Modellierung und Evaluation von Lichtsignalanlagen in Queue-Simulationen

Berlin, den 29. September 2008

Diplomarbeit im Fach Modellierung und Simulation von Verkehr

Eingereicht von:	Gutachter:
Andreas Neumann	Prof. Dr. Kai Nagel, Dipl.-Inf. Dominik Grether
	Institut für Land- und Seeverkehr, Technische Universität Berlin,
	Fachgebiet für Verkehrssystemplanung und Verkehrstelematik

Zusammenfassung Lichtsignalanlagen sind ein Instrument der Verkehrsplanung und beschränken sich nicht nur auf die kurzfristige Verkehrslenkung. Sollen die Planungen vorab simuliert werden, bedarf es eines Modells, welches Lichtsignalanlagen berücksichtigt. Dazu wird das innerhalb der Multi-Agenten-Simulation MATSim Verwendung findende Warteschlangenmodell in dieser Arbeit mit dem Ziel weiterentwickelt, Lichtsignalanlagen und ihre Auswirkungen auf den Verkehrsfluss abbilden zu können.

Das Modell dieser Arbeit ist in der Lage, die für die Kapazität eines Knotenpunktes entscheidenden Eigenschaften wie Anzahl der zur Verfügung stehenden Fahrstreifen, deren Abbiegebeziehungen und Länge abzubilden. Es ist möglich, an einem Knotenarm für jede Abbiegebeziehung getrennte Fahrstreifen einzurichten, mehrere Abbiegebeziehungen auf einem gemischten Fahrstreifen zusammenzufassen oder beides zu kombinieren. Zusätzlich kann die Signalisierung unabhängig von der Aufteilung der Fahrstreifen erfolgen. So können mehrere Fahrstreifen von einer Signalgruppe gemeinsam signalisiert, einzelne Abbiegebeziehungen eines gemischten Fahrstreifens von separaten Signalgruppen signalisiert werden oder es wird auf eine Signalisierung der Spuraufteilung verzichtet. Rückstaus auf einem Fahrstreifen blockieren ungestaute Fahrstreifen und breiten sich auf den gesamten Knotenarm und stromaufwärts anschließende Kanten aus.

Die Arbeit bildet sowohl die Grundlage für die Erprobung von Signalzeitenplänen und deren Koordinierung, als auch für die Entwicklung von Strategien der Netzbeeinflussung sowie adaptiver Ampelsteuerungen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

FlowCapacity C . Maximale Anzahl von Fahrzeugen, die innerhalb eines bestimmten Zeitraums eine bestimmte Stelle der Kante passieren kann. Bestimmt damit die Anzahl der Fahrzeuge, die in einem Zeitschritt die Kante verlassen dürfen. [Fz/h]

FlowQueue ..... Teil einer Kante, welche die Fahrzeuge beim Verlassen der Kante zwischenspeichert. Begrenzt über deren Kapazität (*FlowCapacity*) den Fluss der Kante.

HBS ..... **H**andbuch für die **B**emessung von **S**traßenverkehrsanlagen

inLink ..... Auf den betreffenden Knoten zuführende Kante

Kante ..... Darstellung einer Straße im Graphen und Verbindung zweier Knoten

Kapazität ..... Mit der Kapazität ist hier immer die unter der *FlowCapacity C* beschriebene Kapazität gemeint. Vergleiche dazu *StorageCapacity*

Knoten ..... Darstellung eines Knotenpunktes im Graphen

l ..... Länge der Kante [m]

l_{Fz} ..... Länge eines Fahrzeuges [m]

LSA ..... **L**icht**s**signal**a**nlage

MATSim ..... **M**ulti-**A**gent **T**ransport **Sim**ulation

N_{GE} ..... Anzahl der gestauten Fahrzeuge bei Grünende [Fz]

n_H ..... Anzahl der haltenden Fahrzeuge pro Umlauf [Fz/Umlauf]

n_{Spuren} ..... Anzahl der Fahrstreifen einer Kante [ ]

outLink ..... Vom betreffenden Knoten wegführende Kante

ParkingQueue ... Teil der Kante, welcher Fahrzeuge zwischenspeichert, die ihre Abfahrtzeit noch nicht erreicht haben und z.B. einer Aktivität auf der Kante nachgehen.

q ..... Verkehrsstärke [Fz/h]

q_S ..... Sättigungsverkehrsstärke [Fz/h]

Replanning ..... Bestandteil einer MATSim-Iteration, bei der ein Teil der Agenten seine Pläne modifizieren darf

StorageCapacity . Anzahl der Fahrzeuge, die sich gleichzeitig auf einer Kante aufhalten dürfen.

StorageQueue .... Teil der Kante, welche die auf der Kante fahrenden Fahrzeuge speichert. Begrenzt über deren Kapazität die Zahl der auf der Kante gleichzeitig befindlichen Fahrzeuge.

t_B ..... Mittlerer Zeitbedarfswert; Gibt die Zeitdauer an, die zwischen dem Passieren der Haltelinie durch das erste Fahrzeug und dem Passieren der Haltelinie durch das nachfolgende Fahrzeug verstreicht. [s/Fz]

t_F ..... Freigabezeit oder auch Grünzeit; Zeitdauer in der eine Signalgruppe grün anzeigt. [s]

tt_0 ..... $FreeLinkTravelTime$ oder die für das vollständige Passieren einer Kante minimal benötigte Reisezeit, die von einem Fahrzeug ohne behindert zu werden realisiert werden kann. [s]

t_U ..... Umlaufzeit; Zeitdauer zwischen dem Erscheinen eines Signalzustandes und dem erneuten Erscheinen desselben Signalzustandes. [s]

v_0 ..... $FreeLinkSpeed$ oder maximale Geschwindigkeit, die von einem unbeeinträchtigt fahrenden Fahrzeug auf der Kante realisiert werden kann. [m/s]

1 Einleitung

Die Simulation von Verkehr spielt in der Planung und Bewertung von Verkehrsmaßnahmen eine zunehmend größere Rolle. Bei Einführung einer Maut für die Innenstadt, bei der Veränderung des Straßennetzes durch kurzfristige Sperrungen oder bei langfristigen baulichen Maßnahmen werden im Vorfeld Simulationen durchgeführt, um deren Auswirkungen abschätzen zu können. Ein aktuelles Beispiel ist der Bau einer neuen Umgehungsstraße im schweizerischen Kanton Zürich. Die Umgehungsstraße wird in Form einer Autobahn realisiert und westlich um die Stadt Zürich herumgeführt. Zusätzlich wurde eine Reihe von flankierenden Maßnahmen beschlossen. Zu den wichtigsten davon zählen der Rückbau von Straßen im Zentrum der Stadt, welche bisher hauptsächlich vom Transitverkehr benutzt werden, und der Bau neuer Lichtsignalanlagen an der Autobahnausfahrt Wollishofen (A3). Die LSA dienen als Pförtnerampeln und dosieren den in die Stadt fließenden Verkehr, so dass die Westumfahrung als Alternativroute an Attraktivität gewinnt und der Verkehr durch die Wohngebiete behindert und somit unattraktiver wird (siehe www.Westumfahrung.ch).

Wie das Beispiel zeigt, sind Lichtsignalanlagen ein Instrument der Verkehrsplanung und beschränken sich nicht nur auf die kurzfristige Verkehrslenkung. Sollen die Planungen vorab simuliert werden, bedarf es eines Modells, welches Lichtsignalanlagen berücksichtigt. Da Lichtsignalanlagen zeitabhängig steuern, ist die Zeitabhängigkeit eine Anforderung an das Modell. Im Falle der Westumfahrung besteht zusätzlich die Anforderung, dass die in die Stadt ein- und auspendelnden Verkehrsströme abgebildet werden können. Das Modell muss also in der Lage sein nicht nur die Stadt Zürich, sondern den gesamten Kanton Zürich zu simulieren.

Ein Modell, welches solche Großräume abdecken kann ist das sogenannte Warteschlangenmodell (englisch: queue simulation). Es ist vom Ansatz her eines der schnellsten und ressourcensparendsten Modelle zur Simulation von Verkehr und wird unter anderem am Fachgebiet für Verkehrssystemplanung und Verkehrstelematik der Technischen Universität Berlin eingesetzt, um Regionen wie Berlin-Brandenburg oder die Schweiz zu simulieren. Deren bisherige Implementierung ist jedoch nicht in der Lage, die genaue Schaltung einer Lichtsignalanlage zu berücksichtigen. Häufig zeigt sich jedoch, dass die Koordinierung mit benachbarten Knotenpunkten für die Kapazität eines Knotenpunktes wichtiger ist als die Kapazität des Knotenpunktes selbst. Zum Beispiel findet die bisherige Implementierung bei nicht ausgelasteten Netzen bevorzugt Ausweichrouten in der Innenstadt, anstatt eine unsignalisierte oder koordinierte Umgehungsstraße oder gar

eine Stadtautobahn zu empfehlen. Dieses Verhalten ist eine direkte Folge des Fehlens der von den LSA verursachten zusätzlichen Wartezeiten.

Die bisherige Implementierung berücksichtigt Lichtsignalanlagen nur insofern, als dass durch das Absenken der Kapazität einzelner Strecken ein durch die LSA verursachter Kapazitätseinbruch lediglich imitiert werden kann. Eine solche Absenkung betrifft alle Fahrzeuge, die den Streckenabschnitt passieren, und ist unabhängig von der Richtung, in die abgebogen wird. Alle Abbiegebeziehungen werden damit mit der gleichen Schaltung versehen. Eine genaue Differenzierung und Abbildung der einzelnen Abbiegebeziehungen findet nicht statt.

Der Schwerpunkt der Arbeit liegt in der Weiterentwicklung des Warteschlangenmodells, welches erstens die Modellierung von Lichtsignalanlagen zulässt und zweitens den Einfluss signalisierter Knotenpunkte auf den Verkehrsfluss darzustellen vermag. Das Ziel ist eine ausreichend realistische Abbildung des Verkehrsflusses, da dieser sich wiederum in Verhaltensänderungen der Verkehrsteilnehmer widerspiegelt. Eine Verhaltensänderung seitens der Verkehrsteilnehmer kann das Ergebnis der Simulation entscheidend beeinflussen und Netzwerkeffekte verursachen. Beispielsweise kann die Veränderung des Verhaltens eines Agenten den Verkehrsfluss an einer anderen Stelle im Netz beeinflussen. Das Modell soll auf dem bestehenden Warteschlangenmodell aufbauen und außerhalb der signalisierten Knotenpunkte das gleiche Verkehrsverhalten aufweisen. Insbesondere unterbleibt eine genauere Abbildung des Verkehrsflusses wie zum Beispiel in Form von Spurwechseln, Überholvorgängen oder einem Fahrzeugfolgemodell.

Diese Arbeit entwickelt kein Verkehrsflussmodell auf dem Knotenpunkt selbst. Dort auftretende Konflikte bleiben wie in der bisherigen Implementierung ungelöst. Es wird davon ausgegangen, dass Signalzeitenpläne mit Festzeitsteuerung für die signalisierten Knotenpunkte vollständig vorliegen und diese mögliche Konflikte berücksichtigen.

Auf der anderen Seite muss das neue Modell in der Lage sein, Auswirkungen einzelner Signalgeber einer LSA auf den restlichen Verkehrsfluss abbilden zu können. Dazu sind eine gesonderte Betrachtung der Schaltung einzelner Signalgruppen und eine Einteilung in für die Verkehrsströme eines Knotenpunktes relevante Fahrstreifen nötig. So soll sich ein Rückstau einer Abbiegespur, verursacht z.B. durch eine unzureichende Grünzeit, auf andere Abbiegespuren ausbreiten und dort den Verkehrsfluss beeinträchtigen.

Als Entwicklungsumgebung wird dabei exemplarisch auf die am Fachgebiet Verwendung findende „Multi-Agent Transport Simulation“, kurz MATSim, zurückgegriffen. Dies ermöglicht deren Komponenten wie die gesamte Infrastruktur für die Vorbereitung, Durchführung und Auswertung der Simulation zu verwenden. Angepasst wird lediglich

die Implementierung des Verkehrsflussmodells, da hier der eigentliche Schwerpunkt der Arbeit liegt. Es ist verantwortlich für die physikalische Simulation auf dem Netzwerk, also den eigentlichen Verkehrsablauf.

Das anschließende Kapitel beginnt mit der Einführung in das verwendete Verkehrsflussmodell. Darauf folgt die Beschreibung der „Multi-Agent Transport Simulation“ und der von ihr benutzten Erweiterungen am Warteschlangenmodell. Die Bewertung des neu zu entwickelnden Modells ist Thema des letzten Abschnitts dieses Kapitels.

Im Kapitel 3 werden zuerst die an das neu zu entwickelnde Modell gestellten Anforderungen erläutert. Anschließend werden zwei verschiedene Ansätze für dessen Umsetzung vorgestellt und diskutiert. Einer der Ansätze wird anschließend detailliert ausgearbeitet und spezifiziert. Abschließend wird geprüft, inwiefern sich das neue Modell erweitern ließe.

Im darauf folgenden Kapitel wird beschrieben wie das bisherige Verkehrsflussmodell in MATSim implementiert ist. Das Kapitel 5 beinhaltet die durchgeführten Tests und die dabei mit der eigenen Implementierung erzielten Resultate. Die Arbeit schließt mit einer Zusammenfassung und möglichen Erweiterungen.

2 Grundlagen

Die in diesem Kapitel beschriebenen Grundlagen sollen zum einen das dieser Arbeit zu Grunde liegende Verkehrsflussmodell erläutern und zum anderen einen Einblick in die verwendete Entwicklungsumgebung geben. Abschließend erfolgt eine kurze Vorstellung der bei der ingenieurtechnischen Bewertung von Knotenpunkten verwendeten Methoden.

2.1 Verkehrsflussmodelle

In der Verkehrsplanung werden verschiedene Modelle verwendet um den Verkehrsfluss zu simulieren. Beispiele sind das Fahrzeugfolgemodell von Wiedemann (1974) oder Modelle basierend auf dem Zellularautomaten, wie ihn Nagel u. Schreckenberg (1992) vorgestellt haben. Bei letzterem werden die Strecken in einzelne diskrete räumliche Abschnitte unterteilt. Innerhalb eines Abschnittes kann sich im Normalfall nur ein Fahrzeug aufhalten. Sind zwischen zwei Knoten alle Abschnitte einer Kante belegt, so ist dieser Streckenabschnitt gestaut. Ein Fahrzeug kann sich während eines Zeitschrittes einen oder mehrere Abschnitte weit fortbewegen. Dabei werden Restriktionen hinsichtlich Geschwindigkeit der Fahrzeuge und deren Beschleunigung berücksichtigt. Je nach Komplexität des Modells können weitere Aspekte des Verkehrs modelliert werden. Dazu zählen Spurwechsel sowie Abbiegevorgänge, bis hin zur Simulation des Verkehrs auf einem Knotenpunkt.

Der Vorteil der zellulären Automaten liegt in der exakten Simulation des Verkehrsflusses auf einer Kante. Fahrzeuge lassen sich auf den Kanten lokalisieren und Kenngrößen des Verkehrs aggregieren. Daraus entsteht aber zugleich der größte Nachteil. Je komplexer das Modell ist, desto umfangreicher ist dessen Umsetzung und desto höher ist der Rechenaufwand.

Die Abbildung von detailliertem Fahrverhalten spielt in der Verkehrsplanung oft eine untergeordnete Rolle. Wichtiger ist, dass dessen Auswirkungen wie die Staubildung und deren Auflösung, sich im Netz ausbreitende Rückstaus und die Dynamik des Verkehrsaufkommens realitätsnah abgebildet werden können. An dieser Stelle soll ein minimalistisches Alternativmodell vorgestellt werden, welches diese Anforderungen erfüllt, das so genannte Warteschlangenmodell (englisch: queue model). Dieses wird innerhalb der in Abschnitt 2.2 beschriebenen Multi-Agenten-Simulation verwendet und stellt ein einfaches dynamisches Modell dar, welches für Verkehrssimulationen benutzt wird.

2.1.1 Warteschlangenmodelle

Warteschlangenmodelle gehen davon aus, dass ein Dienst wie z.B. eine Supermarktkasse eine bestimmte Anzahl an Kunden pro Stunde bedienen kann. Dies nennt man die Kapazität des Dienstes. Wollen mehr Kunden die Kasse in Anspruch nehmen als die Kapazität es zulässt, so stauen sich diese Kunden vor der Kasse und es entsteht eine Warteschlange (englisch: queue). Unter der Annahme eines unendlich großen Supermarktes, können sich bei unzureichender Kapazität auch unendlich lange Warteschlangen bilden.

Auf die Simulation von Verkehr übertragen, kann eine als Kante dargestellte Straße als Queue modelliert werden. Der an deren Ende befindliche Knotenpunkt kann eine begrenzte Anzahl an Fahrzeugen je Stunde von dieser Straße „bedienen“ und bestimmt somit die Kapazität der Kante. Bei Überschreitung der Kapazität beginnen sich die Fahrzeuge auf der Kante zu stauen.

Im Gegensatz zum Zellularautomaten werden die Kanten nicht in einzelne Abschnitte unterteilt. Das Netz wird durch einen Graphen repräsentiert, bestehend aus Knoten und Kanten. Die Knoten dienen lediglich als Verknüpfungspunkte für die Kanten und sind räumlich fixiert. Die Kanten dienen als Verbindung zwischen den Knoten. Jede Kante wird über einen Ausgangsknoten und einen Zielknoten definiert und ist damit ebenfalls räumlich fixiert.

Zur Speicherung der Fahrzeuge auf der Kante verwendet das Modell eine Queue. Fahrzeuge, welche die Kante befahren, werden in diese Queue gesetzt. Ohne weitere Restriktion gelangen sie nach dem Prinzip „First-In-First-Out“ (FIFO) augenblicklich am Ende der Kante an und können auf die nachfolgende Kante gesetzt werden.

Da ausschließlich beim Betreten und Verlassen Berechnungen anfallen, werden innerhalb der Kante keinerlei weitere Berechnungen angestellt. Daraus resultiert der größte Vorteil des Warteschlangenmodells in Form eines geringen Rechenaufwands. Der größte Nachteil ist das wenig realistische Verhalten der Fahrzeuge. Zum Beispiel durchfahren Fahrzeuge die komplette Kante innerhalb eines Zeitschrittes und eine Lokalisierung der einzelnen Fahrzeuge auf der Kante wie im Zellularautomaten ist nicht möglich.

Ein weiteres Problem ist die Auswirkung einer gestauten Kante auf benachbarte Kanten. Modelliert man ein gesamtes Straßennetz mit diesem Modell, so werden sich Rückstaus nicht im Netz ausbreiten, da jede Kante eine unbegrenzte Anzahl an Fahrzeugen aufnehmen kann. Dies ist offensichtlich unrealistisch. Eine Lösung ist die im Folgenden von Gawron (1998) vorgestellte Begrenzung der maximalen Fahrzeugzahl, welche sich auf einer Kante gleichzeitig aufhalten darf.

