardino, J.M. Sánchez-Pérez, J.A. Gómez-Polido, and M.A. Vega-Rodríguez. Hybrid population-based incremental learning to assign terminals to concentrators. In *Proceedings of the International Conference on Evolutionary Computation*, 2010. (Cited on page 33.)

M. Bierlaire, G. Antonini, and M. Weber. Behavioral dynamics for pedestrians. In K.W. Axhausen, editor, *Moving through nets: The physical and social dimensions of travel*. Elsevier, 2003. (Cited on page 10.)

A. Bleiweiss. Multi agent navigation on the gpu. Technical report, NVIDIA Corporation, 2008. URL `http://developer.download.nvidia.com/presentations/2009/GDC/Multi-AgentGPU.pdf`. (Cited on page 15.)

Joseé C. Borrero, Kerry Sieh, Mohamed Chlieh, and Costas E. Synolakis. Tsunami inundation modeling for western sumatra. In Barbara A. Romanowicz, editor, *PNAS*, volume 103, pages 19673–19677, 2006. (Cited on page 5.)

Michael Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press, 1987. (Cited on page 13.)

G.W. Brown. Iterative solutions of games by fictitious play. In T.C. Koopermans, editor, *Activity Analysis of Production and Allocation*, pages 374–376, New York, 1951. Wiley. (Cited on page 21.)

E. Cascetta. A stochastic process approach to the analysis of temporal dynamics in transportation networks. *Transportation Research B*, 23B(1): 1–17, 1989. (Cited on pages 21 and 106.)

E. Cascetta and C. Cantarella. A day-to-day and within-day dynamic stochastic assignment model. *Transportation Research A*, 25A(5):277–291, 1991. (Cited on page 12.)

D.G. Cattrysse and L.N. Van Wassenhove. A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60:260–272, 1992. (Cited on page 33.)

N. Cetin, A. Burri, and K. Nagel. A large-scale agent-based traffic microsimulation based on queue model. In *Proceedings of the Swiss Transport Research Conference (STRC)*, Monte Verita, CH, 2003. See www.strc.ch. Earlier version, with inferior performance values: Transportation Research Board Annual Meeting 2003 paper number 03-4272. (Cited on page 15.)

L. Chalmet, R.L. Francis, and P. Saunders. Network models for building evacuation. *Management Science*, 28:86–105, 1982. (Cited on pages 23, 30 and 106.)

J. Chen, R.D. Kleinberg, L. Lovász, R. Rajaraman, R. Sundaram, and A. Vetta. (Almost) tight bounds and existence theorems for confluent flows. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, STOC '04, pages 529–538, New York, NY, USA, 2004. ACM. ISBN 1-58113-852-0. doi: http://doi.acm.org/10.1145/1007352.1007432. URL `http://doi.acm.org/10.1145/1007352.1007432`. (Cited on pages 36 and 111.)

X. Chen and F.B. Zhan. Agent-based modeling and simulation of urban evacuation: Relative effectiveness of simultaneous and staged evacuation strategies. Transportation Research Board Annual Meeting Preprint 04-0329, Washington D.C., 2004. (Cited on page 9.)

Y.-C. Chiu, P. Korada, and P.B. Mirchandani. Dynamic traffic management for evacuation. Transportation Research Board Annual Meeting Preprint 05-2369, Washington D.C., 2005. (Cited on pages 9 and 12.)

H.I. Chowdhury, T.L. Chambers, and V. Zaloom. The use of simulated annealing to solve large transportation-location problems with euclidean distances. In *Proceedings of the International Conference pm Computers and Industrial Engineering (29th ICC&IE)*, 2001. (Cited on pages 31 and 111.)

T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009. (Cited on pages 18 and 122.)

E. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959. (Cited on pages 18 and 93.)

D. Dressler, G. Flötteröd, G. Lämmel, K. Nagel, and M. Skutella. Optimal evacuation solutions for large-scale scenarios. In Bo Hu, Karl Morasch, Stefan Pickl, and Markus Siegle, editors, *Operations Research Proceedings 2010*, Operations Research Proceedings, pages 239–244. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-20009-0. URL `http://dx.doi.org/10.1007/978-3-642-20009-0_38`. Also VSP WP 10-06; see www.vsp.tu-berlin.de/publications. (Cited on pages 27, 57, 58, 106 and 108.)

B. Dunham. Design by natural selection. *Synthese*, 15:254–258, 1963. (Cited on page 32.)

H. Ehtamo, S. Heliövaara, T. Korhonen, and S. Hostikka. Game theoretic best-response dynamics for evacuees' exit selection. *Adavances in Complex Systems*, 13(1):113–134, 2010. (Cited on page 11.)

I.J. Farkas. `pedsim` source code, accessed 2008. See pedsim.elte.hu. (Cited on page 14.)

G. Flötteröd. *Traffic state estimation with multi-agent simulations*. PhD thesis, Berlin Institute of Technology, 2008. URL `http://opus.kobv.de/tuberlin/volltexte/2008/1980/pdf/floetteroed_gunnar.pdf`. (Cited on page 12.)

G. Flötteröd and G. Lämmel. Evacuation simulation with limited capacity sinks. In *Proceedings of the International Conference on Evolutionary Computation*, 2010. (Cited on pages 31, 90 and 111.)

L.R. Ford and D.R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962. (Cited on pages 11, 18, 23, 30, 91 and 106.)

M.L. Fredman and R.E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34(3):596–615, 1987. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/28869.28874. (Cited on page 18.)

D. Fuldenberg and D.K. Levine. *The Theory of Learning in Games*. The MIT Press, 1998. (Cited on page 21.)

E. R. Galea, editor. *Pedestrian and Evacuation Dynamics*. Proceedings of the 2nd international conference, London, 2003. CMS Press, University of Greenwich, UK, 2003. (Cited on page 10.)

E.R. Galea. Simulating evacuation and circulation in planes, trains, buildings and ships using the EXODUS software. In M. Schreckenberg and S. D. Sharma, editors, *Pedestrian and Evacation Dynamics*, Proceedings of the 1st international conference, Duisburg, 2001, pages 203–225. Springer, 2002. (Cited on page 10.)

M. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979. ISBN 0716710455. (Cited on page 31.)

P. Gattermann, N. Waldau, and M. Schreckenberg, editors. *Pedestrian and Evacuation Dynamics*. Proceedings of the 3rd international conference, Vienna. Springer, Berlin, 2006. (Cited on page 10.)

C. Gawron. An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model. *International Journal of Modern Physics C*, 9(3):393–407, 1998. (Cited on pages 21 and 45.)

B. George, S. Kim, and S. Shekhar. Spatio-temporal network databases and routing algotihms: A summary of results. In *Proceedings of International Symposium on Spatial and Temporal Databases (SSTD'07)*, 2007. (Cited on page 27.)

C. Gloor, P. Stucki, and K. Nagel. Hybrid techniques for pedestrian simulations. In P. Sloot, B. Chopard, and A. Hoekstra, editors, *Cellular automata, Proceedings*, number 3305 in Lecture Notes in Computer Science, pages 581–590. Springer, 2004a. URL `http://www.springerlink.com/content/ng75m5lyyd8b617p/`. (Cited on pages 11 and 16.)

C. Gloor, P. Stucki, and K. Nagel. Hybrid techniques for pedestrian simulations. In *Proceedings of the Swiss Transport Research Conference (STRC)*, Monte Verita, CH, 2004b. See www.strc.ch. (Cited on page 15.)

N. Goseberg and T. Schlurmann. Enhanced hazard mapping on a medium-resolved numerical grid for the city of padang, west sumatra. *Journal of Ship Technology*, 5(2):13–21, July 2009. (Cited on pages 69, 81 and 96.)

N. Goseberg, A. Stahlmann, S. Schimmels, and T. Schlurmann. Highly-resolved numerical modeling of tsunami run-up and inundation scenarios in the city of Padang, West Sumatra. In F. Lehfeldt and H. Schüttrumpf, editors, *Poster proceedings of the International Conference on Coastal engineering*, pages 27–40, 2009. (Cited on pages 69, 81 and 96.)

D. Grether, B. Kickhöfer, and K. Nagel. Policy evaluation in multi-agent transport simulations. *Transportation Research Record: Journal of the Transportation Research Board*, 2175:10–18, 2010. ISSN 0361-1981. doi: 10.3141/2175-02. (Cited on page 48.)