2.1.2 Warteschlangenmodell nach Gawron

Das bisher vorgestellte Warteschlangenmodell basiert ausschließlich auf der Restriktion in Form der *FlowCapacity*. Gawron erweitert es in seiner Arbeit (Gawron, 1998), indem er die Anzahl der Fahrzeuge auf einer Kante begrenzt und eine minimale Reisezeit für die Kante einführt. Für die Simulation von Straßenverkehr bedarf es in dem Modell mindestens dreier Kenngrößen, um den Verkehrsfluss realitätsnah abbilden zu können:

- *FreeLinkTravelTime*: Die für das vollständige Passieren einer Kante minimal benötigte Reisezeit, welche von einem Fahrzeug ohne Beeinträchtigungen realisiert werden kann.
- *FlowCapacity*: Die maximale Anzahl von Fahrzeugen, die innerhalb eines bestimmten Zeitraums eine Kante passieren kann.
- *StorageCapacity*: Die maximale Anzahl von Fahrzeugen, die sich bei Stillstand gleichzeitig auf der Kante aufhalten kann.

Voraussetzung ist, dass die folgenden Größen für alle Kanten des Netzes bekannt sind.

- Zulässige Höchstgeschwindigkeit der Kante v_0 [m/s], auch *FreeLinkSpeed*
- Länge der Kante l [m]
- *FlowCapacity* C [Fz/h]
- Anzahl der Fahrstreifen n_{Spuren} [ ]
- Fahrzeuglänge l_{Fz} [m]

Aus diesen Werten lassen sich die fehlenden beiden Größen *FreeLinkTravelTime* und *StorageCapacity* ableiten. Die *FreeLinkTravelTime* tt_0 kann aus *FreeLinkSpeed* und Länge l der Kante abgeleitet werden und ergibt sich zu

$$tt_0 = \frac{l}{v_0} \tag{1}$$

Die *StorageCapacity* berechnet sich zu

$$C_{max} = n_{Spuren} \cdot \frac{l}{l_{Fz}} \tag{2}$$

Ohne *StorageCapacity* könnten bei einer vorliegenden Belastung, die höher als die *FlowCapacity* ist, unendlich viele Fahrzeuge auf der Kante Platz finden. Man spricht dann von einer „point queue“. Realistischer ist, dass die Kante mit der Zeit zustaut und darüber hinaus beginnt, die stromaufwärts liegenden Kanten ebenfalls zu stauen. Dies wird in der englischen Literatur auch „spatial queue“ genannt.

Verhalten auf der Kante Durch die *FlowCapacity* werden die während eines Zeitschritts von einer Kante abfließenden Fahrzeuge begrenzt. Ein komplettes Abfließen aller Fahrzeuge einer Kante ist dann nicht mehr möglich, falls die Anzahl der abfließenden Fahrzeuge die *FlowCapacity* übersteigt.

Wird die Geschwindigkeit auf der Kante durch Angabe des *FreeLinkSpeed*s begrenzt, so beträgt die *FreeLinkTravelTime* nicht mehr null. Fahrzeuge werden also nicht mehr sofort nach Betreten einer Kante an dessen Ende gesetzt, sondern benötigen entsprechend Zeit um die Kante langsam zu „durchfahren“. Beim Betreten der Kante wird jedes Fahrzeug mit einem Zeitstempel versehen. Dieser besagt den Zeitpunkt an dem das Fahrzeug die Kante frühestens verlassen kann und errechnet sich aus dem Zeitpunkt des Betretens der Kante plus den Aufschlag der *FreeLinkTravelTime*. Da alle Fahrzeuge den gleichen Aufschlag bekommen, durchfahren alle Fahrzeuge die Kante mit der gleichen definierten maximal möglichen Geschwindigkeit. In Folge ist es nicht möglich, dass Fahrzeuge die gleiche Kante mit unterschiedlichen Geschwindigkeiten passieren und sich beispielsweise gegenseitig überholen. Auch lässt sich zähflüssiger Verkehr innerhalb einer Kante ohne weitere Restriktionen nicht simulieren.

Ein Einholen ist durch eine Stauung am Ende der Kante dennoch möglich. Können keine Fahrzeuge abfließen, da z.B. die folgende Kante bereits gestaut ist, so verbleiben sie trotz abgelaufener Zeitstempel auf der Kante. Mehrere Fahrzeuge mit abgelaufenem Zeitstempel können als sich am Ende der Kante stauend interpretiert werden. Ist ein Abfließen wieder möglich, können unter Berücksichtigung der *FlowCapacity* alle Fahrzeuge mit abgelaufenem Zeitstempel die Kante nach dem FIFO-Prinzip verlassen. Dies hat zur Folge, dass bei einer vormals komplett gestauten Kante mit verbleibender *StorageCapacity* gleich null, eine am Ende der Kante neu entstandene Fahrzeuglücke ohne Zeitverluste an den Anfang der Kante gereicht wird. Diese steht dann für den nachfolgenden Verkehr sofort zur Verfügung. Da dies unabhängig von der Länge der Kante geschieht, wandert solch eine Lücke innerhalb eines Simulationsschrittes durch beliebig lange Kanten und gegebenenfalls innerhalb weniger Zeitschritte über mehrere Kanten und Kilometer.

Verhalten am Knotenpunkt Die nach Simon u. a. (1999) einfachste Knotenpunktlogik besteht darin über alle Kanten eines Netzes zu iterieren und für jede Kante die folgenden drei Bedingungen zu prüfen:

1. Das erste Fahrzeug der Queue hat das Ende der Kante erreicht: Die zum Passieren einer Kante minimal benötigte Zeit beträgt tt_0. Daraus folgt, dass ein zum Zeitpunkt t_0 einfahrendes Fahrzeug, die Kante nicht vor dem Zeitpunkt $t_0 + tt_0$ wieder verlassen kann.

2. Die *FlowCapacity* C der Kante wird nicht überschritten: Beim Abfluss der Fahrzeuge am Ende der Kante wird geprüft, ob die maximale Anzahl der Fahrzeuge für diesen Zeitschritt nicht schon überschritten wurde. Formal geschieht das durch Prüfen der folgenden beiden Bedingungen:

$$N < int(C) \tag{3}$$

oder

$$N = int(C) \text{ und } rnd < C - int(C) \tag{4}$$

   Dabei ist N die Anzahl der Fahrzeuge, welche die Kante in diesem Simulationsschritt bereits verlassen haben. Ist diese kleiner als der ganzzahlige Anteil von C, so kann ein weiteres Fahrzeug die Kante in diesem Zeitschritt verlassen und N wird um eins erhöht.

   Wollen stets weitere Fahrzeuge die Kante verlassen, so erreicht N den ganzzahligen Anteil von C $[N = int\,(C)]$. Ist dies der Fall, kann genau ein weiteres Fahrzeug die Kante verlassen, falls die Zufallszahl $rnd \in [0:1[$ kleiner ist als der Rest von C $[rnd < C - int\,(C)]$. Die Verwendung dieser zweiten Bedingung (Gleichung 4) ermöglicht es mit einer gewissen Wahrscheinlichkeit die verbleibende Restkapazität der Kante zu nutzen.

   Da die Bedingung am Ende der Kante und nicht an deren Anfang überprüft wird ist es möglich, dass innerhalb eines simulierten Zeitschritts mehr Fahrzeuge die Kante betreten als durch die *FlowCapacity* der zu betretenden Kante zulässig wären. Dies ist zum Beispiel der Fall, wenn sich in einem Knoten mehrere auf ihn zuführende Kanten bündeln.

3. Die nachfolgende Kante hat Platz: Wird die *StorageCapacity* der folgenden Kante erreicht, ist diese voll und kein weiteres Fahrzeug kann die Kante betreten. Ein verschieben des Fahrzeuges auf die nachfolgende Kante unterbleibt.

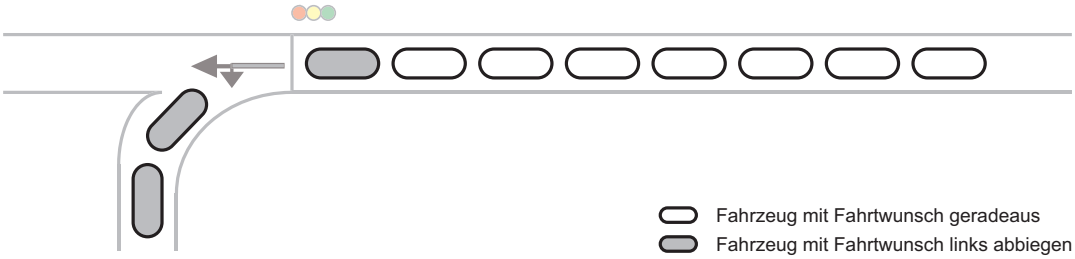


Abbildung 1: Blockieren einer mehrspurigen Kante durch ein einzelnes Fahrzeug

Sind alle diese drei Bedingungen erfüllt, so wird das Fahrzeug über den Knoten bewegt. Auf dem Knoten selbst wird kein Verkehr simuliert. Fahrzeuge werden ohne Zeitverlust von einer Kante auf die nachfolgende Kante verschoben.

2.1.3 Diskussion des Modells

Im Vergleich zu einem Zellularautomaten wird der Verkehrsfluss weniger realistisch abgebildet wie z.B. im Falle der Behandlung von am Ende der Kante entstandenen Lücken, die innerhalb eines Zeitschrittes bis zum Anfang der Kante wandern und dort dem nachfolgenden Verkehr zur Verfügung stehen. Im Zellularautomaten wandert solch eine Lücke nur langsam zum Anfang der Kante.

Weiter einschränkend kommt hinzu, dass Fahrzeuge unterschiedlicher Art wie Pkw, Lkw, Busse oder auch Fahrräder alle als dieselbe Art von Fahrzeugen behandelt werden. Dazu zählt, dass alle mit der gleichen maximal zulässigen Höchstgeschwindigkeit auf der Kante unterwegs sind. Komplexe Verkehrsabläufe wie das Überholen von langsameren Fahrzeugen oder Spurwechsel sind nicht möglich.

Ein dritter Punkt wird durch den Schwerpunkt dieser Arbeit adressiert. Laut Gawron (1998) können Verzögerungen an Knotenpunkten dadurch modelliert werden, dass die Kapazität der Kanten modifiziert wird. Jedoch trifft dies für signalisierte Knotenpunkte nicht zu. Entscheidend für die Kapazität eines signalisierten Knotenpunktes ist das komplexe Zusammenspiel zwischen den auftretenden Verkehrsströmen und dem darauf optimierten Signalzeitenplan der Lichtsignalanlage. Damit verbunden ist der Mangel an separaten Abbiegespuren. Wie in Abbildung 1 dargestellt, kann ohne Erweiterung des Modells ein einzelnes Fahrzeug, welches auf eine zugestaute Nebenstraße abbiegen möchte, den kompletten mehrspurigen Verkehr auf den anderen Abbiegebeziehungen zum Erliegen bringen, da alle Abbiegebeziehungen gezwungen sind dieselbe Queue zu nutzen.

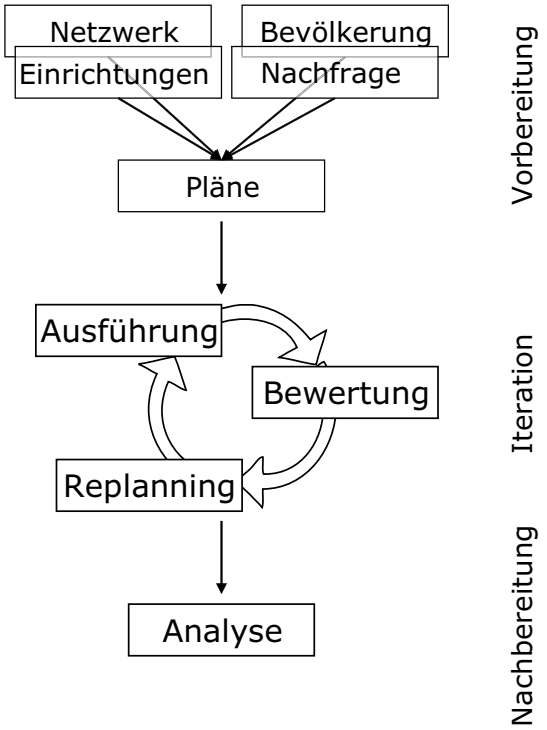


Abbildung 2: Der Ablauf einer MATSim-Simulation kann in die drei Abschnitte Vorbereitung, Iteration und Nachbereitung unterteilt werden. Der Schwerpunkt der Arbeit liegt im Bereich der Ausführung.

2.2 MATSim

MATSim bedeutet „Multi Agent Traffic Simulation“ und ist der Name einer hauptsächlich an der Eidgenössischen Technischen Hochschule Zürich und der Technischen Universität Berlin entwickelten Mikrosimulation für den Verkehr. Dabei kann für jeden Verkehrsteilnehmer, im Folgenden Agent genannt, individuelles und zeitabhängiges Verhalten simuliert werden. MATSim ist in der Lage, Szenarios mit bis zu 8 Millionen Agenten gleichzeitig zu simulieren und dabei Netze zu verwenden, die mehrere hunderttausend Kanten umfassen und beispielsweise die gesamte Schweiz abdecken. Dabei hat jeder einzelne Agent einen bis mehrere Pläne, welche verschiedene Aktivitäten enthalten. Diese Pläne werden für alle Agenten während der Verkehrsflusssimulation simultan ausgeführt. Werden mehrere Simulationen hintereinander ausgeführt, so sind die Agenten in der Lage, zwischen den Simulationen ihre Pläne zu modifizieren.

Der Ablauf einer MATSim-Simulation kann in drei Abschnitte eingeteilt werden, welche in Abbildung 2 dargestellt sind. Der erste Abschnitt ist die Vorbereitung aller für die Simulation benötigter Eingangsdaten. Als Eingangsdaten dienen eine Netzwerkbeschreibung, Beschreibungen von Einrichtungen wie Arbeitsplätzen und Sportstätten, die zu

simulierende Bevölkerung und die daraus resultierende anfängliche Nachfrage. Aus den Eingangsdaten werden für jeden aus der Bevölkerung generierten Agenten Pläne erstellt.

Im zweiten Abschnitt, der eigentlichen Simulation, kommen diese Pläne zur Ausführung. Dabei wird genau ein Plan jedes Agenten mit Hilfe der Verkehrsflusssimulation ausgeführt. Im Anschluss wird der ausgeführte Plan eines Agenten bewertet. Da die Ausführung für alle Agenten simultan erfolgt, beeinflussen sich Agenten gegenseitig und konkurrieren um begrenzte Ressourcen wie die Kapazität einer Kante. Nach der Bewertung erhält ein Teil der Agenten die Möglichkeit seinen Plan zu modifizieren. Dieser Vorgang wird als Replanning bezeichnet. Im Anschluss kommen die Pläne erneut zur Ausführung. Durch die Wiederholung der drei Schritte Ausführung, Bewertung und Replanning wird das Lernen der Agenten simuliert. Eine solche Wiederholung wird als Iteration bezeichnet.

Der dritte und letzte Abschnitt, die Nachbereitung, befasst sich mit der Analyse der bei jeder Iteration erzeugten Daten. Das Ergebnis einer Simulation mit MATSim ist eine Datei, in der alle Ereignisse (Events) aufgelistet sind. Diese Datei kann als Ableitung der Simulation verstanden werden.

Im Folgenden soll auf die für diese Arbeit wichtigen Bestandteile von MATSim genauer eingegangen werden.

Planerzeugung, -ausführung und -modifizierung Jeder Agent kann mehrere Pläne besitzen. Ein Plan enthält dabei alle Aktivitäten des Agenten für einen Tag. Unter Aktivitäten versteht MATSim z.B. Tätigkeiten wie Arbeiten, Wohnen, Freizeit oder Einkaufen. Eine Aktivität besitzt zusätzlich zum Typ Eigenschaften wie deren Anfang und deren Ende oder die mindestens dort zu verbringende Zeit. Zusätzlich ist in den Plänen vermerkt, auf welcher Kante sich die Aktivität befindet und wie der Agent von einer Aktivität zur sich anschließenden Aktivität kommt. Diese sogenannten Legs beinhalten Informationen zum verwendeten Fortbewegungsmittel und die zu fahrende Route nebst deren Länge und voraussichtlicher Fahrtzeit.

Zu Beginn der Iteration muss jeder Agent einen ihm gehörenden Plan auswählen und diesen als aktiv markieren. Dieser Plan wird simultan zu den Plänen der anderen Agenten ausgeführt. Die Verkehrsflusssimulation protokolliert während der Ausführung die verschiedenen Ereignisse in Form von Events. Zu den wichtigsten Events gehören die Zeitpunkte des Betretens und Verlassens einer Kante oder Aktivität. Aus diesen Daten lässt sich hinterher rekonstruieren, wie lange ein Agent für die einzelnen Abschnitte seines Plans benötigt hat. Diese Information wird als Grundlage benutzt, um den Plan

bewerten zu können. Die erhaltene Bewertung dient anschließend dazu, den vorhandenen Plan zu modifizieren oder einen anderen bereits existierenden Plan für die nachfolgende Iteration als aktiv zu markieren.

Replanning Lediglich ein Teil der Agenten darf im Anschluss an die Ausführung seinen Plan modifizieren. Zu den Möglichkeiten der Modifikation gehört die Wahl einer anderen Abfahrtzeit, um besonders staugefährdete Zeiträume zu meiden. Alternativ kann auch eine neue Route zwischen zwei Aktivitäten gesucht werden.

Der wohl am häufigsten benutzte Algorithmus dafür ist der von Dijkstra (1959) entwickelte und nach ihm benannte Dijkstra-Algorithmus zum Finden kürzester Wege. Dessen erweiterte zeitbasierte Variante wird in MATSim verwendet. Die zeitbasierte Variante erlaubt es, die Kosten der Kante in Abhängigkeit des Startzeitpunktes zu variieren. Identische Routen liefern zum Beispiel zu unterschiedlichen Zeitpunkten verschiedene Reisezeiten.

Der Algorithmus benötigt als Datengrundlage ein Netzwerk in dem alle Kanten miteinander verbunden sind und jeder mögliche Zielknoten zumindest vom Startknoten aus erreichbar ist. Die in MATSim verwendeten Netzwerke sind in dieser Hinsicht weniger restriktiv, da bei ihnen lediglich von jedem aus den Plänen der Agenten abgeleiteten Startknoten aus jeder aus den Plänen abgeleitete Zielknoten erreicht werden können muss. Da die Pläne der Agenten aber die Anfragen zur Routensuche bestimmen, werden auch nur dem Netz nach gültige Anfragen gestellt.

Darüber hinaus werden Angaben zu den zeitabhängigen Kosten jeder Kante benötigt. MATSim unterscheidet die zwei Kostenobjekte „Reisezeit je Kante“ und „Kosten je Kante“. Im einfachsten Fall werden die Kosten der Kante direkt aus den Reisezeiten abgeleitet. Weitere Kosten können jedoch zum Beispiel in Höhe einer Maut beaufschlagt werden.

Im Falle der Reisezeit wird dafür die beim Eintritt in die Kante zu erwartende Reisezeit verwendet. Diese entspricht in der ersten Iteration der $FreeLinkTravelTime\ tt_0$. In den folgenden Iterationen entspricht sie den von den Agenten realisierten Reisezeiten. Um diese zu erhalten werden die während der Simulation generierten Events ausgewertet.