S.J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey. Clearpath: highly parallel collision avoidance for multi-agent simulation. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 177–187, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-610-6. doi: http://doi.acm.org/10.1145/1599470.1599494. URL `http://doi.acm.org/10.1145/1599470.1599494`. (Cited on page 15.)

H.W. Hamacher and S.A. Tjandra. Mathematical modelling of evacuation problems: A state of art. Technical report, Fraunhofer ITWM, 2001. (Cited on page 11.)

H.W. Hamacher and S. Tufekci. On the use of lexicographical min cost flows in evacation modeling. *Naval Research Logistics*, 34:487–503, 1987. (Cited on pages 30 and 110.)

A.F. Han. TEVACS: Decision support system for evacuation planning in Taiwan. *Journal of Transportation Engineering*, 116(6):821–830, 1990. (Cited on page 10.)

L.D. Han and F. Yuan. Evacuation modeling and operations using dynamic traffic assignment and most desirable destination approaches. Transportation Research Board Annual Meeting Preprint 05-2401, Washington D.C., 2005. (Cited on page 9.)

S. Han and B.G. Heydecker. Consistent objectives and solutions of dynamic user equilibrium models. *Transpn. Res. B*, 40:16–34, 2005. (Cited on page 21.)

M. Hauskrecht and T. Singlair. Monte carlo optimizations for resource allocation problems in stochastic network systems. In *Proceedings of the Nineteenth International Conference on Uncertainty in Artificial Intelligence*, pages 305–312, 2003. (Cited on pages 31 and 111.)

D. Helbing and P. Molnár. Social force model for pedestrian dynamics. *PRE*, 51(5):4282–4286, May 1995. doi: 10.1103/PhysRevE.51.4282. (Cited on page 12.)

D. Helbing, I. Farkas, and T. Vicsek. Simulating dynamical features of escape panic. *Nature*, 407:487–490, 2000. (Cited on page 12.)

D. Helbing, I.J. Farkas, P. Molnar, and T. Vicsek. Simulation of pedestrian crowds in normal and evacuation situations. In M. Schreckenberg and S. D. Sharma, editors, *Pedestrian and Evacation Dynamics*, Proceedings of the 1st international conference, Duisburg, 2001, pages 21–58. Springer, 2002. (Cited on page 10.)

D. Helbing, L. Buzna, A. Johansson, and T. Werner. Self-organized pedestrian crowd dynamics: experiments, simulations and design solutions. *Transportation Science*, 39:1–24, 2005. doi: 10.1287/trsc.1040.0108. (Cited on page 10.)

A.G. Hobeika and C. Kim. Comparison of traffic assignments in evacuation modeling. *IEEE Transactions on Engineering Management*, 45(2):192–198, 1998. (Cited on page 9.)

A.G. Hobeika, S. Kim, and R.E. Beckwith. A decision support system for developing evacuation plans around nuclear power stations. *Interfaces*, 24 (5):22–35, 1994. (Cited on page 14.)

E. Hohnecker, A. Buchmann, F. Wenzel, T. Titzschkau, G. Bonn, D. Hilbring, and F. Quante. Frühwarnsysteme für schienenverkehrswege ews transport. In *GEOTECHNOLOGIEN Science Report No. 13: "Early Warning Systems in Earth Management"*, 2010. in German. (Cited on page 2.)

S.P. Hoogendoorn, P.H.L Bovy, and W. Daamen. Microscopic pedestrian wayfinding and dynamic modelling. In M. Schreckenberg and S. D. Sharma, editors, *Pedestrian and Evacation Dynamics*, Proceedings of the 1st international conference, Duisburg, 2001, pages 123–154. Springer, 2002. (Cited on page 11.)

M. Hoppe and H.S. Marhadiko. 30 Minutes in the City of Padang: Lessons for Tsunami Preparedness and Early Warning from the Earthquake on September 30, 2009. Technical report, GTZ, 2010. (Cited on page 55.)

M.W. Hoppe. Early warning experiences in Padang after first Bengkulu earthquake on 12 september 2007. Technical report, GTZ-International Services, December 2007. (Cited on page 55.)

T. Ishida, Y. Nakajima, Y. Murakami, and H. Nakanishi. Augmented experiment: Participatory design empowered by multiagent simulation. In *International Joint Conference on Artificial Intelligence (IJCAI-07)*, 2007. (Cited on page 14.)

M. Jafari, I. Bakhadyrov, and A. Maher. Technological advances in evacuation planning and emergency management: Current state of the art. Final Research Reports EVAC-RU4474, Center for Advanced Infrastructure and Transportation (CAIT), Rutgers University, NJ, 2003. URL `cait.rutgers.edu/finalreports/EVAC-RU4474.pdf`. (Cited on page 10.)

B.N. Janson. Dynamic traffic assignment for urban road networks. *Transpn. Res. B*, 25B(2/3):143–161, 1991. (Cited on page 20.)

J. Jarvis and H. Ratliff. Some equivalent objectives for dynamic network flow problems. *Management Science*, 28:106–108, 1982. (Cited on pages 11 and 23.)

M. Jha, K. Moore, and B. Pashaie. Emergency evacuation planning with microscopic traffic simulation. Transportation Research Board Annual Meeting Preprint 04-2414, Washington D.C., 2004. (Cited on page 9.)

D.E. Kaufman and R.L. Smith. Fastest paths in time-dependent networks for intelligent vehicle-highway systems application. *Journal of Intelligent Transportation Systems*, 1:1–11, 1993. (Cited on page 27.)

S. Kirkpatrick. Optimization by Simulated Annealing: Quantitative Studies. *Journal of Statistical Physics*, 34(5/6):975–986, 1984. (Cited on page 33.)

S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, 1983. (Cited on page 33.)

W.W.F. Klingsch, C. Rogsch, A. Schadschneider, and M. Schreckenberg, editors. *Pedestrian and Evacuation Dynamics 2008*. Springer, Berlin Heidelberg, 2010. (Cited on page 10.)

F. Klügl. Optimierung von Zuordnungsproblemen durch Simuliertes Ausglühen und Genetische Algorithmen. In *3. German Expert System Conference*, 1995. (Cited on page 33.)

F. Klügl and G. Rindsfüser. Large-scale agent-based pedestrian simulation. In *MATES '07: Proceedings of the 5th German conference on Multiagent System Technologies*, volume 4687 of *LNAI*, pages 145–156, Berlin, Heidelberg, 2007. Springer. ISBN 978-3-540-74948-6. doi: http://dx.doi.org/10.1007/978-3-540-74949-3_13. (Cited on page 13.)

F. Klügl, G. Klubertanz, and G. Rindsfüser. Agent-Based Pedestrian Simulation of Train Evacation Integrating Environmental Data. volume 5803 of *LNCS (LNAI)*, pages 631–638, Berlin Heidelberg, 2009. Springer. (Cited on page 13.)

H. Klüpfel. The simulation of crowd dynamics at very large events – Calibration, empirical data, and validation. In P. Gattermann, N. Waldau, and M. Schreckenberg, editors, *Pedestrian and Evacuation Dynamics*, Proceedings of the 3rd international conference, Vienna, pages 285–296. Springer, Berlin, 2006. (Cited on page 10.)

H. Klüpfel, T. Meyer-König, A. Keßel, and M. Schreckenberg. Simulating evacuation processes and comparison to empirical results. In M. Fukui et al, editor, *Traffic and granular flow '01*, pages 449–454. Springer, Berlin Heidelberg New York, 2003. (Cited on page 10.)

E. Köhler, K. Langtau, and M. Skutella. Time-expanded graphs for flow-dependent transit times. In *10th Annual European Symposium on Algorithms*, pages 599–611, 2002. (Cited on page 27.)

E. Köhler, R.H. Möhring, and M. Skutella. Algorithmics of large and complex networks. chapter Traffic Networks and Flows over Time, pages 166–196. Springer-Verlag, Berlin, Heidelberg, 2009. ISBN 978-3-642-02093-3. doi: http://dx.doi.org/10.1007/978-3-642-02094-0_9. URL `http://dx.doi.org/10.1007/978-3-642-02094-0_9`. (Cited on pages 21, 57, 104 and 106.)

E. Kuligowski. Review of 28 egress models. Technical report, National Institute of Standards and Technology (NIST), Gaithersburg, MD, 2004. URL `www.nist.gov`. (Cited on page 10.)