Für jeden Agenten wird beim Betreten und beim Verlassen einer Kante jeweils ein Event generiert, die sogenannten *LinkEnter*- und *LinkLeave*-Events. Diese beinhalten eine eindeutige Identifikationsnummer sowohl des Agenten sowie auch der Kante für welches das Event geschrieben wurde. Zusätzlich enthält das Event einen Zeitstempel, welcher angibt, wann das Event geschrieben wurde. Aus diesen Informationen lässt sich

für jede Kante durch Differenzenbildung der beiden Zeitstempel ableiten, wie lange jeder Agent zum Passieren dieser Kante gebraucht hat.

Da auf dem Knoten keine zeitlich relevanten Aktionen ausgeführt werden, wird ein Fahrzeug innerhalb eines Zeitschrittes von einer auf die nächste Kante transferiert. Infolgedessen werden *LinkLeave*-Event einer Kante und *LinkEnter*-Event der nachfolgenden Kante zum gleichen Zeitpunkt generiert und sind inhaltlich austauschbar.

In einem zweiten Schritt werden die Reisezeiten in einzelnen Zeitintervallen zusammengefasst. Ein Zeitintervall beinhaltet die arithmetisch gemittelten Reisezeiten aller Agenten, die innerhalb z.B. einer viertel Stunde die Kante betreten haben. Dabei ist es unerheblich, wann der Agent die Kante wieder verlassen hat. Dies kann durchaus in einem der nachfolgenden Zeitintervalle geschehen. Da der Dijkstra-Algorithmus vom Startknoten beginnend vorwärts rechnet, ist es für diesen entscheidend zu wissen, welche Reisezeit zu erwarten ist, wenn zum Zeitpunkt X die Kante betreten wird.

Von der Berechnung nicht berücksichtigt werden Agenten, die auf der Kante einer Aktivität nachgehen. Die Dauer einer Aktivität kann mehrere Stunden betragen, so dass das zum *LinkEnter*-Event korrespondierende *LinkLeave*-Event gegebenenfalls auch erst nach mehreren Stunden generiert werden würde. Die resultierende Fahrtzeit von mehreren Stunden würde damit das Ergebnis für die Kante verfälschen, da diese nicht die tatsächliche Fahrtzeit repräsentiert.

Liegen für ein bestimmtes Zeitintervall einer Kante keine Reisezeitinformationen vor, weil in besagtem Intervall kein Agent die Kante betreten hat, so wird die Kante als frei angenommen und die Reisezeit entspricht der $FreeLinkTravelTime$. Die neu berechnete Route wird abschließend zusammen mit ihrer Länge und der zu erwartenden Fahrtzeit in den Plan übernommen.

2.2.1 Erweiterung des Warteschlangenmodells nach Gawron

Zurzeit bietet MATSim eine Reihe von Modulen zur Simulation und Analyse von Verkehr an. Der modularisierte Aufbau erlaubt es, einzelne Module separat zu nutzen oder diese zu kombinieren. Entscheidend für diese Arbeit ist das Modul Verkehrsflusssimulation, welches das Warteschlangenmodell implementiert. Im Idealfall wird dieses durch eine eigene Implementierung ersetzt und kann mit Hilfe der restlichen unveränderten MATSim-Module getestet werden. Dazu wird im folgenden Abschnitt beschrieben, welche weiteren Veränderungen bei der Implementierung von MATSim an dem bisher vorgestellten Warteschlangenmodell nach Gawron vorgenommen wurden.

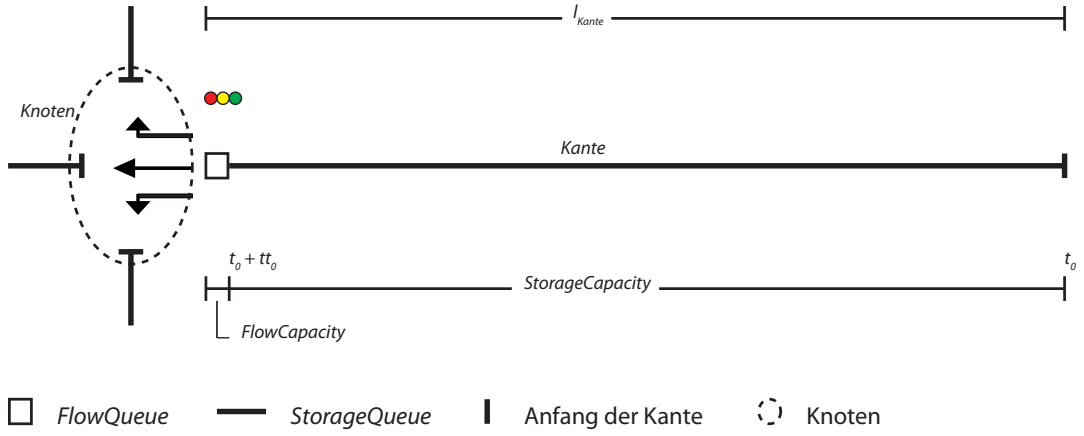


Abbildung 3: Interner Aufbau einer Kante in MATSim
Dargestellt ist die Anbindung einer Kante, bestehend aus *StorageQueue* und *FlowQueue*, an einen Knoten. In MATSim werden alle Abbiegebeziehungen auf eine Kante abgebildet.

Die Implementierung von MATSim erweitert das Modell nach Gawron durch die Einführung eines Buffers am Ende der Kante. Bei diesem handelt es sich um eine weitere Queue. Die Anzahl der Fahrzeuge, die sich gleichzeitig in der Queue aufhalten dürfen, entspricht der Höhe des ganzzahligen Anteils der *FlowCapacity* bzw. der nächst höheren ganzzahligen Zahl, wenn es sich bei der *FlowCapacity* um einen Bruch handelt. Im Folgenden wird dieser Buffer entsprechend seiner Bedeutung *FlowQueue* genannt. Eine Kante in MATSim besitzt also eine *StorageQueue* und eine daran angeschlossene *FlowQueue*. Grundsätzlich bestimmt die *StorageCapacity* die Größe der *StorageQueue* und die *FlowCapacity* die Größe der *FlowQueue*. In Abbildung 3 ist deren Beziehung zueinander grafisch veranschaulicht. Die Kante ist mit ihren beiden Bestandteilen *FlowQueue* (Quadrat) und *StorageQueue* (Linie) dargestellt. Für ein besseres Verständnis des sich anschließenden Knotens ist in Abbildung 4 ein kompletter mit diesem Modell umgesetzter vierarmiger Knotenpunkt zu sehen.

Abweichend zu dem in Unterabschnitt 2.1.2 beschriebenen Verfahren wird jetzt die erste Bedingung (Fahrzeug hat das Ende der Kante erreicht) nicht mehr am Ende der Kante geprüft, sondern es wird geprüft, ob das erste Fahrzeug der *StorageQueue* diese Bedingung erfüllt. Ist diese erfüllt und lässt die Kapazität der sich anschließenden *FlowQueue* die Aufnahme weiterer Fahrzeuge zu (zweite Bedingung) wird das Fahrzeug in die *FlowQueue* verschoben und nicht wie bisher über den Knoten bewegt. Dieser Vorgang wiederholt sich für alle Kanten im Netz. Anschließend wird über alle Knoten iteriert und die dritte Bedingung, nachfolgende Kante hat freie Kapazität, geprüft. Das Abgreifen der Fahrzeuge aus der *FlowQueue* geschieht während des Iterierens über alle Knoten.

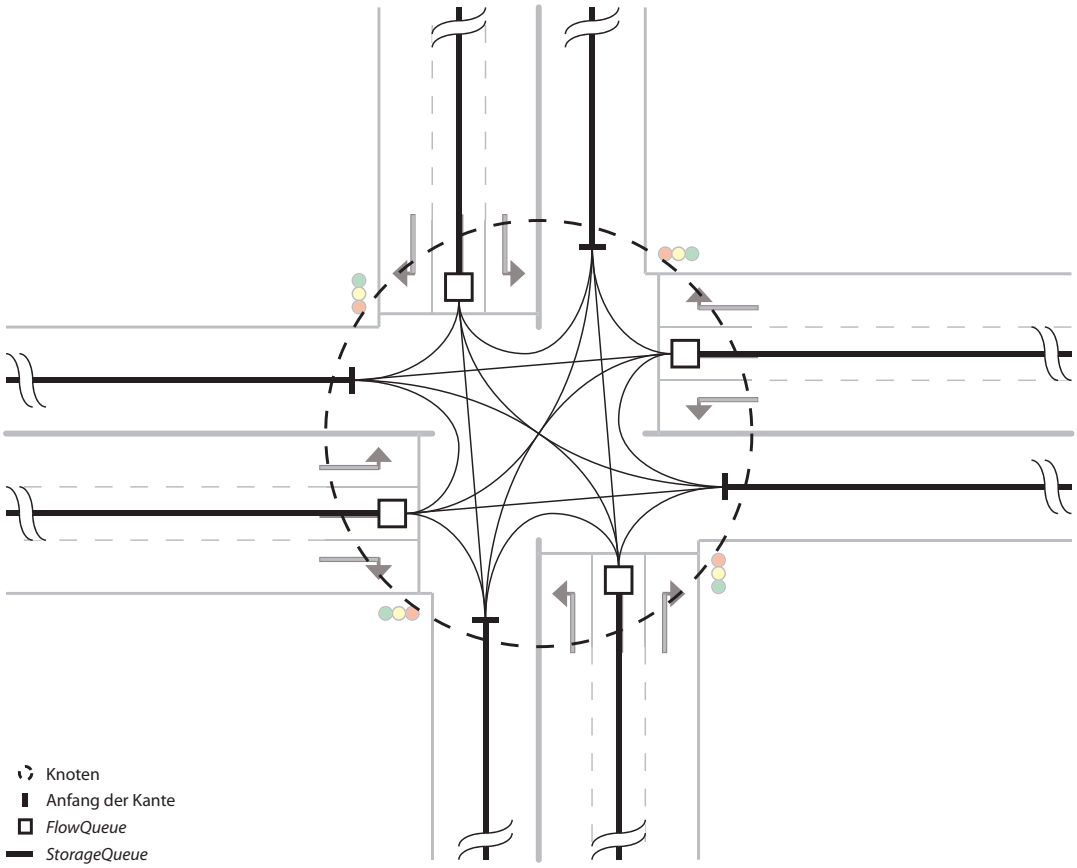


Abbildung 4: Vierarmiger Knotenpunkt, umgesetzt mit der bisherigen Implementierung von MATSim
Alle Abbiegebeziehungen und Fahrstreifen eines Armes werden mittels einer Kante modelliert. Die vier zuführenden und die vier wegführenden Strecken werden demnach durch insgesamt acht Kanten präsentiert. Da bei den zuführenden Kanten alle Abbiegebeziehungen auf einer Kante zusammengefasst sind, kann auch von einer Kante in alle Richtungen abgebogen werden (geschwungene Linien). Das beinhaltet auch das Wenden, da alle wegführenden Kanten gleichberechtigt sind und Abbiegebeziehungen nicht explizit definiert werden.

Eine weitere Änderung betrifft die Art, mit der die vorhandene *FlowCapacity* umgesetzt wird. Meist umfasst ein Simulationsschritt in MATSim eine Sekunde. Es ist daher anzunehmen, dass für typische Straßen mit einer Kapazität von 600 bis 4200 Fz/h die *FlowCapacity* in den meisten Fällen gleich oder weniger ein Fahrzeug je Simulationsschritt beträgt und daher die Bedingung in Gleichung 4 auf Seite 16 ausgewertet wird. Abweichend zu der dort geschilderten Verwendung einer Zufallszahl, wird im von MATSim verwendeten Modell die verbliebene Restkapazität $(C - int\,(C))$ aufsummiert. Übersteigt die Summe in den folgenden Simulationsschritten eins, so kann ein zusätzliches Fahrzeug in die *FlowQueue* verschoben werden. Anschließend wird die Summe der Restkapazität um eins verringert.

Die für die *StorageCapacity* maßgebende Länge eines Fahrzeuges l_{Fz} wird in MATSim mit durchschnittlich 7,5 m angenommen. Dieser Wert berücksichtigt den bei kompletttem Stillstand benötigten Abstand zwischen zwei Fahrzeugen und ist ein bei zellulären Automaten üblicher Wert.

In der derzeitigen Implementierung in MATSim besitzt jede Kante mehrere Queues, um für diese Kante relevante Fahrzeuge zu speichern. Im vorangegangenen Abschnitt wurden die *StorageQueue* und die *FlowQueue* bereits erwähnt.

Neu hinzu kommt die in Abbildung 3 nicht dargestellte *ParkingQueue*. Diese besteht eigentlich aus mehreren Datenstrukturen, wird hier aber vereinfacht als eine einfache Queue angenommen. Sie speichert alle Fahrzeuge, die sich zwar auf der Kante befinden aber derzeit nicht am Verkehrsgeschehen teilnehmen, weil sie beispielsweise gerade einer der Kante zugeordneten Aktivität nachgehen. Als Sortierkriterium der Queue dient die gewünschte Abfahrtszeit der Fahrzeuge, also in den meisten Fällen das Ende der momentanen Aktivität. Die Größe der *ParkingQueue* ist nicht limitiert.

Bei jedem Simulationsschritt wird kontrolliert, ob die Abfahrtszeit des ersten Fahrzeugs der *ParkingQueue* bereits erreicht oder überschritten wurde. Ist dies der Fall wird das Fahrzeug bei vorhandener Kapazität in die *FlowQueue* gesetzt. Kapazität ist vorhanden, solange die Restriktionen hinsichtlich der *FlowCapacity* beachtet werden.

Ein bei dem bisher vorgestellten Verkehrsflussmodell noch bestehendes Problem ist die Reihenfolge, in der über die einzelnen Kanten iteriert wird. Diese ist bisher fest, so dass in jedem Zeitschritt die gleichen Kanten bevorzugt werden. Wann welche Kante eines Knotens aufgerufen wird, ist jedoch nicht trivial. Das folgende auf Abbildung 5 basierende Beispiel soll dies veranschaulichen.

An einem Knoten sind die beiden Kanten 1 und 3 zuführend, aber lediglich eine Kante 2 wegführend. In jedem Zeitschritt werden die Kanten in der Reihenfolge 1-2-3

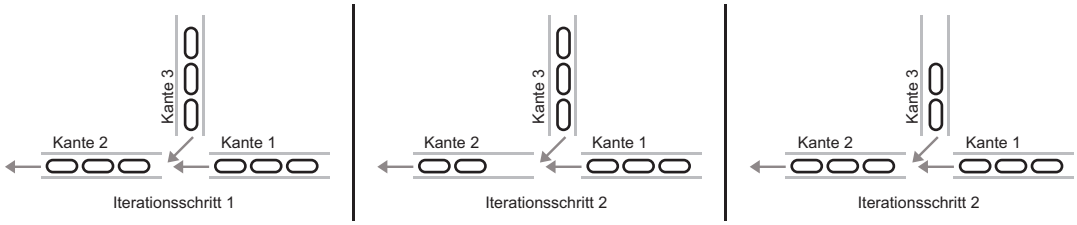


Abbildung 5: Auswirkung der Abarbeitungsreihenfolge der Kanten eines Knotens auf die realisierbare Kapazität der zuführenden Kanten

abgearbeitet. Angenommen die Queue der wegführenden Kante 2 ist komplett gestaut. Dann kann von Kante 1 kein Fahrzeug nachrücken. Anschließend wird Kante 2 aufgerufen und es entsteht eine Lücke für genau ein Fahrzeug. Diese kann dann von Kante 3 genutzt werden, womit Kante 2 wiederum komplett gestaut ist. Der Vorgang wiederholt sich in jedem Zeitschritt. Kante 1 hat somit immer das Nachsehen, obwohl bisher keine Vorfahrtsregel implementiert wurde. Resultat ist ein ruhender Verkehr auf Kante 1 und ein im Idealfall frei fließender Verkehr auf Kante 3.

Abhilfe schafft das zufällige Auswählen der auf einen Knoten zuführenden Kanten, der sogenannten *inLink*s. Die momentane Implementierung gewichtet darüber hinaus mit der *FlowCapacity* der einzelnen *inLink*s, so dass *inLink*s mit einer hohen *FlowCapacity* bevorzugt werden.

Dazu sortiert jeder Knoten beim ersten Aufruf innerhalb einer Simulation seine *inLink*s. Deren Reihenfolge wird für spätere Aufrufe zwischengespeichert und erlaubt es, in jedem Zeitschritt in der gleichen Reihenfolge die *inLink*s abzuarbeiten. Während eines Zeitschrittes werden alle *inLink*s vermerkt, bei denen sich Fahrzeuge in der zugehörigen *FlowQueue* befinden. Die anderen *FlowQueue*s bleiben in diesem Zeitschritt unberücksichtigt.

Die Reihenfolge, in der die *FlowQueue*s der verbliebenen *inLink*s angesprochen werden, ist abhängig von der Gesamtkapazität aller vermerkten *inLink*s und der Kapazität des zu einer *FlowQueue* gehörenden *inLink*s. Die Wahrscheinlichkeit, dass Fahrzeuge den Knotenpunkt passieren können, steigt mit der Kapazität des zur *FlowQueue* gehörenden *inLink*s. Wird eine *FlowQueue* ausgewählt, so werden alle sich darin befindenden Fahrzeuge entsprechend ihrer gewünschten Zielkante über den Knoten gesetzt, falls die Kapazität der Zielkante es zulässt. Sollte dies einmal nicht der Fall sein, so verbleibt dieses Fahrzeug in der *FlowQueue* und staut die sich hinter ihm befindlichen Fahrzeuge. Anschließend wird die nächste *FlowQueue* aufgerufen. Der Vorgang wird so lange wiederholt, bis alle vermerkten *inLink*s mit ihrer *FlowQueue* an der Reihe waren.

Wird ein Fahrzeug über einen Knoten bewegt, so verlässt es eine Kante und betritt eine neue. An dieser Stelle ist es Aufgabe der Kanten, dies durch das Schreiben eines *LinkLeave*-Events bzw. Aufgabe der Zielkante, dies durch das Schreiben eines *Link-Enter*-Events zu vermerken.

2.3 Ingenieurtechnische Bewertung

Bei der ingenieurtechnischen Bewertung geht es darum, das Modell anhand fester Größen zu vermessen und diese an den derzeit gültigen technischen Regelwerken zu verifizieren. Innerhalb des deutschsprachigen Raums sind dies vor allem die mit den deutschen „Richtlinien für die Anlage von Straßen“ (RAS) vergleichbaren österreichischen „Richtlinien und Vorschriften für den Straßenbau“ (RVS) und die für die Schweiz geltenden beiden Teilnormen SN640022 und SN640023a, beide herausgegeben vom „Schweizer Verband der Straßen- und Verkehrsfachleute“.

Für die Niederlande gibt es das vergleichbare „Handboek verkeerslichtenregelingen“ herausgegeben vom CROW, für Kanada den „Canadian Capacity Guide for Signalized Intersections“ (CCG) und für die USA das „Highway Capacity Manual“ (HCM).

Für die Bewertung in dieser Arbeit wird das in Deutschland gültige „Handbuch für die Bemessung von Straßenverkehrsanlagen“, kurz HBS, verwendet. Dieses setzt als Grundlage eine Reihe von Daten voraus, welche teils aus der Definition des Netzwerkes und dem Lageplan entnommen werden können, teilweise aber auch in den die LSA beschreibenden Daten enthalten sind oder erhoben werden müssen. Die folgenden Daten werden dabei als Eingangsgrößen benötigt:

- Definition von Lageplan und erlaubten Knotenströmen
- Kapazität der Zugangsstraßen
- Kapazität der einzelnen Fahrstreifen
- Stärke der einzelnen Verkehrsströme
- Phasensystem (gegeben über Signalzeitenplan).

Im Folgenden werden verschiedene Kriterien vorgestellt mit denen sich nach HBS die Qualität eines Knotenpunktes bewerten lässt.

2.3.1 Bewertung anhand der Knotenpunktkapazität

Die Knotenpunktkapazität C_K ergibt sich durch Aufsummieren der einzelnen Fahrstreifenkapazitäten aller Fahrstreifen i eines Knotens:

$$C_K = \sum_{i=1}^{n} \frac{t_{Fi}}{t_U} \cdot q_{Si} \tag{5}$$

Die Freigabezeit t_F und die Umlaufzeit t_U sind dem Signalzeitenplan zu entnehmen.

q_S ist die sogenannte Sättigungsverkehrsstärke. Diese gibt die maximale Anzahl an Fahrzeugen an, die den Fahrstreifen bei ungehindertem Abfluss innerhalb einer Grünstunde ($t_F = 3600s$) passieren kann. Ist diese nicht gegeben, kann sie mit Hilfe des mittleren Zeitbedarfswertes t_B zu $q_s = \frac{1}{t_B}$ berechnet oder nach Tabelle 1 interpoliert werden. Auf die Verwendung von Angleichungsfaktoren wie im HBS angegeben wird verzichtet. Diese verringern die Sättigungsverkehrsstärke weiter und werden für den Fall eines hohen Schwerverkehrsanteils, schmaler Fahrstreifen, enger Abbiegeradien, starker Fahrbahnlängsneigungen und starken Fußgängerverkehrs angewendet. Liegen Daten dazu vor und sind Auswirkungen auf ein Szenario zu erwarten, können diese später einfach implementiert werden.

Tabelle 1: Sättigungsverkehrsstärken q_S (HBS, Tabelle 6-4)

t_F [s/Fz] aus SZP	q_S [Pkw/h]	t_B [s/Fz]
>10	2000	1,8
10	2400	1,5
6	3000	1,2

Innerhalb der Simulation werden die während einer Stunde über einen Knotenpunkt gesetzten Fahrzeuge gezählt und mit C_K verglichen. Voraussetzung ist das Vorhandensein der maximal möglichen Verkehrsnachfrage je Fahrstreifen, so dass die zur Verfügung stehende Kapazität auch nahezu vollständig genutzt wird. Alternativ können auch Knotenarme einzeln betrachtet werden.

Weiterführende Diskussion Der hier dargestellte Weg für die Bewertung orientiert sich am HBS. Zur Verwendung des Zeitbedarfswertes t_B gibt es jedoch unterschiedliche Auffassungen, welche im Folgenden diskutiert werden sollen.

Die Kapazität einer 2-streifigen Straße liegt zwischen 2500 Fz/h für eine Landstraße und 4200 Fz/h für einen Autobahnabschnitt innerhalb eines Ballungsraums (HBS). Umgerechnet auf Sekunden liegt die Kapazität für eine Fahrspur zwischen 0,34 und 0,59 Fz/s oder umgerechnet auf den Zeitbedarfswert zwischen 1,69 und 2,94 s/Fz.

Die Sättigungsverkehrsstärke einer Abbiegespur kann jedoch stark von der eines normalen Straßenabschnittes ohne Knotenpunkt abweichen. Es muss daher entschieden werden, welche Kapazität für die Abbiegespuren gilt. Diese kann zum Beispiel dem HBS folgend nach Art des Fahrstreifens festgelegt werden. In diesem Fall wird im Wesentlichen zwischen Abbiege- und Geradeausspur unterschieden.

Alternativ kann wie oben beschrieben der sogenannte Zeitbedarfswert t_B herangezogen werden, welcher sich nach der zu erwartenden Grünzeit richtet. Dieser wurde für die Schweiz empirisch ermittelt, siehe IVT, Kapitel 6.17. Es konnte gezeigt werden, dass die ersten Fahrzeuge nach einem Phasenwechsel bedingt durch Verluste beim Anfahren einen leicht erhöhten Zeitbedarfswert t_B von 2,2 s haben. Nachfolgende Fahrzeuge besitzen je nach Kolonnenlänge einen geringeren Zeitbedarfswert von 1,6 bis 1,8 Sekunden. Daraus resultieren Kapazitäten für eine komplette Stunde Grünzeit zwischen 1636 Fz/h und bis zu 2250 Fz/h bei den Kolonnen.

Die oben verwendeten Berechnungen nach HBS kommen zu einem gegensätzlichen Ergebnis. Dort wird davon ausgegangen, dass bei kürzeren Freigabezeiten der Anteil der von den Fahrzeugen genutzten Gelbzeit steigt. Infolgedessen sinkt der Zeitbedarfswert t_B mit abnehmenden Freigabezeiten, siehe auch Tabelle 1. Demnach liegt die Kapazität zwischen 2000 und 3000 Fz/h.

Beiden Argumentationen kann gefolgt werden. Aufgrund der diametralen Auswirkungen wurde für diese Arbeit entschieden, dass die Dauer der zu erwartenden Grünzeit für die Berechnung nicht explizit berücksichtigt wird. Die Sättigungsverkehrsstärke wird demnach weder erhöht noch gesenkt, sondern leitet sich direkt aus der im Netzwerk definierten Kapazität ab.

Zum weiteren Vergleich soll hier die Argumentation des „Canadian Capacity Guide for Signalized Intersections“ (CCG, Seite 3-39) wiedergegeben werden. Demnach benötigen die ersten fünf bis sieben Fahrzeuge zu Beginn einer Grünphase mehr Zeit und es kann nicht die volle Kapazität genutzt werden. Bei 6 s Grünzeit werden lediglich 88 % der Kapazität genutzt. Ab einer Grünzeit von 20 Sekunden wird die volle Kapazität genutzt. Für gesättigte Grünphasen ab einer Länge von 50 s sinkt die Kapazität anschließend wieder auf 90 %. Soweit folgt die Argumentation dem schweizerischen Modell und die

Sättigungsverkehrsstärke einer Fahrspur mit einer Kapazität von 1800 Fz/h und einer Grünzeit von 10 s liegt bei 1649 Fz/h.

Im Folgenden rechnet das CCG jedoch mit einer effektiven und nicht wie im HBS mit der signalisierten Grünzeit weiter. Diese beträgt für die Auslegung 1 s mehr als die signalisierte Grünzeit, kann aber bei der Bewertung von nahe an der Kapazität belasteten Knotenpunkten auch mehr als 2 s darüber liegen (CCG, Seite 3-59). Infolgedessen wird ein Teil des Kapazitätsverlustes wieder ausgeglichen und das CCG berücksichtigt wie das HBS die Nutzung der Gelbzeit.

2.3.2 Bewertung anhand der Knotenpunktqualität

Die Bewertung anhand der Knotenpunktqualität wurde innerhalb dieser Arbeit in Teilen angewandt, um das entwickelte Modell zu testen. Da sie in der ingenieurtechnischen Bewertung von Signalzeitenplänen eine große Rolle spielt und sie zu einem späteren Zeitpunkt zur Bewertung der Pläne herangezogen werden kann, sollen hier auch die nicht verwendeten Bestandteile kurz beschrieben werden.

Die Qualität eines Knotenpunktes leitet sich aus dem zeitlichen Aufwand seitens der Verkehrsteilnehmer ab, der zum Passieren des Knotenpunktes benötigt wird. Im Wesentlichen setzt sich dieser aus der Wartezeit und der Anzahl der Halte zusammen. Zur Berechnung der Wartezeit an einem Knotenpunkt, muss bekannt sein, wie stark die einzelnen Fahrstreifen ausgelastet sind. Dazu wird der Sättigungsgrad g nach HBS wie folgt berechnet:

$$g = \frac{q}{q_S} \cdot \frac{t_U}{t_F} \tag{6}$$

q ist dabei die tatsächlich auftretende Belastung, die auf der Spur gemessen wird und q_S die Sättigungsverkehrsstärke. Bei einem Sättigungsgrad von größer eins ist der Fahrstreifen überlastet und führt unweigerlich zu einem Rückstau. Sättigungsgrade zwischen 0,9 und 1 können bei ungleichmäßiger Ankunft der Fahrzeuge und damit Belastung zu kurzzeitigen Überlastungen führen, die aber langfristig wieder abgebaut werden können.

Ist der Sättigungsgrad bekannt kann die mittlere Wartezeit w berechnet werden. Diese setzt sich aus der Grundwartezeit w_G, verursacht durch die Sperrung der Knotenpunktzufahrt durch die LSA, und der Reststauwartezeit w_R zusammen (siehe Gleichung 7). Die Reststauwartezeit zwingt Fahrzeuge, mehrmals nachzurücken, bis sie den eigentlichen Knotenpunkt überqueren können. Laut HBS ist der Reststau bis zu einem Sättigungsgrad von $g \leq 0,65$ null, für $0,65 < g \leq 0,90$ konstant und für $g > 0,90$ zeitabhängig wachsend.

Die mittlere Wartezeit berechnet sich nach HBS zu:

$$\begin{aligned} w &= w_G + w_R \\ w &= \frac{t_U \left(1 - \frac{t_F}{t_U}\right)^2}{2\left(1 - \frac{q}{q_S}\right)} + \frac{3600 \cdot N_{GE}}{q_S} \cdot \frac{t_U}{t_F} \end{aligned} \tag{7}$$

N_{GE} ist die Anzahl der gestauten Fahrzeuge bei Grünende und abhängig vom Sättigungsgrad g. Es kann aus dem HBS, Tabelle 6-6 entnommen werden. Spezielle Wartezeiten für ÖPNV, Fußgänger und Radfahrer werden ignoriert, da MATSim diese bisher nicht speziell berücksichtigt.

Des Weiteren kann die Anzahl der zum Halten gezwungenen Fahrzeuge je Umlauf n_H berechnet werden:

$$n_H = \frac{q\,(t_U - t_F + N_{GE} \cdot t_B)}{3600\left(1 - \frac{q}{q_S}\right)} \tag{8}$$

Dies kann entweder absolut geschehen oder als prozentualer Anteil aller zum Halten gezwungenen Fahrzeuge an der Gesamtzahl. Als Halt gilt das Blockieren eines Fahrzeuges, obwohl auf dem Ziellink freie Kapazität vorhanden ist. Dabei kann maximal ein Halt pro Umlauf zustande kommen. Diese Art der Bewertung ist für diese Arbeit nur bedingt aussagekräftig, spielt aber bei der Wirksamkeitsuntersuchung von Knotenpunktkoordinierungen eine große Rolle.

3 Entwicklung des Modells

In diesem Kapitel werden die Anforderungen an das zu entwickelnde Verkehrsflussmodell formuliert und anschließend das Modell spezifiziert.

3.1 Anforderungen an das Modell

Im Folgenden sollen die drei Hauptforderungen an das zu entwickelnde Modell definiert werden, welche es mindestens zu erfüllen hat.

1. Der erste Punkt ist die Überprüfung des neuen Modells hinsichtlich seines Verhaltens wenn keine Signalzeitenpläne vorliegen und somit Dauergrün geschaltet ist. Dies ist wichtig, wenn für das Modell keine oder nur für einen Teil der Knotenpunkte Eingangsdaten zu den Lichtsignalanlagen vorliegen und signalisierte und unsignalisierte Knoten parallel simuliert werden müssen. Dieser Fall wird immer dann Auftreten, wenn innerhalb eines Verkehrsnetzes nicht alle Knotenpunkte signalisiert sind.

   Das neue Modell mit Lichtsignalanlage aber ohne Signalzeitenpläne sollte dabei den gleichen Verkehrsfluss generieren und die gleiche Knotenpunktkapazität aufweisen können wie das ursprüngliche Modell ohne LSA. Ein einfacher Knoten mit nur einem *inLink* und einem *outLink* wie zum Beispiel an einem signalisierten Fußgängerüberweg sollte bei Dauergrün keinen Abfall in der Kapazität verzeichnen.

   Wichtig an dieser Stelle ist, dass das Verkehrsflussmodell verantwortlich ist für das Verschieben der Fahrzeuge und damit der Agenten von einer Kante auf die nächste. Damit liegt es auch in der Verantwortung dieses Moduls, die anderen Bestandteile der Simulation darüber zu informieren, welcher Agent wann welche Kante betreten oder verlassen hat. Da dies durch das Generieren spezieller Events geschieht, muss eine neue Implementierung in der Lage sein, die von ihr erwarteten Events zu schreiben.

2. Der zweite Punkt ist eine an die Realität angelehnte Abbildung der Spuraufteilung am Knotenpunkt. Dazu sollen nicht die exakten Verhältnisse wie auf dem Lageplan verzeichnet wiedergegeben werden, sondern lediglich die für die Kapazität des Knotenpunktes entscheidende Anzahl der zur Verfügung stehenden Fahrstreifen am Knotenpunkt, deren Abbiegebeziehungen und Länge. Entscheidenden Einfluss

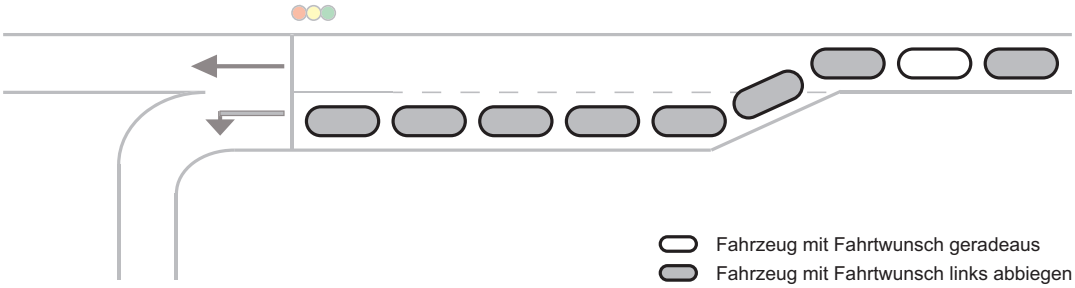


Abbildung 6: Überstauen von Abbiegespuren
Fahrzeuge mit dem Fahrtwunsch links abbiegen blockieren durch überstauen ihrer Abbiegespur Fahrzeuge mit einem anderen Fahrtwunsch, hier das weiße Fahrzeug mit dem Fahrtwunsch geradeaus fahren.

haben diese Größen, wenn sich eine Straße am Knotenpunkt aufweitet und damit an der Haltelinie eine höhere Kapazität genutzt werden kann.

3. Dem angeschlossen ist die Forderung an das Modell, das Überstauen von Abbiegespuren und deren Auswirkungen abbilden zu können. Bei einem Verkehrsaufkommen auf einer der Abbiegespuren weit über deren für die Auslegung zu Grunde liegenden Belastung soll diese sich zustauen. Nachdem die Abbiegespur komplett gestaut ist, soll sich der Stau auf die stromaufwärts liegenden Streckenabschnitte ausweiten und dort gegebenenfalls den Zugang zu anderen Abbiegespuren blockieren können. Der Vorgang ist in Abbildung 6 veranschaulicht.

Es werden im Folgenden zwei Ansätze für die LSA-Modellierung innerhalb eines Warteschlangenmodells vorgestellt. Beide Ansätze werden hinsichtlich ihrer Eignung für die Verwendung in MATSim diskutiert. Dazu wird jeder Ansatz zuerst spezifiziert und anschließend dessen Auswirkungen auf die Simulation beurteilt.

3.2 Netzwerkmodifikationsmodell

Der erste Ansatz für ein Modell wird im Folgenden Netzwerkmodifikationsmodell genannt. Dieser Ansatz zielt darauf ab die Eingangsdaten von MATSim und hier insbesondere das Netzwerk zu modifizieren. Im Gegenzug kann die bisherige interne Netzwerkarchitektur von MATSim genutzt werden. Änderungen sind dann lediglich bei den die Knotenpunktlogik betreffenden Klassen nötig.

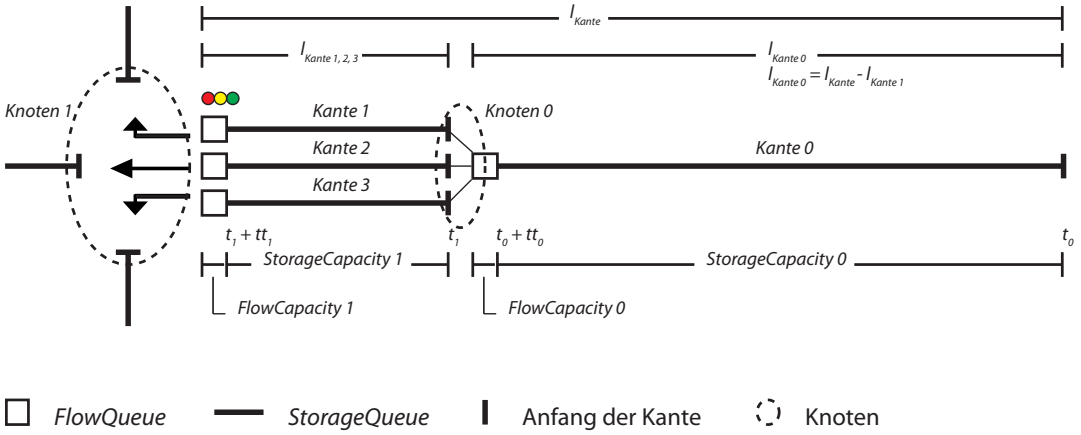


Abbildung 7: Interner Aufbau einer Kante im Netzwerkmodifikationsmodell
Diese Modifikation kürzt die originale Kante 0 und ersetzt den fehlenden Abschnitt durch eine vollwertige Kante je Abbiegebeziehung (Kanten 1 bis 3)

3.2.1 Spezifikation

Für dieses Modell müssen die Eingangsdaten von MATSim verändert werden. Um das Abbiegen von mehreren Fahrspuren aus zu ermöglichen, werden gesonderte Kanten für das Abbiegen eingeführt. In Abbildung 7 sind dies die Kanten 1 bis 3. Die neuen Abbiegekanten werden wie echte Kanten bereits in der Netzdatei definiert. Damit sind sie von den konventionellen Kanten nicht zu unterscheiden und werden während der Generierung des Simulationsnetzes wie vollwertige Kanten behandelt und auch als solche modelliert. Ihre Länge richtet sich nach dem Lageplan des Knotenpunktes. Jede Abbiegekante besitzt wie eine konventionelle Kante eine *ParkingQueue*, eine *StorageQueue* und eine *FlowQueue*. In den Plänen der Agenten werden die Aktivitäten unverändert auf die gleichen Kanten referenziert. Die Agenten starten ihre Fahrt in der *ParkingQueue* der ursprünglichen Kante 0 und werden anschließend in deren *FlowQueue* gesetzt. Analog beenden die Agenten ihre Fahrt am Ende der *StorageQueue* von Kante 0. Demzufolge existieren keinerlei Aktivitäten auf den Abbiegekanten 1, 2, 3 und deren *ParkingQueue* wird von keinem Agenten genutzt. Im Ablauf der Simulation ändert sich daher nichts. Die *ParkingQueue*s der Abbiegekanten werden dennoch in jedem Simulationsschritt abgefragt, nur befinden sich keine Fahrzeuge in ihnen.