E. Kwon and S. Pitt. Evaluation of emergency evacuation strategies for downtown event traffic using a dynamic network model. Transportation Research Board Annual Meeting Preprint 05-2164, Washington D.C., 2005. (Cited on page 9.)

G. Lämmel and G. Flötteröd. Towards system optimum: Finding optimal routing strategies in time-tependent networks for large-scale evacuation problems. volume 5803 of *LNCS (LNAI)*, pages 532–539, Berlin Heidelberg, 2009. Springer. doi: 10.1007/978-3-642-04617-9_67. (Cited on pages 24, 25 and 58.)

G. Lämmel, H. Klüpfel, and K. Nagel. The MATSim network flow model for traffic simulation adapted to large-scale emergency egress and an application to the evacuation of the Indonesian city of Padang in case of a tsunami warning. In H. Timmermans, editor, *Pedestrian Behavior*, chapter 11, pages 245–265. Emerald Group Publishing Limited, 2009. (Cited on pages 40 and 113.)

G. Lämmel, D. Grether, and K. Nagel. The representation and implementation of time-dependent inundation in large-scale microscopic evacuation simulations. *Transportation Research Part C: Emerging Technologies*, 18 (1):84–98, February 2010a. ISSN 0968-090X. doi: DOI:10.1016/j.trc.2009.04.020. URL `http://www.science-direct.com/science/article/B6VGJ-4WH6KH1-1/2/9603aa762051292d295bc94d7e8b13b5`. (Cited on page 68.)

G. Lämmel, M. Rieser, K. Nagel, H. Taubenböck, G. Strunz, N. Goseberg, T. Schlurmann, H. Klüpfel, N. Setiadi, and J. Birkmann. Emergency preparedness in the case of a tsunami – evacuation analysis and traffic optimization for the Indonesian city of Padang. In W.W.F. Klingsch, C. Rogsch, A. Schadschneider, and M. Schreckenberg, editors, *Pedestrian and Evacuation Dynamics*, pages 159–170, Berlin Heidelberg, 2010b. Springer. (Cited on page 40.)

G. Lämmel, H. Klüpfel, and K. Nagel. Risk minimizing evacuation strategies under uncertainty. In *Pedestrian and Evacuation Dynamics 2010*, in press. URL `https://svn.vsp.tu-berlin.de/repos/public-svn/publications/vspwp/2010/10-01`. (Cited on pages 28 and 78.)

L.W. Lan and C.-W. Chang. Inhomogeneous cellular automata modeling for mixed traffic with cars and motorcycles. *Journal of Advanced Transportation*, 39(3):323–349, 2005. (Cited on page 10.)

N. Lefebvre and M. Balmer. Fast shortest path computation in time-dependent traffic networks. In *Proceedings of the Swiss Transport Research Conference (STRC)*, Monte Verita, CH, September 2007. See www.strc.ch. (Cited on page 18.)

C. Lieberman, D. Kurowski, M.R. Avila, L. Ricci, N. Thomas, C. Collyer, and B. Aguirre. Conceptual framework for simulating the pedestrian evacuation behavior from buildings. Transportation Research Board Annual Meeting Preprint 05-2297, Washington D.C., 2005. (Cited on page 12.)

E. Lim and B. Wolshon. Modeling and performance assessment of contraflow evacation termination points. *Transportation Research Record*, 1922:118–128, 2005. (Cited on page 10.)

W.-H. Lin and H.K. Lo. Are the objective and solutions of dynamic user-equilibrium models always consistent? *Transpn. Res. A*, 34:137–144, 2000. (Cited on page 20.)

H.X. Liu, J.X. Ban, Ma. Wenteng, and P.B. Mirchandani. Model reference adaptive control framework for real-time traffic management under emergency evacuation. *Journal of Urban Planning and Development*, 133(1): 43–50, 2007. (Cited on page 12.)

S.M. Lo, H.C. Hunag, Wang P., and K.K. Yuen. A game theory based exit selection model for evacuation. *Fire Safety Journal*, 41(5):364–369, 2006. (Cited on page 11.)

S. Lozano, F. Guerrero, L. Onieva, and J. Larrañeta. Kohonen maps for solving a class of location-allocation problems. *European Journal of Operational Research*, 108:106–117, 1998. (Cited on pages 31 and 111.)