Für die Erzeugung der Abbiegekanten wird die ursprüngliche Kante 0 in ihrer Länge gekürzt. Die Länge der Abbiegekanten 1, 2 und 3 entspricht dem gekürzten Abschnitt, so dass die Gesamtlänge mit der ursprünglichen Kante übereinstimmt. Die Verbindung

zwischen der gekürzten Kante 0 und den neuen Abbiegekanten wird über einen zusätzlich eingefügten Knoten realisiert (Knoten 0 in Abbildung 7).

Die *FlowCapacity* der Abbiegekanten wird entweder von der ursprünglichen Kante übernommen oder, wenn zum Beispiel Daten zu der Anzahl und Art der Spuren vorliegen, entsprechend angepasst. Die *FlowCapacity* der gekürzten Kante 0 bleibt unverändert. Dagegen muss die *StorageCapacity* für alle Kanten angepasst werden, da diese sich nach der Länge und der Anzahl der Fahrspuren richtet. Ebenfalls von der Länge abhängig ist die *FreeLinkTravelTime* tt_0, welche daher gleichfalls entsprechend der Länge angepasst werden muss.

Ein kompletter Knotenpunkt modelliert mit dem Netwerkmodifikationsmodell ist in Abbildung 8 dargestellt. Dieser verfügt über vier zuführende und vier wegführende Kanten. Die möglichen Abbiegebeziehungen sind mit geschwungenen Linien dargestellt. Für jede Abbiegebeziehung musste zuvor eine neue Kante im Netzwerk definiert werden, hier bestehend aus *FlowQueue*, *StorageQueue* und Anfang der Kante. Die zusätzlich eingefügten Knoten sind als kleine gestrichelte Knoten ohne Signalzeitenplan dargestellt.

3.2.2 Auswirkungen

Vorteile Verfolgt man diesen Ansatz, so ist als größter Vorteil der geringe Implementierungsaufwand innerhalb von MATSim zu nennen. Das Einlesen des Netzwerks und der Pläne geschieht mit den gleichen Methoden wie bisher. Darüber hinaus kann das Modul zur Routensuche unverändert mit dem vorhandenen Kürzeste-Wege-Algorithmus von Dijkstra durchgeführt werden. Informationen zu den Reisezeiten werden über die Events für jede Kante separat generiert. Es liegen somit Informationen zu den Reisezeiten für das Abbiegen (Abbiegekanten 1, 2 und 3) und die eigentliche Kante (gekürzte ursprüngliche Kante 0) getrennt vor. Für den Dijkstra-Algorithmus werden Abbiege- und ursprüngliche Kanten gleich behandelt. Damit kann der Dijkstra mit den veränderten Netzwerken ohne Anpassungen verwendet werden.

Nachteile Als Nachteil ist die veränderte Abbildung der Routen zu nennen. Durch die Hinzunahme neuer Kanten sind die bisher in den Plänen der Agenten gespeicherten Routen unvollständig. Es werden zusätzlich Knoten eingeführt, die so nicht in den Plänen der Agenten existieren aber von den Agenten befahren werden müssen um auf dem neuen Netz ihr Ziel erreichen zu können. Sollen bisher verwendete Pläne in dem neuen Modell ebenfalls verwendet werden, müssen sie zuerst für das neue Netzwerk konvertiert werden. Umgekehrt gilt das auch für durch das neue Modell generierte Pläne. Hier müssen die

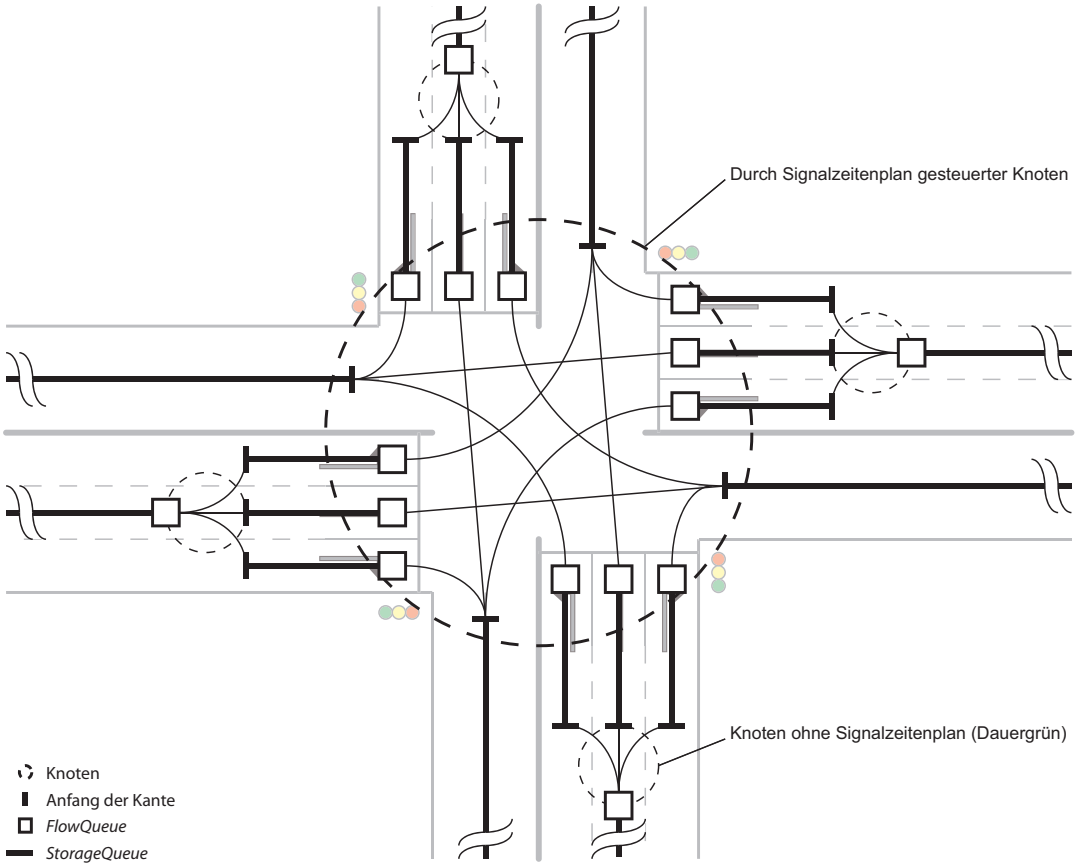


Abbildung 8: Vierarmiger Knotenpunkt, umgesetzt mit dem Netzwerkmodifikationsmodell
Dargestellt ist ein vierarmiger Knotenpunkt. Bei Verwendung des Netzwerkmodifikationsmodells werden für jede auf einen signalisierten Knotenpunkt zuführende Kante eine neue Kante je Abbiegebeziehung (geschwungene Linien) eingeführt. Jede dieser Kanten verfügt wiederum über eine *FlowQueue*, eine *StorageQueue* und, hier zusätzlich dargestellt, einen Anfang. Als Verbindung zwischen alten und neuen Kanten werden zusätzliche Knoten ohne Signalzeitenplan eingeführt.

Routen für das alte Netzwerk zurück konvertiert werden. Es muss also nicht nur ein neues Netzwerk definiert und generiert werden, sondern es entsteht auch zusätzlicher Aufwand bei der Konvertierung der übrigen Daten.

Weit folgenschwerer ist die veränderte Generierung der Events. Wie bereits in Abschnitt 2.2 erwähnt, findet die Abbildung der Simulation in Form der Events statt. Dieser Output ist damit Grundlage für den Vergleich der Ergebnisse zweier Simulationen. Die neuen Abbiegekanten werden in dieser Hinsicht behandelt wie die ursprünglichen Kanten. Als Folge sind deren Events von der Struktur her gleich denen der Abbiegekanten und beide Typen von Kanten sind nicht voneinander zu unterscheiden. Ergebnisse einer Simulation mit Lichtsignalanlage können somit nicht direkt mit bereits bestehenden Ergebnissen einer konventionellen Simulation ohne Lichtsignalanlage verglichen werden. Die Events der einen Simulation müssen daher unter Zuhilfenahme der Netzwerke umständlich übersetzt werden.

Fazit Die Nachteile insbesondere in der Kompatibilität zu der vorhanden Infrastruktur und den Daten überwiegen. Zusätzlich birgt jede Konvertierung der Daten bzw. deren Vor- und Nachbearbeitung die Gefahr zusätzlicher Fehler. Das Netzwerkmodifikationsmodell wird aus diesen Gründen als nicht kompatibel zur bisherigen MATSim-Struktur angesehen und zugunsten eines alternativen Ansatzes verworfen. Dieser wird im folgenden Abschnitt beschrieben.

3.3 Subnetzmodell

Der Schwerpunkt des in diesem Abschnitt vorgestellten Ansatzes liegt auf der Kompatibilität zu bereits bestehenden Eingangsdaten wie Netzwerken und Plänen der Agenten, wie auch auf der Vergleichbarkeit der Simulationsergebnisse. Der Ansatz fügt auf jeder konventionellen Kante ein Subnetz virtueller Kanten ein.

3.3.1 Spezifikation

Die Eingangsdaten in Form der Agentenpläne und des Netzwerks werden unverändert eingelesen. Das eingelesene Netzwerk wird in einem zweiten Schritt erweitert. Dazu wird auf eine konventionelle Kante ein Netz von virtuellen Kanten, sogenannten Links, gelegt. Virtuell deshalb, da sich diese nach außen gegenüber der Simulation wie eine einzelne konventionelle Kante verhalten. Für einige Knoten liegen Informationen zu Lichtsignalanlagen vor und zu anderen nicht. Die auf Knoten ohne Lichtsignalanlage zuführenden

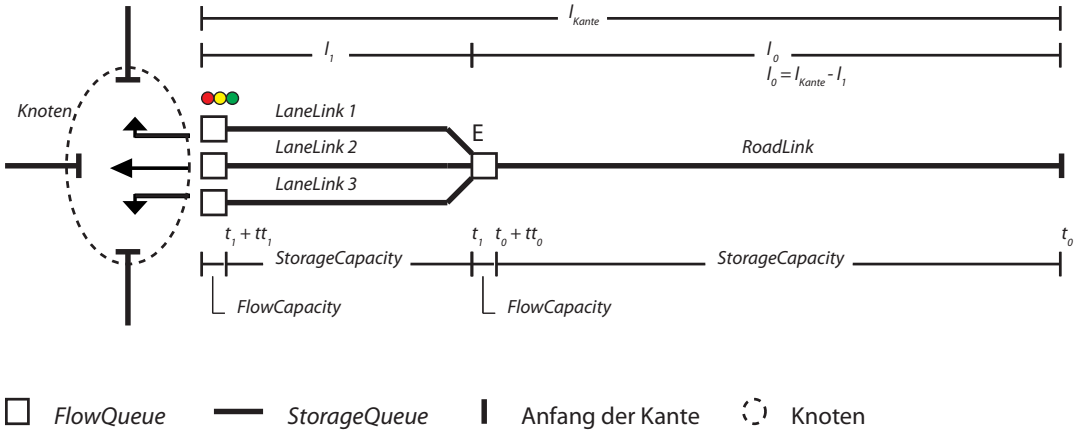


Abbildung 9: Interner Aufbau einer Kante im Subnetzmodell
Diese Modifikation ergänzt die ursprüngliche Kante um eine zusätzliche virtuelle Kante je Abbiegebeziehung

Kanten unterscheiden sich auch in ihrer internen Struktur nicht von konventionellen Kanten. Deren virtuelles Netzwerk besteht im Wesentlichen aus einem einzelnen Link, welcher wie in Unterabschnitt 2.2.1 beschrieben über eine *StorageQueue*, eine *FlowQueue* und eine *ParkingQueue* verfügt. Für eine Kante, die auf einen Knoten mit Lichtsignalanlage zuläuft, ist das interne virtuelle Netzwerk in Abbildung 9 dargestellt.

Für das in Abbildung 9 dargestellte Beispiel wurden in den die Lichtsignalanlage beschreibenden Daten drei Signalgruppen definiert, die jede für sich eine eigene Fahrspur steuert. Jede Signalgruppe erlaubt das Abbiegen auf eine andere Kante. Für jede Fahrspur wird die ursprüngliche Kante um einen zusätzlichen *LaneLink* ergänzt. Sie sind in Abbildung 9 als *LaneLink* 1 bis 3 dargestellt. Jeder *LaneLink* verfügt über eine *StorageQueue* und eine *FlowQueue*. Diese im Folgenden *LaneStorageQueue* und *LaneFlowQueue* bezeichneten Queues besitzen die gleiche Aufgabe wie die Abbiegekanten im Netzwerkmodifikationsmodell. Es handelt sich dabei ebenfalls um eine Art virtueller Kanten, die nicht in der Netzwerkdatei existieren und erst zur Laufzeit generiert werden. Die *LaneStorageCapacity* hängt von der Distanz zwischen dem Knotenpunkt und der Stelle, an welcher die Abbiegespur beginnt, ab, also von der Länge der Abbiegespur l_1 selbst. Alle Abbiegespuren besitzen in diesem Modell die gleiche Länge, zweigen also am gleichen Entscheidungspunkt E von der ursprünglichen Kante, dem *RoadLink*, ab. Liegen Daten zu den Längen der einzelnen Spuren vor, können diese auch im Modell berücksichtigt werden. Der *RoadLink* wird von seiner Gesamtlänge l_{Kante} um die Länge der Abbiegekanten (*LaneLink* 1 bis 3) auf $l_0 = l_{Kante} - l_1$ gekürzt. Dessen *StorageCapacity* wird der neuen Länge entsprechend angepasst. Die *LaneFlowCapacity* der

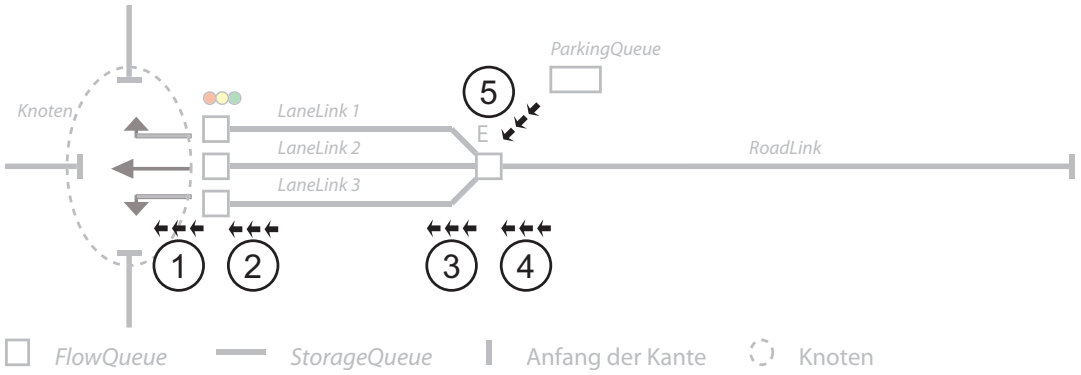


Abbildung 10: Abarbeitungsreihenfolge der Bestandteile eines Subnetzes
Schritt 1 wird während des Iterierens über alle Knoten ausgeführt, die restlichen Schritte in aufsteigender Reihenfolge während des Iterierens über alle Kanten

einzelnen *LaneLink*s muss ebenfalls angepasst werden. Diese richtet sich zum Beispiel nach der Anzahl der Fahrspuren. Eine genauere Erläuterung des Themas findet sich in Unterabschnitt 3.4.1.

Weitere Änderungen ergeben sich aus der veränderten internen Struktur. Da die Pläne der Agenten nicht modifiziert werden, starten und enden deren Routen auf jeweils einer einzelnen Kante. In Konsequenz werden sie von der Simulation an die Kante mit dem Startpunkt weitergereicht. Für Kanten ohne Lichtsignalanlage ändert sich nichts. Bei den anderen handelt es sich jetzt um ein virtuelles Netzwerk, in welches das Fahrzeug mit dem Agenten aufgenommen werden muss. In diesem Modell wird dieses in die *ParkingQueue* des *RoadLink*s gesetzt und nach Beginn der Fahrt an dessen *FlowQueue* an Punkt E weiter gereicht. Dagegen erreicht ein Fahrzeug sein Ziel, wenn es die *StorageQueue* des *RoadLink*s verlassen kann. Auf den *LaneLink*s finden keine Aktivitäten statt. Infolgedessen beginnen und enden dort auch keine Routen und eine *ParkingQueue* wird nicht benötigt.

Eine weitere Änderung ergibt sich aus dem Iterieren über alle Knoten und alle Kanten. Während des Iterierens über die Knoten werden die Fahrzeuge aus der *LaneFlowQueue* abgegriffen und in die *StorageQueue* des nachfolgenden *RoadLink*s des *outLink*s transferiert. Dies ist in Abbildung 10 mit Schritt 1 dargestellt. In der anschließenden Iteration über die Kanten werden Fahrzeuge erst aus der *LaneStorageQueue* in die *LaneFlowQueue* (Schritt 2), dann in Schritt 3 von der *FlowQueue* des *RoadLink*s in die gewünschte *LaneStorageQueue* und abschließend aus der *StorageQueue* des *RoadLink*s in dessen *FlowQueue* transferiert (Schritt 4). Im letzten Schritt 5 werden Fahrzeuge aus der *ParkingQueue* in die *FlowQueue* des *RoadLink*s gesetzt. Voraussetzung ist dabei

immer die Beachtung der Restriktionen hinsichtlich vorhandenen Platzes in den Queues und minimal benötigter Reisezeit. Kann ein Fahrzeug nicht in die gewünschte *LaneStorageQueue* gesetzt werden, so verbleibt es in der *FlowQueue* des *RoadLink*s und staut gegebenenfalls Fahrzeuge mit einer abweichenden *LaneStorageQueue* als Ziel.

Als letzten Punkt ändert sich die Art, mit der bisher implementierte Routensuchalgorithmen Abbiegebeziehungen unterschiedlich bewerten können und in ihrer Routensuche berücksichtigen. Anmerkungen zur Routensuche und den dafür benötigten Änderungen werden gesondert im Unterabschnitt 3.4.2 behandelt.

3.3.2 Auswirkungen

Vorteile In diesem Modell können die bisherigen Pläne und Netzwerke weiter verwendet werden. Liegen keinerlei Informationen zu Lichtsignalanlagen vor, so ändert sich für die Simulation nichts. Die ursprünglichen Kanten werden in einzelne virtuelle Kanten ausgelagert. Sie sind damit einfach gekapselt und können wie bisher über dieselben Schnittstellen aufgerufen werden.

Ist eine Kante dagegen durch eine Lichtsignalanlage gesteuert, so ändert sich dessen interne Struktur grundlegend. Aus Sicht der Simulation hat sich jedoch ebenfalls nichts geändert, denn sie ist wiederum gekapselt und wird weiterhin über die gleichen Schnittstellen angesprochen.

Nachteile Der einzige Nachteil ergibt sich aus dem Implementierungsaufwand. Die beschriebene Generierung eines Subnetzes ist wesentlich komplexer als die bisherige Implementierung und verursacht daher sehr wahrscheinlich längere Rechenzeiten. Dieses trifft jedoch auch auf das Netzwerkmodifikationsmodell zu, da dieses mehr echte Kanten zu simulieren hat. Es kann an dieser Stelle nicht geklärt werden, welches der Modelle mehr Ressourcen in Anspruch nehmen wird. Des Weiteren muss ein komplexeres Modell umfassender getestet werden, um Fehler ausschließen zu können, was Zeit bei der Umsetzung in Anspruch nimmt.

Fazit Die Umsetzung ist gegenüber dem Netzwerkmodifikationsmodell komplexer und wird mehr Ressourcen kosten. Dieser Aufwand entsteht jedoch nur ein einziges Mal, so dass die Vorteile langfristig überwiegen werden. Es wurde daher beschlossen den Ansatz des Subnetzmodells weiter zu verfolgen, es zu implementieren und zu testen. Diese Schritte werden in den folgenden beiden Kapiteln beschrieben. Zunächst gilt es jedoch, das Modell genauer zu spezifizieren.

3.4 Ausarbeitung des Subnetzmodells

Im Folgenden werden einzelne Aspekte des Subnetzmodells genauer ausgearbeitet und spezifiziert. Die hier beschriebenen Ausarbeitungen dienen als Grundlage für die spätere Beispielimplementierung.

3.4.1 Fluss der einzelnen Links

Die *FlowCapacity* einer Kante wird durch die Definition des Netzwerks festgelegt. Bei Verwendung des Subnetzmodells überträgt sich diese im unsignalisierten Fall auf den einzig vorhandenen Link und bestimmt dort die Anzahl der Fahrzeuge, die in der *FlowQueue* maximal Platz finden.

Im signalisierten Fall existieren sowohl *LaneLink*s wie auch *RoadLink*s. Die *FlowCapacity* bestimmt die Kapazität der *FlowQueue*s aller *RoadLink*s. Für diese werden alle Werte der ursprünglichen Kante übernommen und entsprechend ihrer tatsächlichen Länge gemäß den Gleichungen 1 und 2 angepasst. Für die Werte der *LaneLink*s wird deren Länge zu Grunde gelegt.

Betrachtet man den Raum kurz vor der Kreuzung, der durch die *LaneLink*s abgedeckt wird, so kann sich in der Realität die Straße in diesem Abschnitt aufweiten. Die Kapazität von zusätzlichen Abbiegespuren steht dem Verkehr zur Verfügung. Eine höhere Anzahl an Fahrstreifen ändert zum einen die vorhandene Stellfläche. Es können je Streckenkilometer mehr Fahrzeuge aufgenommen werden, was sich insbesondere bei einem Stau auswirkt und durch die verschiedenen parallelen *LaneStorageQueue*s repräsentiert wird. Zum anderen können durch eine höhere Zahl an Fahrstreifen auch mehr Fahrzeuge einen Querschnitt gleichzeitig passieren. Weitet sich an einem Knotenpunkt eine zweispurige Straße auf vier Fahrstreifen auf, so können bei gleichzeitiger Signalisierung theoretisch doppelt so viele Fahrzeuge am Anfang der nächsten Grünphase die Haltelinie passieren.

Für die Umsetzung im Subnetzmodell gibt es demnach zwei Möglichkeiten. Die erste Möglichkeit leitet aus dem Lageplan oder dem nach Unterabschnitt 3.4.5 rekonstruierten Lageplan die Art der Fahrstreifen ab. Abbiegende, geradeaus führende und Mischfahrstreifen weisen nach HBS unterschiedliche Kapazitäten auf und können pauschal übernommen werden. Nachteilig wirkt sich aus, dass, falls die Daten dazu vorliegen, für jeden Knotenpunkt jeder Fahrstreifen separat definiert werden muss oder aber auf der Rekonstruktion basierend eine Abschätzung vorgenommen wird, welche stark von den wirklichen Verhältnissen abweichen kann.

Die zweite Möglichkeit umgeht das Problem der Datenbeschaffung und schätzt die Kapazität direkt aus der im Netzwerk definierten Kapazität der ursprünglichen Kante. Aus den Daten des Netzes ist bekannt, wie hoch die Kapazität der Kante ist und auf wie viele Spuren sich diese aufteilt. Damit lässt sich für diese Kante eine durchschnittliche Kapazität $C_{Durchschnitt}$ je Fahrspur berechnen.

$$C_{Durchschnitt} = \frac{C}{n_{Spuren}} \tag{9}$$

Der Durchschnitt wird anschließend auf die *LaneLink*s übertragen. Besitzt ein *LaneLink* mehrere Fahrspuren so erhöht sich dessen Kapazität entsprechend.

$$C_{LaneLink} = C_{Durchschnitt} \cdot n_{SpurenLaneLink} \tag{10}$$

Weitet sich eine zweispurige Straße am Knotenpunkt durch zwei separate Abbiegespuren auf vier Spuren auf und verbleiben zwei Spuren für den geradeaus fahrenden Verkehr, so besitzt der ihn repräsentierende *LaneLink* die gleiche Kapazität wie der *RoadLink* ($2 \cdot C_{Durchschnitt}$). Die beiden abbiegenden *LaneLink*s besitzen dagegen jeweils nur die Hälfte der Kapazität ($1 \cdot C_{Durchschnitt}$). Dieses Verfahren wurde letztlich umgesetzt.

3.4.2 Routensuche

Die Funktionsweise des für die Routensuche verwendeten Dijkstra-Algorithmus ändert sich für dieses Modell nicht, wohl aber die Daten, mit denen er versorgt wird. Da beim Dijkstra jeder Kante genau eine Reisezeit zugeordnet wird, ist es für den Algorithmus nicht möglich, abhängig von der Wahl der nachfolgenden Kante verschiedene Reisezeiten zu beaufschlagen. Dazu ist für die Suche ein Netzwerk nötig, welches die Abbiegebeziehungen gesondert darstellen kann, so dass jeder Abbiegebeziehung eine individuelle Reisezeit zugeordnet werden kann.

Ein möglicher Ansatz ist die Verwendung eines invertierten Netzwerkes. Dieser Ansatz wurde erfolgreich in der Dissertation von Flötteröd (2008) verwendet. Dazu wird für die Routensuche ein neues Netzwerk auf Basis des ursprünglichen Netzwerkes erstellt und eine Route als Folge von Abbiegevorgängen statt als Folge von Kanten dargestellt. Zum besseren Verständnis ist der Vorgang in Abbildung 11 dargestellt.

In einem ersten Umwandlungsschritt wird für jede Kante im ursprünglichen Netzwerk ein Knoten mit gleicher Identifikationsnummer erstellt. Anschließend wird in einem zweiten Schritt für jeden *outLink* des am Ende der ursprünglichen Kante befindlichen

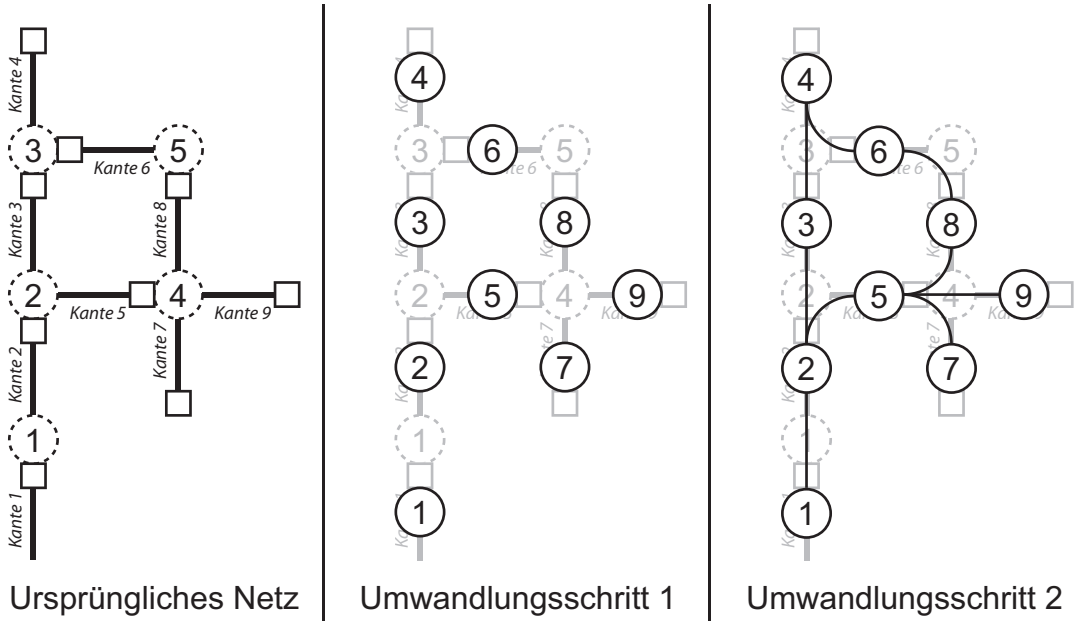


Abbildung 11: Erstellung eines invertierten Netzwerks
Dargestellt sind das ursprüngliche Netzwerk links (Ausschnitt), das Netzwerk nach dem ersten Umwandlungsschritt in der Mitte und das fertig invertierte Netzwerk rechts

Knotens eine neue Kante angelegt. Zur Veranschaulichung dient das folgende, auf Abbildung 11 basierende, Beispiel. Der Knoten 2 des ursprünglichen Netzes besitzt Kante 2 als *inLink* und die beiden Kanten 3 und 5 als *outLink*s. Kante 2 wird im ersten Umwandlungsschritt zu Knoten 2. Im invertierten Netz wird Knoten 2 anschließend durch Einfügen neuer Kanten mit den Knoten 3 und 5 verbunden. Als Resultat ist ein Knoten im ursprünglichen Netz durch eine Reihe von Kanten ersetzt, welche alle möglichen *inLink-outLink* Kombinationen dieses Knotens repräsentieren.

Die Routensuche erfolgt, indem die Kanten der Start- und der Zielaktivität übergeben werden. Diese können direkt in einen Start- und einen Zielknoten innerhalb des invertierten Netzes übersetzt werden. Anschließend erfolgt eine Suche nach dem kürzesten Weg. Als Resultat liegt eine Route als Folge von Knoten des invertierten Netzes vor. Diese wird abschließend in eine Folge von Knoten des ursprünglichen Netzes konvertiert, indem zu jeder rückübersetzten Kante der am Anfang der Kante befindliche Knoten gesucht wird.

Sucht man zum Beispiel eine Route von Kante 1 nach Kante 4 (Abbildung 11 links), so erhält man die beiden invertierten Routen 1-2-3-4 und 1-2-5-8-6-4. Auf das ursprüngliche Netz übertragen entspricht dies den Kanten 1-2-3-4 bzw. 1-2-5-8-6-4. Diese werden zu

den auf Knoten basierenden Routen 1-2-3 und 1-2-4-5-3. Der erste Knoten der Route ist Knoten 1, da nach den derzeitigen MATSim-Plänen Routen als Folge von sich zwischen den Aktivitäten befindenden Knoten definiert werden. Der hier nicht dargestellte Knoten am Beginn der Kante 1 entfällt. Demnach fehlt der Route die Information darüber, auf welcher Kante sich die Aktivität befindet. Diese Information ist bereits in der Aktivität enthalten und Routen beginnen auf dem sich am Ende der Kante anschließenden Knoten.

Mit dem invertierten Netzwerk ist die Grundlage dafür geschaffen, für jede Abbiegebeziehung eine andere Reisezeit angeben zu können. Bisher entspricht die Reisezeit einer Kante aber der gemittelten Reisezeit aller Agenten die innerhalb eines Zeitintervalls die Kante betreten haben. Es wird dabei nicht unterschieden, welche der nachfolgenden Kanten der Agent anschließend betritt. Als Folge betrachtet der Dijkstra alle Abbiegebeziehungen als unabhängige Kanten, kann aber mangels weiterer Informationen lediglich alle ausgehenden Kanten eines *inLink*s nur mit den gleichen Reisezeiten beaufschlagen. Um das gesamte Potential des Netzwerkes ausschöpfen zu können, müssen auch separate Daten für die Abbiegebeziehungen vorliegen. Im Folgenden werden zwei Möglichkeiten diskutiert, wie die Reisezeiten anders erfasst werden können.

Möglichkeit 1 - Verwendung der bisherigen Datenbasis

Bei der Auswertung der *LinkEnter*- und *LinkLeave*-Events wird zwischen den einzelnen Zielkanten der Agenten unterschieden. Es wird also das *LinkEnter*-Event der nachfolgenden Kante ebenfalls mit ausgewertet. Dadurch kann für jede gewünschte Abbiegebeziehung eine individuelle Reisezeit angegeben werden. Probleme gibt es, wenn alle Fahrzeuge eine Abbiegebeziehung bevorzugen und daher für diese dann Reisezeiten vorliegen, für die anderen hingegen nicht. In diesem Fall wird für alle anderen Richtungen in Ermangelung von Informationen die $FreeLinkTravelTime$ zu Grunde gelegt. Diese Annahme muss aber nicht korrekt sein, da durch die einseitige Belastung die Gefahr eines Staus besteht, welcher Fahrzeuge der anderen Abbiegebeziehungen ebenfalls beeinträchtigt (vgl. Abbildung 1 und 6). Dieses Problem wird sich voraussichtlich nach mehreren Iterationen einer MATSim-Simulation lösen, da die Agenten in den Fahrzeugen in der Lage sind zu lernen und diese „falschen“ Reisezeiten zu berücksichtigen.

Möglichkeit 2 - Einführung eines zusätzlichen Events

Es wird ein zusätzliches *LaneLinkEnter*-Event eingeführt. *LinkEnter* wird dabei wie bisher beim Betreten der ersten *StorageQueue* einer Kante geschrieben. Das *Lane-*

LinkEnter-Event wird beim Betreten der *LaneStorageQueue* generiert. Das Schreiben von *LinkLeave* erfolgt wie bisher beim Verlassen der *LaneFlowQueue*.

Bei der anschließenden Auswertung der Events und der Aufbereitung für den Dijkstra, müssen die Reisezeiten für jede Abbiegebeziehung getrennt zusammengeführt werden. Die Zeit zwischen *LinkEnter*- und *LaneLinkEnter*-Event ist dabei für alle Abbiegebeziehungen gleich. Der Rückstau einer Richtung innerhalb dieses Abschnitts wirkt sich damit auf alle Richtungen aus. Zusätzlich kommt eine für jede Beziehung unterschiedliche Zeit hinzu, die dem Zeitraum zwischen *LaneLinkEnter*- und *LinkEnter*-Event der nachfolgenden Kante entspricht. Diese spiegelt den Zeitaufwand für den Abbiegevorgang wider und kann sich je nach Signalzeitenplan stark in den einzelnen Richtungen unterscheiden.

Geht ein Agent auf der Kante einer Aktivität nach, so darf er bei der Berechnung der Reisezeit wiederum nicht berücksichtigt werden. Folgt auf ein *LinkEnter*-Event direkt ein *LinkLeave*-Event ohne ein *LaneLinkEnter*-Event dazwischen, so kann davon ausgegangen werden, dass es sich um eine Kante handelt, die auf einen unsignalisierten Knotenpunkt zuläuft. Zur weiteren Vereinfachung kann auf das *LinkLeave*-Event verzichtet werden. Dessen Funktion kann vom nachfolgenden *LinkEnter*-Event übernommen werden.

3.4.3 Implementierung der Lichtsignalanlage

Es gibt verschiedene Ansätze dafür, eine Lichtsignalanlage zu implementieren. Simon u. Nagel (1998) haben für einen Zellularautomaten ein virtuelles Fahrzeug mit der Geschwindigkeit null an der Haltelinie eingefügt. Dieses zwingt die nachfolgenden Fahrzeuge zum Anhalten. Schaltet die Lichtsignalanlage wieder auf grün, wird dieses entfernt.

Ein analoger Ansatz kann hier umgesetzt werden. Dazu müsste am Anfang der *LaneStorageQueue* ein virtuelles Fahrzeug vor alle anderen Fahrzeuge eingefügt werden, welches bei „grün“ wieder entfernt wird. Es muss nicht in jedem Zeitschritt die *LaneStorageQueue* modifiziert werden. Ein Fahrzeug wird hineingesetzt bzw. wieder entfernt, wenn sich der Schaltzustand für die *LaneStorageQueue* ändert. Dazu muss in jedem Zeitschritt nachgeschlagen werden, ob sich der Schaltzustand geändert hat. Handelt es sich um einen gemischten Fahrstreifen, so ist entscheidend, in welche Richtung das erste Fahrzeug abbiegen möchte. Auch auf gemischten Fahrstreifen ist es möglich, dass nur eine Fahrtrichtung grün geschaltet ist, auf der anderen jedoch nicht. Ein virtuelles Fahrzeug am Anfang der Queue blockiert alle Fahrzeuge, egal mit welchem Ziel. Es kann also nicht unterlassen werden zu prüfen, in welche Richtung das erste Fahrzeug abbiegen

möchte. Im Gegensatz zum Zellularautomaten besitzt das Warteschlangenmodell kein Fahrzeugfolgemodell. Es ist also nicht entscheidend, dass Fahrzeuge auf ein Hindernis, in diesem Fall die Lichtsignalanlage in Form des virtuellen Fahrzeuges, auffahren und langsam abbremsen. Entscheidend beim Warteschlangenmodell ist, ob das Fahrzeug alle Bedingungen zum Überqueren des Knotens erfüllt und zusätzlich gerade grün geschaltet ist. Aus diesem Grund wird der Ansatz mit dem Einfügen eines Fahrzeuges nicht übernommen.

Es wird stattdessen für jede *LaneStorageQueue* geprüft, ob für diese eine Signalgruppe grün geschaltet ist. Wenn ja, so wird anschließend geprüft, ob eine der grün geschalteten Signalgruppen für den Fahrtwunsch des ersten Fahrzeuges zuständig ist. Ist dies der Fall und sind alle Bedingungen zum Überqueren des Knotens erfüllt, wird es über den Knoten gesetzt. Ist dies nicht der Fall, verbleibt es in der *LaneStorageQueue*. *LaneStorageQueue*s, für die alle Signalgruppen rot geschaltet sind, bleiben unberührt.

Im Folgenden werden zwei Varianten vorgestellt, mit denen die Lichtsignalanlage modelliert und umgesetzt wurde.