Q. Lu, B. George, and S. Shekhar. Capacity constrained routing algorithms for evacuation planning: A summary of results. *LNCS*, 3633:291–307, 2005. (Cited on pages 18, 24, 26 and 91.)

L. Marshall, R. Fisher, R. Jaikumar, and L.N. Van Wassenhove. A multiplier adjustment method for the generalized assignment problem. *Management Science*, 32(9):1095–1103, 1986. (Cited on page 31.)

J. McCloskey, D. Lange, F. Tilmann, S.S. Nalbant, A.F. Bell, D.H. Natawidjaja, and A. Rietbrock. The september 2009 padang earthquake. *Nature Geoscience*, 3:70–72, 2010. (Cited on page 5.)

D.K. Merchant and G.L:; Nemhauser. A model and an algorithm for the dynamic traffic assignment problem. *Transportation Science*, 12:183–199, 1978. (Cited on page 24.)

B. Mertsching, M. Hund, and Z. Aziz, editors. *KI 2009: Advances in Artificial Intelligence*, volume 5803 of *LNCS (LNAI)*. Springer, Berlin Heidelberg, 2009. (Not cited.)

N. Metropolis, A. Rosembluth, M. Rosembluth, and A. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953. (Cited on pages 22, 32 and 51.)

C. Meyers. *Network Flow Problems and Congestion Games: Complexity and Approximation Results.* PhD thesis, Massachusetts Institute of Technology, 2006. (Cited on page 20.)

S. Meyers. *Effective STL: 50 specific ways to improve the use of the Standard Template Library.* Addison-Wesley, 2002. (Cited on page 122.)

Y. Murakami, T. Ishida, T. Kawasoe, and R. Hishiyama. Scenario description for multi-agent simulation. In *Autonomous agents and multiagent systems (AAMAS'03)*, Melbourne, Australia, July 2003. URL `www.aamas-conference.org`. (Cited on pages 12 and 113.)

J. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951. (Cited on pages 11, 20 and 106.)

D.H. Natawidjaja, K.Sieh, M. Chlieh, J. Galetzka, B.W. Suwargadi, H. Cheng, R.L. Edwards, J-P. Avouac, and S.N. Ward. Source parameters of the great Sumatran megathrust earthquakes of 1797 and 1833 inferred from coral microatolls. *Journal of Geophysical Research*, 111, 2006. (Cited on page 5.)

K. Nishinari, A. Kirchner, A. Nazami, and A. Schadschneider. Extended floor field CA model for evacuation dynamics. *IEICE Transactions on Information and Systems*, E87-D(3):726–732, 2004. (Cited on page 11.)

R. Oleson, D.J. Kaup, T.L. Clark, L.C. Malone, and L. Boloni. Social potential models for traffic and transportation. In A.L.C. Bazzan and F. Klügl, editors, *Multi-agent systems for traffic and transportation engineering*, chapter VII, pages 155–175. Information Science Reference (IGI Global), 2008. (Cited on page 12.)

J. de D. Ortúzar and L.G. Willumsen. *Modelling transport.* John Wiley Sons Ltd, Chichester, 3 edition, 2001. (Cited on page 9.)

I.H. Osman. Heuristics for the generalised assignment problem: simulated annealing and tabu search approaches. *OR Spektrum*, 17:211–225, 1995. (Cited on page 33.)

S. Pallottino and M.G. Scutella. Shortest path algorithms in transportation models: Classical and innovative aspects. In P. Marcotte and S. Nguyen, editors, *Equillibrium and Adavanced Transportation Modelling*, pages 245–281. Kluwer, 1998. (Cited on page 27.)

X. Pan, C.S. Han, K. Dauber, and K.H. Law. A multi-agent based framework for the simulation of human and social behavior during emergency

evacuations. *AI and Society*, 22(2):113–132, 2007. ISSN 0951-5666. doi: http://dx.doi.org/10.1007/s00146-007-0126-1. (Cited on page 13.)

R. Peacock, E. Kuligowski, and J. Averill, editors. *Pedestrian and Evacuation Dynamics 2010*. Springer, Berlin, 2011. doi: 10.1007/978-1-4419-9725-8_26. (Cited on page 10.)

S. Peeta and H.S. Mahmassani. System optimal and user equilibrium time-dependent traffic assignment in congested networks. *Annals of Operations Research*, 60:81–113, 1995. (Cited on page 24.)

W. Predtetschenski and A. Milinski. *Planning for Foot Traffic in Buildings*. Amerind Publishing Co. Pvt. Ltd., New Delhi, 1978. (Cited on pages 10 and 48.)

F.P. Preparata. The medial axis of a simple polygon. In *Mathematical Foundations of Computer Science 1977*, volume 53 of *LNCS*, pages 443–450, Berlin Heidelberg, 1977. Springer. (Cited on page 47.)

B. Raney and K. Nagel. Iterative route planning for large-scale modular transportation simulations. *Future Generation Computer Systems*, 20(7): 1101–1118, 2004. (Cited on page 21.)

A.S. Rao and M.P. Georgeff. Modeling rational agents within a BDI-architecture. In James Allen, Richard Fikes, and Erik Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 473–484. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1991. URL `http://jmvidal.cse.sc.edu/library/rao91a.pdf`. (Cited on page 13.)

H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951. (Cited on page 49.)

C. Rogsch. Vergleichende Untersuchungen zur Simulation von Personenströmen. Diploma thesis, Universität Wuppertal, 2005. (Cited on page 10.)

S.J. Russel and P. Norvig. *Artificial intelligence: A modern approach*. Series in Artificial Intelligence. Prentice Hall, 1995. (Cited on pages 18 and 32.)

P. Sattayhatewa and B. Ran. Developing a dynamic traffic management model for nuclear power plant evacuation. Transportation Research Board Annual Meeting Preprint 00-0207, Washington D.C., 2000. (Cited on page 10.)

H. Sbayti, C-C. Lu, and H. Mahmassani. Efficient implementation of method of successive averages in simulation-based dynamic traffic assignment models for large-scale network applications. *Transportation Research Record*, (2029):22–30, 2007. (Cited on page 14.)

Q. Schiermerier. Tsunami watch. *Nature News*, 426(7276):968–969, 2009. (Cited on page 3.)

V. Schneider and R. Könnecke. Simulating evacuation processes with ASERI. In M. Schreckenberg and S. D. Sharma, editors, *Pedestrian and Evacation Dynamics*, Proceedings of the 1st international conference, Duisburg, 2001, pages 303–314. Springer, 2002. (Cited on page 10.)

M. Schreckenberg and S. D. Sharma, editors. *Pedestrian and Evacation Dynamics*. Proceedings of the 1st international conference, Duisburg, 2001. Springer, 2002. (Cited on page 10.)

N. Setiadi, H. Taubenböck, S. Raupp, and J. Birkmann. Integrating Socio-Economic Data in Spatial Analysis: An Exposure Analysis Method for Planning Urban Risk Mitigation. In M. Schrenk, V. V. Popopvich, D. Engelke, and P. Elisei, editors, *REAL CORP 2010 Proceedings*, 2010. (Cited on pages 54 and 55.)

Y. Sheffi. *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1985. (Cited on page 9.)

P.M. Simon and K. Nagel. Simple queueing model applied to the city of Portland. Transportation Research Board Annual Meeting Preprint 99-0861, Washington D.C., 1999. (Cited on page 48.)

M. Skutella. An introduction to network flows over time. In W. Cook, L. Lovász, and J. Vygen, editors, *Research Trends in Combinatorial Optimization*, pages 451–842. Springer, 2009. (Cited on page 23.)

R. Standish. Classdesc Library, accessed 2005. See parallel.hpc.unsw.edu.au/rks/classdesc. (Cited on page 15.)

D. Strippgen. *Investigating the Technical Possibilities of Real-time Interaction with Simulations of Mobile Intelligent Particles*. PhD thesis, TU Berlin, 2009. URL `http://opus.kobv.de/tuberlin//frontdoor.php?source_opus=2364`. (Cited on pages 62 and 114.)

D. Strippgen and K. Nagel. Using common graphics hardware for multi-agent simulation with CUDA. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, Rome, 2009. (Cited on page 15.)

H. Taubenböck and A. Roth. A transferable and stable classification approach in various urban areas and various high resolution sensors. In *Urban Remote Sensing Joint Event*, Paris, 2007. (Cited on page 52.)