Variante I Jeder signalisierte Knotenpunkt verfügt über einen Controler, der die Lichtsignalanlagen steuert. Bei ihm sind die Informationen zu den Signalzeitenplänen hinterlegt. Während des Iterierens über alle Knoten werden für jede an den Knoten als *inLink* angeschlossene Kante deren *LaneFlowQueue*s geprüft. Sind diese grün geschaltet, werden Fahrzeuge gemäß Unterabschnitt 3.4.4 über den Knoten gesetzt, andernfalls bleibt die *LaneFlowQueue* unberührt. Von Seiten der Kante ist keine Aktivität nötig. Die Informationen zum Schaltzustand der relevanten Signalgruppen werden vom Controler abgefragt. Das Abgreifen und Übergeben der Fahrzeuge von bzw. zur Kante erfolgt über die gleichen Schnittstellen. Dort werden auch die Events generiert.

Variante II Erste Resultate bei den Tests haben gezeigt, dass die Kapazität einer *LaneFlowQueue* vom Verhältnis von Freigabe- zu Umlaufzeit abhängt. Die genauen Vorgänge und Gründe hierfür sind in Kapitel 5 dargelegt. Für die Verwendung der Simulation in der Forschung sind interpretierbare Ergebnisse entscheidend. Dazu muss der Verkehrsfluss Regeln folgen, die vorhersagbare Ergebnisse liefern und gegebenenfalls nachgerechnet werden können. Dazu benötigt man einen Verkehrsfluss in Abhängigkeit vom relativen und nicht vom absoluten Grünanteil.

In der Variante II ist die Kante nicht mehr vollständig von der Lichtsignalanlage isoliert. Während des Iterierens über die Knoten wird für jeden *LaneLink* vermerkt, ob

in diesem Zeitschritt eine für ihn relevante Signalgruppe grün geschaltet ist. In der anschließenden Iteration über die Kanten ist dann bekannt, ob für den *LaneLink* in diesem Zeitschritt grün oder rot geschaltet ist. Davon abhängig wird die verbliebene Restkapazität der *LaneFlowCapacity* weiter aufsummiert oder nicht. Ist gerade grün geschaltet könnten Fahrzeuge fahren und die Restkapazität erhöht sich. Andernfalls verbleibt sie auf dem alten Wert. Kanten ohne Lichtsignalanlage werden automatisch in jedem Zeitschritt als grün geschaltet vermerkt, so dass sich der Verkehrsfluss im Vergleich zur alten Implementierung nicht ändert.

3.4.4 Knotenpunktlogik

Die Vorgänge auf dem Knotenpunkt sind abhängig von der Art des Knotenpunktes. Es wird zwischen den beiden Fällen unsignalisiert und signalisiert unterschieden. Ist der Knoten signalisiert so verfügt er über einen die Lichtsignalanlagen steuernden Controler. Im unsignalisierten Fall dagegen existiert dieser nicht und es wird das Verhalten der bisherigen Implementierung wie in Unterabschnitt 2.2.1 beschrieben übernommen.

Wird ein Knoten durch eine Lichtsignalanlage gesteuert, so tritt der signalisierte Fall in Kraft. In diesem Fall ist die Kapazität der *inLink*s für die Reihenfolge der Bearbeitung unerheblich, da das Modell lediglich vollständig gesicherte Knotenpunkte modellieren kann. Fahrzeuge aus *LaneLink*s, die grün geschaltet sind, dürfen den Knotenpunkt bei vorhandener Kapazität auf der Zielkante passieren. Es ist Aufgabe des Signalzeitenplans darauf zu achten, dass Zwischenzeiten eingehalten und die unverträglichen Verkehrsströme konfliktfrei geführt werden. Ausnahmen bilden die beiden Fälle des bedingt verträglichen Linksabbiegers und des bei Rot rechts abbiegenden Fahrzeuges. Diese können ungehindert den Knotenpunkt passieren, da kein Modell implementiert ist, welches Konflikte auf dem Knotenpunkt auflöst oder das Fahrerverhalten zum Beispiel zum Finden einer geeigneten Lücke abbildet.

Wie im unsignalisierten Fall ist es Aufgabe der Kanten, die benötigten *LinkLeave*- und *LinkEnter*-Events zu schreiben.

3.4.5 Implementierung eines Wendevorgangs

Unter einem Wendevorgang versteht man das Ändern der Fahrtrichtung, um anschließend in entgegengesetzter Richtung die Fahrt fortzusetzen. Dieses Fahrmanöver wird zum Beispiel benötigt, wenn ein Agent auf einer Kante startet aber sein Ziel in entgegengesetzter Richtung liegt.

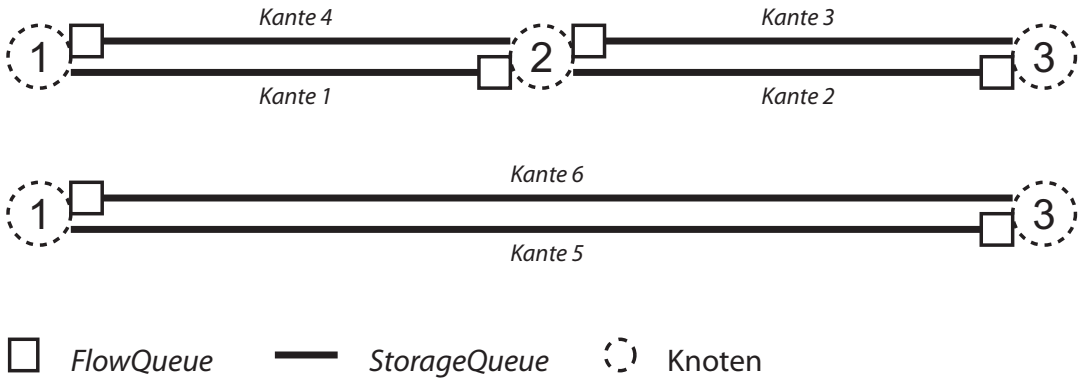


Abbildung 12: Typische Vereinfachung eines Verkehrsnetzes
Durch den Wegfall der Wendemöglichkeit an Knoten 2 verlängert sich die zu fahrende Route von Kante 1 nach Kante 4.

Die Kanten in MATSim sind unidirektional. Somit ist das Wenden auf einer einzelnen Kante nicht möglich, sondern kann immer nur an einem Knoten erfolgen. Deshalb wird Wenden hier als der Wechsel zwischen zwei entgegengesetzt und parallel verlaufenden Kanten definiert. Besitzt eine Kante eine zu ihr entgegengesetzte Kante, so ist ein Wendevorgang möglich. Beide Kanten verbinden dabei die gleichen Knoten nur in umgekehrter Richtung. In Abbildung 12 verbinden die Kanten 1 und 4 beide die gleichen Knoten 1 und 2. Es ist daher zum Beispiel möglich an Knoten 2 von Kante 1 auf Kante 4 zu wenden.

Viele Netzwerke werden zwecks Steigerung der Rechengeschwindigkeit stark reduziert. Dafür werden sekundäre Kanten wie Nebenstraßen gelöscht und anschließend Durchgangsknoten wie Knoten 2 in Abbildung 12 entfernt. Die zu- und abführenden Kanten 1 und 2 werden dabei durch eine äquivalente durchgehende Kante 5 ersetzt. Gleiches geschieht mit der Gegenrichtung. Als Folge entfällt eine Wendemöglichkeit am Knoten 2. Fahrzeuge mit der Startkante 1 und Kante 4 als Ziel sind jetzt gezwungen, die komplette *StorageQueue* der neu entstandenen Kante 6 zu durchfahren. Bisher wurden sie gleich bei Verlassen der *FlowQueue* von Kante 1 in die *StorageQueue* von Kante 4 gesetzt. Sonst gleiche Eigenschaften vorausgesetzt, verdoppelt sich dadurch die zu fahrende Strecke und damit die Fahrtzeit. Ist im Folgenden das Wenden an Knoten 3 nicht möglich, so muss eine Alternativroute gefunden werden, welche die Fahrtzeit weiter erhöht. Diese ist in Abbildung 12 nicht dargestellt. Daher ist es unerlässlich, dass eine Möglichkeit zum Wenden vorgesehen wird.

Wie in Abbildung 4 zu sehen ist das Abbiegen am Knotenpunkt im alten Modell von jeder Kante zu jeder anderen Kante möglich. Ist der Knotenpunkt dagegen signalisiert,

wie es im Subnetzmodell der Fall ist, bestimmt der Signalzeitenplan mit seinen Signalgruppen die Abbiegemöglichkeiten. Gesonderte Signalgruppen für das Wenden existieren jedoch selten und die implizierte Wendemöglichkeit des alten Modells entfällt. Demzufolge muss das Subnetzmodell erweitert werden, um das Wenden wieder zu ermöglichen. Im folgenden Abschnitt wird die in dieser Arbeit verwendete Erweiterung des Modells beschrieben.

Spezifikation

In der bisherigen Implementierung besteht kein Unterschied zwischen einem Wendevorgang und einem konventionellen Abbiegen. In beiden Fällen wird ein Fahrzeug von einem *inLink* auf einen *outLink* transferiert. Die *outLink*s sind alle gleichwertig und werden auch gleich behandelt. Nimmt man jetzt Lichtsignalanlagen hinzu, so sind die einzelnen Abbiegebeziehungen gegebenenfalls unterschiedlich reglementiert. Insbesondere fehlt in den meisten Fällen eine gesonderte Signalisierung für die wendenden Fahrzeuge. Wird das Wenden nicht verboten, so nutzen Fahrzeuge in der Realität normalerweise den äußersten linken Fahrstreifen. Ist keine Linksabbiegerspur vorhanden kann dies auch eine Geradeausspur sein.

Für die Simulation bedeutet dies folgendes: In den Eingangsdaten für die Lichtsignalanlagen wird definiert, welche Signalgruppe welche Abbiegebeziehung bzw. *inLink-outLink* Kombination steuert. Gesonderte Informationen für einen Wendevorgang liegen jedoch nicht vor. Auch sind keine Informationen darüber vorhanden, von welcher Spur aus gewendet werden kann oder in welche Spur hin gewendet wird.

In welchen *outLink* gewendet wird, kann entweder daraus geschlossen werden, dass keine Signalgruppe für die entsprechende *inLink-outLink* Kombination vorliegt oder aber, was realistischer ist, aus der Topographie des Netzwerkes.

Für die Suche nach der Fahrspur, von der aus gewendet wird, gibt es im Wesentlichen zwei Ansätze. Der erste führt eine neue Fahrspur ein. Alle wendenden Fahrzeuge werden auf einen zusätzlichen *LaneLink* umgeleitet. Für diesen wird entweder permanent grün signalisiert oder es wird die Signalisierung der äußersten linken Fahrspur übernommen. Alternativ nutzen die wendenden Fahrzeuge normal die äußerste linke Spur mit und werden entsprechend auch über deren Signalgeber gesteuert. Diese Lösung lehnt sich an die Realität an, da in den meisten Fällen eine gesonderte Wendemöglichkeit nicht vorgesehen ist und wendende Fahrzeuge den übrigen Verkehr beeinflussen können. Sie wird daher gegenüber der ersten Lösung präferiert.

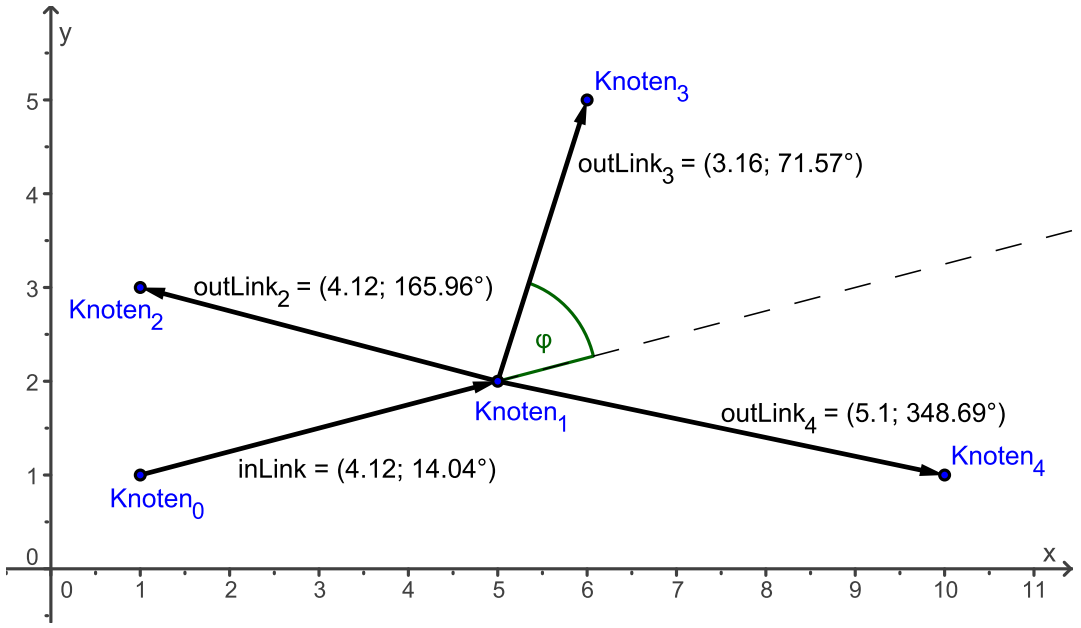


Abbildung 13: Differenzwinkelberechnung zwischen *inLink* und *outLink* Exemplarischer Netzausschnitt mit einem *inLink* und vier *outLink*s. Der *outLink* 1 ist die zum *inLink* gegenläufige Kante.

Es wird bei beiden Ansätzen vorausgesetzt, dass bekannt ist, welche der Spuren sich am äußersten linken Fahrbahnrand befindet. Dies geht aus den Signalzeitenplänen jedoch nicht hervor. Als Alternative kann die Information aus der Netzwerktopographie abgeschätzt werden. Jeder Knoten kann über seine kartesischen Koordinaten lokalisiert werden. Da die Kanten immer zwischen zwei Knoten verlaufen, können diese implizit auch als Vektoren dargestellt werden. Alle vom Knoten abgehenden *outLink*s werden als von diesem Knoten ausgehende Vektoren abgebildet, siehe dazu *outLink* 1 bis 4 in Abbildung 13. Von Interesse ist der Winkel φ zwischen dem *inLink* und dem jeweiligen *outLink*. Im Folgenden wird erläutert wie dieser Winkel zwischen zwei Vektoren berechnet werden kann.

Die Vektoren zweier Kanten berechnen sich direkt aus den kartesischen Koordinaten der Knoten zu:

$$\vec{Kante} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_{Zielknoten} \\ y_{Zielknoten} \end{pmatrix} - \begin{pmatrix} x_{Startknoten} \\ y_{Startknoten} \end{pmatrix} \tag{11}$$

Zur weiteren Vereinfachung der Berechnung werden die Vektoren in Polarkoordinaten überführt. Dabei muss je nach Vorzeichen eine Fallunterscheidung für den Winkel θ erfolgen:

$$r = \sqrt{x^2 + y^2} \tag{12}$$

$$\theta = \begin{cases} \arctan \frac{y}{x} & \text{für } x > 0 \\ \arctan \frac{y}{x} + \pi & \text{für } x < 0,\ y \geq 0 \\ \arctan \frac{y}{x} - \pi & \text{für } x < 0,\ y < 0 \\ +\pi/2 & \text{für } x = 0,\ y > 0 \\ -\pi/2 & \text{für } x = 0,\ y < 0 \end{cases} \tag{13}$$

Der Radius r wird für die Berechnung des Winkels zwischen *inLink* und *outLink* nicht gebraucht. Man erhält den Winkel φ durch Bildung der Differenz der θ zweier Kanten:

$$\varphi = \theta_{outLink} - \theta_{inLink} \tag{14}$$

Für φ muss anschließend noch eine Fallunterscheidung vorgenommen werden. Ist $\varphi < -\pi$ muss 2π addiert werden. Ist $\varphi > \pi$ muss 2π subtrahiert werden. Als Ergebnis erhält man $\varphi \in]-\pi : \pi]$. Gesucht wird nun diejenige Kante mit dem größten φ. Ausnahme bildet hierbei der zum *inLink* entgegengesetzte *outLink*. Dieser wird bei der Berechnung ausgeklammert.

Dieser Algorithmus greift nicht, wenn die in der Realität am weitesten nach links abbiegende Spur nicht auch nach links führt. Dies kann der Fall sein, wenn zwar links abgebogen wird, im Anschluss aber gleich zum Beispiel mittels einer Überführung die Fahrbahn wiederum überquert wird, so dass letztendlich rechts abgebogen wurde. Die Situation ist in Abbildung 14 veranschaulicht. Vom Knoten A aus betrachtet analysiert der Algorithmus die beiden ausgehenden Kanten 201 und 301. Da Kante 301 zum Knoten C führt und dieser von Kante 100 aus betrachtet weiter „links“ liegt, wird folgerichtig geschlossen, dass dann Fahrspur 1 auch die am weitesten links liegende ist. In der Simulation würden die wendenden Fahrzeuge dann die Geradeausspur 1 als vermeintliche Abbiegespur zum Wenden nutzen. Dies ist offensichtlich nicht richtig.

Der Einfluss des hier geschilderten Falls auf das Simulationsergebnis wird als gering eingeschätzt, da eine Verkettung von als selten eingestuften Ereignissen auftreten muss. Die Problematik taucht lediglich in Situationen auf, in denen ein Fahrzeug wenden will und gleichzeitig die Topographie die beschriebene Besonderheit aufweist.

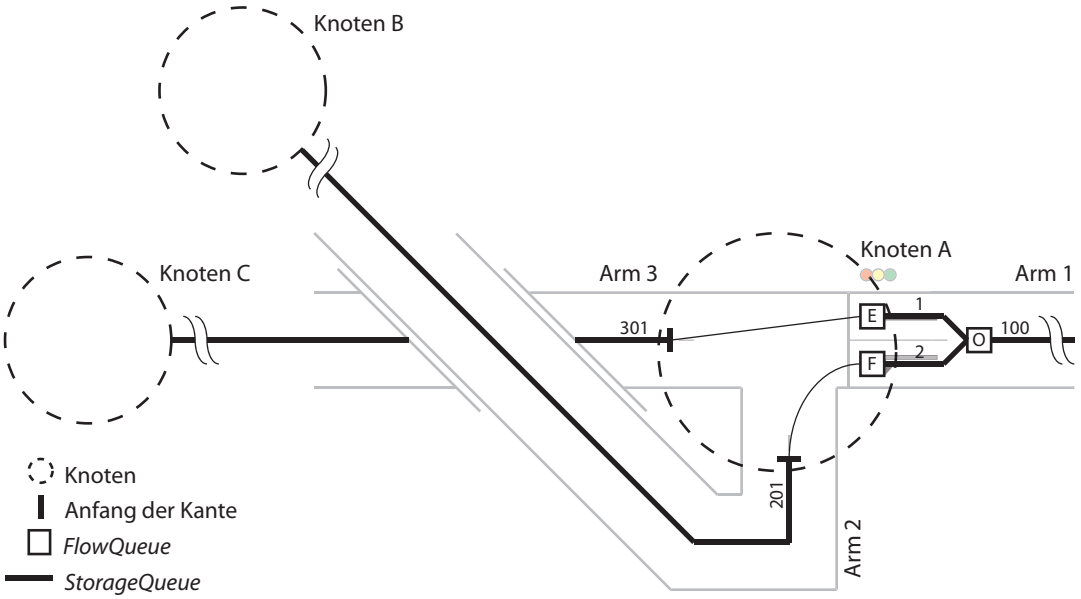


Abbildung 14: Spezialfall beim Abbiegen
In diesem Fall führt die an Knoten A weitesten links liegende Fahrspur nicht zu dem Algorithmus zu folge links liegenden Knoten C, sondern zum Knoten B.