H. Taubenböck, N. Goseberg, N. Setiadi, G. Lämmel, F. Moder, M. Oczipka, H. Klüpfel, R. Wahl, T. Schlurmann, G. Strunz, J. Birkmann, K. Nagel, F. Siegert, F. Lehmann, S. Dech, A. Gress, and R. Klein. "Last-Mile" preparation for a potential disaster – Interdisciplinary approach towards tsunami early warning and an evacuation information system for the coastal city of Padang, Indonesia. *Natural Hazards and Earth System Science*, 9(4):1509–1528, 2009a. ISSN 1561-8633. URL `http://www.nat-hazards-earth-syst-sci.net/9/1509/2009/`. (Cited on page 4.)

H. Taubenböck, N. Goseberg, N. Setiadi, G. Lämmel, F. Moder, T. Schlurmann, M. Oczipka, H. Klüpfel, G. Strunz, J. Birkmann, K. Nagel, F. Siegert, F. Lehmann, S. Dech, A. Gress, and R. Klein. Last-Mile – numerical last-mile Tsunami early warning and evacation information system. In *Geotechnologien Science Report No. 13: "Early Warning Systems in Earth Management"*, 2009b. (Cited on page 94.)

H. Taubenböck, N. Goseberg, G. Lämmel, N. Setiadi, T. Schlurmann, K. Nagel, F. Siegert, J. Birkmann, K.-P. Traub, S. Dech, V. Keuck, F. Lehmann, G. Strunz, and H. Klüpfel. Risk mitigation at the "Last-Mile" – from science to action by the example of Padang, Indonesia. submitted to NHESS, 2010. URL `https://svn.vsp.tu-berlin.de/repos/shared-svn/publications/vspwp/2010/10-16/`. (Cited on page 3.)

G. Theodoulou and B. Wolshon. Alternative methods to increase the effectiveness of freeway contraflow evacation. *Transportation Research Record*, 1865:48–56, 2004. (Cited on page 10.)

V. Černý. Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithmen. *Journal of Optimization Theory and Applications*, 45(1), 1985. (Cited on pages 33 and 34.)

P. Vortisch. *Model-Supported Propagation of Measured Values for Real-Time Traffic State Estimation in Urban Road Networks.* PhD thesis, University of Karlsruhe, Germany, 2005. URL `www.ptvamerica.com/docs/ModelSupportedProp.pdf`. (Cited on page 12.)

J. L. Walsh. A closed set of normal orthogonal functions. *American Journal of Mathematics*, 45(1):5–24, 1923. ISSN 00029327. (Cited on page 114.)

L. Wang, B. Mao, S. Chen, and K. Zhang. Mixed flow simulation at urban intersections: Computational comparisons between conflict-point detection and cellular automata models. In *International Joint Conference on Computational Sciences and Optimization*, 2009. (Cited on page 10.)

J.G. Wardrop. Some theoretical aspects of road traffic research. *Proceedings of the Institute of Civil Engineers*, 1:325–378, 1952. (Cited on pages 20 and 23.)

U. Weidmann. *Transporttechnik der Fussgänger*, volume 90 of *Schriftenreihe des IVT*. Institute for Transport Planning and Systems ETH Zürich, 2 edition, 1993. In German. (Cited on pages 10, 47 and 94.)

Y. Wen, R. Balakrishna, M. Ben-Akiva, and S. Smith. Online deployment of dynamic traffic assignment: architecture and run-time management. In *Intelligent Transport Systems, IEE Proceedings*, volume 153, pages 76–84, 2006. (Cited on page 15.)

P.D. Whiting and J.A. Hillier. A method for finding the shortest route through a road network. *Operational Research Quarterly*, 11(1):37–40, 1960. (Cited on page 18.)

M.R. Wilhelm and T.L. Ward. Solving quadratic assignment problems by 'simulated annealing'. *IIE Transactions*, 19:107–119, 1987. (Cited on page 33.)

K. Zhang, H.S. Mahmassani, and C.-C. Lu. Probit-based time-dependent stochastic user equilibrium traffic assignment model. *Transportation Research Record*, 2085:86–94, 2008. (Cited on page 9.)