Weiterhin ist zu beachten, dass das angegebene Verfahren ein kartesisches Koordinatensystem voraussetzt, die Oberfläche der Erde aber mit Kugelkoordinaten beschrieben wird. Die Lösung ist daher nur eine Näherungslösung aber für ein Land wie die Schweiz hinreichend genau.

3.5 Erweiterung des Subnetzmodells

An dieser Stelle soll eine Erweiterung des Modells und sich daraus ergebende Anwendungsmöglichkeiten vorgestellt werden. Das bisherige Subnetzmodell besitzt den Nachteil, dass die Abbiegekanten jeweils an einem gemeinsamen Punkt von der konventionellen Kante abzweigen. Zwar ließen sich Länge und damit *StorageCapacity* für jede Abbiegekante anpassen, jedoch ändert sich damit die Länge der gesamten Kante und der Entscheidungspunkt für das Abbiegen bleibt für alle Fahrstreifen gleich. Dies hat Auswirkungen auf die Verkehrsdynamik und dort vor allem auf Rückstaueffekte. Das erweiterte Subnetzmodell baut auf dem vorhergehenden auf und erweitert es um die Möglichkeit verschieden langer Abbiegekanten.

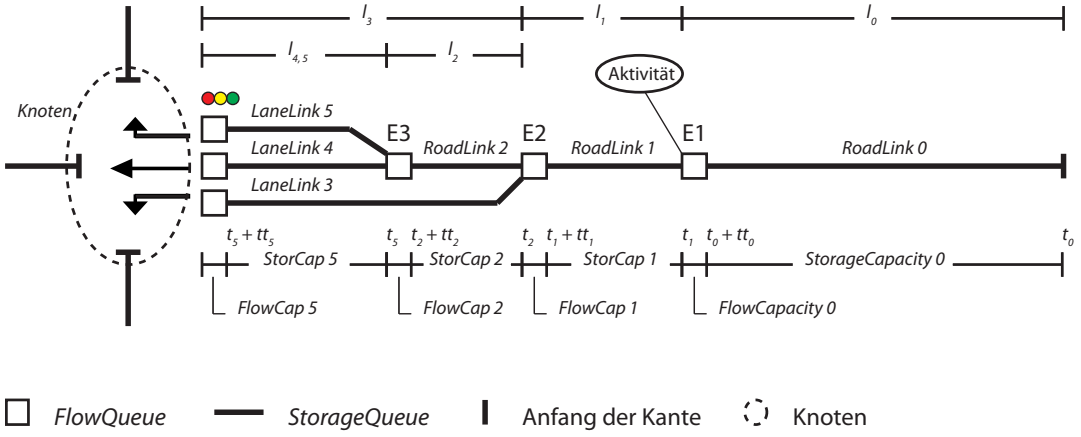


Abbildung 15: Interner Aufbau einer Kante im erweiterten Subnetzmodell
Dieses Modell erweitert das Subnetzmodell um die Möglichkeit weitere Entscheidungspunkte zu definieren.

3.5.1 Spezifikation

Im weiteren Verlauf ist es Ziel, das Abbiegen an räumlich verschiedenen Punkten von der konventionellen Kante zu ermöglichen. Dies setzt eine Lokalisierbarkeit der Fahrzeuge auf der Kante voraus. Eine Lokalisierung über den Zeitstempel alleine ignoriert Restriktionen der Queue. Befindet sich ein Fahrzeug in der *StorageQueue* und lässt dessen Zeitstempel einen Transfer in die *FlowQueue* zu, so kann dieses dennoch durch einen Rückstau behindert werden, da andere Fahrzeuge sich vor ihm in der *StorageQueue* aufhalten und ebenfalls auf einen Transfer warten. Es gilt weiterhin ein Überholverbot, so dass der Fahrtwunsch keinen Einfluss auf die Behandlung der Fahrzeuge hat. Nachfolgend wird vorausgesetzt, dass die Positionen aller möglichen Abbiegepunkte (Entscheidungspunkte) auf der Kante definiert wurden, zum Beispiel in Metern vom Linkende aus betrachtet, und somit bekannt sind.

Wie in Abbildung 15 zu sehen, wird die ursprüngliche Kante wie vorher auch durch ein virtuelles Netzwerk ersetzt. Neu sind die Entscheidungspunkte E1, E2 und E3. Der Entscheidungspunkt E3, vor *LaneLink* 4 und 5 gelegen, befindet sich $l_{4,5}$ Meter vom Ende der Kante entfernt. Der Entscheidungspunkt E2 für *LaneLink* 3 liegt dagegen bei l_3 Metern. Dazwischen wird eine neue virtuelle Kante eingefügt, *RoadLink* 2. Diese besitzt eine eigene *FlowQueue* und *StorageQueue*. Die *FlowCapacity* wird von *RoadLink* 0 übernommen. Die *StorageCapacity* ergibt sich aus der Länge l_2 zwischen den Entscheidungspunkten E2 und E3 sowie aus der Anzahl der Spuren der ursprünglichen Kante. Dieser Wert kann von *RoadLink* 0 übernommen werden.

Fahrzeuge, welche die Kante betreten werden weiterhin in die erste *StorageQueue* gesetzt (*RoadLink* 0) und durchlaufen diese wie gewohnt. Fahrten, deren Anfang oder Ende sich auf dem Link befindet, beginnen in der *FlowQueue* bzw. enden am Ende der *StorageQueue* von *RoadLink* 0. Die *FreeLinkTravelTime* der veränderten und neuen Links ist so bemessen, dass sie jeweils deren Länge, also der Länge zwischen zwei Entscheidungspunkten oder zwischen Entscheidungspunkt und Ende der Kante entspricht.

Durch die Hinzunahme der Entscheidungspunkte muss jedes Mal geprüft werden, welche der stromab nachfolgenden Links zu der vom Fahrzeug gewünschten nächsten echten Kante führt. Dazu muss für jeden Link vorgehalten werden, welche Kanten über ihn erreicht werden können. Im bisherigen Modell waren die folgenden Kanten direkt zu erreichen. Jetzt können abhängig von der Topographie des Knotenpunktes ein oder mehrere Links dazwischen liegen.

Für das Iterieren über die Knoten ändert sich nichts. Während des Iterierens über die Kanten sind jetzt mehrere Links zu durchlaufen. Auch deren Reihenfolge ist entscheidend. Begonnen wird mit den Links am Ende der Kante, den *LaneLink*s 3 bis 5 in Abbildung 15. Die *RoadLink*s folgen entsprechend ihrer Distanz zum Knotenpunkt, also der Strecke zwischen dem nächsten Entscheidungspunkt und dem Ende der nach außen sichtbaren Kante. Für *RoadLink* 2 beträgt diese Strecke $l_{4,5}$ Meter. Sie liegt damit vor *RoadLink* 1. Die Reihenfolge der verbliebenen Links ist demnach *RoadLink* 2, *RoadLink* 1 und *RoadLink* 0.

Ist die *StorageCapacity* eines Links bereits erreicht, verbleibt das Fahrzeug in der *FlowQueue* des vorangegangenen Links und staut den nachfolgenden Verkehr. Als Folge sind Überstauungen von Entscheidungspunkten möglich. Ist die *LaneStorageCapacity* von *LaneLink* 5 erreicht, beginnt sich *RoadLink* 2 zu stauen. Im weiteren Verlauf wird die *StorageCapacity* von *RoadLink* 2 erreicht und der Entscheidungspunkt E2 am Ende von *RoadLink* 1 wird ebenfalls überstaut. Damit werden die Linksabbieger auf *LaneLink* 3 von den rechts abbiegenden Fahrzeugen auf *LaneLink* 5 behindert.

Der Unterschied zum vorangegangenen nicht erweiterten Subnetzmodell liegt darin, dass nicht alle Abbiegebeziehungen gleichzeitig gestaut werden, sondern dies abhängig von den Positionen der Entscheidungspunkte und dem bereits aufgebauten Rückstau geschieht.

Das Modell lässt sich erweitern, indem man neben den Abbiegekanten auch Aktivitäten auf der Kante an Entscheidungspunkten verankert. Vorteil ist die feste Lokalisierung der Aktivität auf der Kante. Insbesondere auf langen Kanten ist es entscheidend,

ob sich eine Aktivität an deren Anfang, Ende oder in der Mitte befindet. Entsprechend ändern sich die Fahrtzeiten und die Attraktivität einer Aktivität.

Nachteilig ist die Beeinflussung des Verkehrsflusses auf der Kante durch die Anzahl der Aktivitäten. Diese skaliert entsprechend der Aktivitätenanzahl. Die Anzahl der Aktivitäten ist begrenzt durch die räumliche Länge der Kante. Bei einer angenommenen Fahrzeuglänge von 7,5 m sind maximal 13 Entscheidungspunkte auf einer 100 m langen Kante möglich, da andernfalls die minimale *StorageCapacity* von einem Fahrzeug je *StorageQueue* nicht gewährleistet ist. Eine Aneinanderreihung von nur ein Fahrzeug fassenden *StorageQueue*s kommt einem einfachen Zellularautomaten gleich und revidiert den Geschwindigkeitsvorteil des Warteschlangenmodells. Lösen kann man dies durch das Zusammenlegen von räumlich dicht beieinander liegenden Aktivitäten bzw. Entscheidungspunkten. Aber auch hier wird der Verkehrsfluss beeinflusst. Werden die Queues in entgegengesetzter Richtung, also vom Anfang der Kante beginnend, bearbeitet, so wandert eine während eines Staus am Ende der Kante entstehende Lücke zeitlich leicht verzögert zu deren Anfang. Die Verzögerung ist abhängig von der Anzahl der Entscheidungspunkte und sollte je Entscheidungspunkt einen Simulationsschritt betragen.

3.5.2 Auswirkungen

Vorteile Vorteile sind insbesondere bei der Behandlung der Aktivitäten zu erwarten. Aktivitäten können räumlich differenziert auf der Kante verankert werden. Dabei müssen dies nicht zwangsläufig die Aktivitäten wie Arbeit oder Wohnen sein. Grundsätzlich lassen sich auch andere Vorgänge an den Entscheidungspunkten verankern. Zum Beispiel lassen sich bei einer zukünftigen Implementierung eines öffentlichen Personennahverkehrs die Entscheidungspunkte als Haltestellen nutzen.

Nachteile Hier ist vor allem ein wiederum erhöhter Aufwand bei der Implementierung zu nennen. Des Weiteren müssen die genauen Auswirkungen auf den Verkehrsfluss getestet werden, was innerhalb dieser Arbeit nicht geschieht.

Fazit Die Erweiterung des Subnetzmodells zeigt, dass es sich für künftige Anforderungen erweitern lässt. Insbesondere dort, wo die dafür benötigten Daten vorliegen, kann die lokale Genauigkeit erhöht werden.

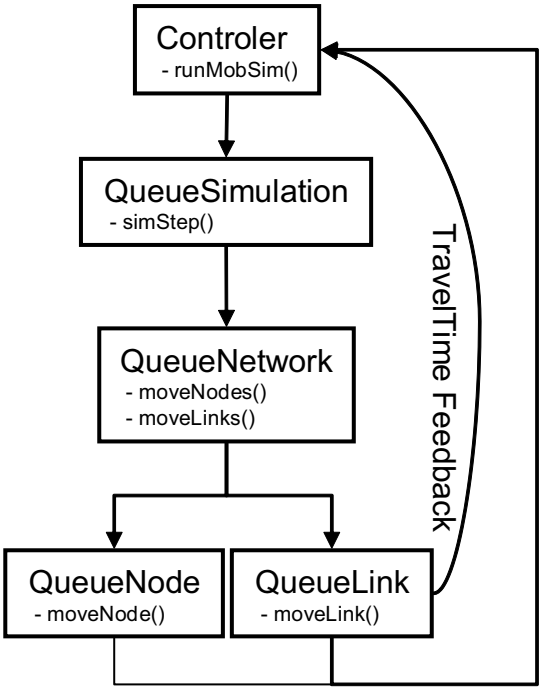


Abbildung 16: Grundaufbau der derzeitigen Implementierung von MATSim
In MATSim werden zuerst die Knoten und anschließend die Kanten aktualisiert. Dies geschieht durch Aufruf von `moveNodes()` bzw. `moveLinks()` und wird koordiniert durch das `QueueNetwork`. Dieses ist Bestandteil der `QueueSimulation`, welche wiederum durch den `Controler` gesteuert wird.

4 Implementierung

Ausgehend vom im vorherigen Kapitel beschriebenen Subnetzmodell und den in Abschnitt 2.2 dargelegten Aufbau von MATSim soll hier erläutert werden, wie das eigene Modell eingebunden werden kann. Dazu wird im ersten Abschnitt beschrieben, wie das bisherige in Java implementierte Verkehrsflussmodell in MATSim integriert ist und von den anderen Modulen angesprochen werden kann. Weiter führende Informationen dazu bietet die Veröffentlichung von Balmer u. a. (2008). Im zweiten Abschnitt folgt dann die Beschreibung der Beispielimplementierung des eigenen Modells.

4.1 Einbindung des Verkehrsflussmodells in MATSim

Der Grundaufbau einer MATSim-Simulation kann der Abbildung 16 entnommen werden. Kern ist die `Controler`-Klasse. Entsprechend der in Abschnitt 2.2 beschriebenen wesentlichen Schritte einer MATSim-Simulation, steuert diese die Vorbereitung, die Iteratio-

nen und die Nachbereitung. Die Ausführung der Verkehrsflusssimulation übernimmt die Klasse `QueueSimulation`. Sie wird über die Methode `runMobSim()` von der `Controler`-Klasse aus gestartet und ruft ihrerseits in jedem Simulationsschritt die `simStep()`-Methode der untergeordneten `QueueNetwork`-Klasse auf.

Die entscheidenden Aufrufe geschehen innerhalb eines `simStep`s. Dort wird zunächst für jeden Knoten (`QueueNode`-Klasse) die zugehörige `moveNode()`-Methode aufgerufen. `moveNode()` ist für die Abläufe auf dem Knoten zuständig. Dazu gehört, dass die Fahrzeuge aus der *FlowQueue* der *inLink*s in die nachfolgende *StorageQueue* des entsprechenden *outLink*s transferiert werden.

Ist der Aufruf von `moveNode()` für alle Knoten abgeschlossen, wird analog dazu für alle Kanten die `moveLink()`-Methode innerhalb der `QueueLink`-Klasse aufgerufen. Dort werden zuerst unter Beachtung der ebenfalls in Unterabschnitt 2.2.1 beschriebenen Restriktionen Fahrzeuge aus der *StorageQueue* in die *FlowQueue* bewegt und anschließend Fahrzeuge aus der *ParkingQueue* in die *FlowQueue*.

Die `QueueNode`-Klasse ist gegenüber der `QueueLink`-Klasse sehr einfach strukturiert, da hier weder eine Knotenpunktlogik für einen signalisierten noch für einen unsignalisierten Knotenpunkt implementiert ist. Auch ist `QueueLink` verantwortlich für das Schreiben der Events, welche für die Planbewertung benötigt werden. Insbesondere die *LinkEnter*- und *LinkLeave*-Events werden durch `QueueLink` und nicht durch `QueueNode` geschrieben, obwohl Fahrzeuge während `moveNode()` die Kante wechseln.

Zusammenfassend werden die folgenden Schritte innerhalb eines Simulationsschrittes durchgeführt:

- Für alle Knoten:
  Leeren der *inLink-FlowQueue* durch `QueueNode.moveNode()` und setzen der Fahrzeuge in die entsprechende *outLink-StorageQueue*

- Für alle Kanten:
  Transferieren von Fahrzeugen aus der *StorageQueue* in die *FlowQueue* durch den Aufruf von `QueueLink.moveLink()`. Anschließend auffüllen der *FlowQueue* durch Fahrzeuge aus der *ParkingQueue*

4.2 Einbindung des eigenen Verkehrsflussmodells in MATSim

Die Implementierung des neuen Modells ersetzt zum einen vorhandene Klassen durch eigene Klassen mit erweiterter Funktionalität und zum anderen werden neue Klassen im bisherigen Programmablauf verankert. Eine Übersicht aller für das Modell benötigten Klassen und deren Interaktion mit dem bisherigen MATSim-Aufbau kann dem Klassendiagramm in Abbildung 17 entnommen werden. Im Folgenden werden die Bestandteile der einzelnen Pakete und ihre Funktion beschrieben.

Paket sim Um eigene Funktionalität hinzufügen zu können, wird der MATSim-`Controler` abgeleitet und die Methode zum Aufruf der Verkehrsflusssimulation, `runMobSim()`, überschrieben. An dieser Stelle wird definiert, welches alternative Verkehrsflussmodell, hier `QSim`, statt der Standardimplementierung `QueueSimulation` eingebunden wird. Entscheidend ist, dass bereits im Constructor die eigene `TrafficLightQueueNetworkFactory` verankert wird, die garantiert, dass ein für die eigene QueueSimulation kompatibles Netzwerk generiert wird. Dadurch erzeugt die `QueueNetwork`-Klasse keine `QueueLink`s und `QueueNode`s mehr, sondern die davon abgeleiteten und erweiterten Klassen `QLink` und `QNode`. `QNode` und `QLink` sind die beiden Hauptklassen und verantwortlich für die Fortbewegung der Agenten auf dem Netz. Dabei übernimmt `QNode` die Physik der Knotenpunkte und `QLink` die Physik der die Knotenpunkte verbindenden Streckenabschnitte. Entsprechend stellen diese Klassen die Methoden `moveNode()` bzw. `moveLink()` zur Verfügung.

`QueueSimulation` stellt die Funktionen `prepareSim()` und `cleanupSim()` zur Verfügung um die Ausführung vorzubereiten und nachzubereiten. `prepareSim()` ist verantwortlich für das Setzen der Start- und Stoppzeiten der Ausführung wie auch der Erzeugung der Agenten aus den Plänen. Darüber hinaus wird in `QSim` die neue Methode `readSignalSystemControler()` aufgerufen. Diese liest die für die Lichtsignalanlagen relevanten Daten ein und erzeugt für jeden Knoten mit Lichtsignalanlage einen LSA-Controler (`SignalSystemControlerImpl`), der die Signalzeitenpläne vorhält. Bis zu diesem Zeitpunkt unterscheidet sich das Netz in seinem Verhalten jedoch nicht von einer Standardimplementierung. Daher werden zusätzlich über den Aufruf `QLink.reconfigure()` die signalisierten Kanten neu konfiguriert. Dieser erzeugt dann entsprechend den in den `SignalSystemControlerImpl` enthaltenen Informationen zu Spuren und Abbiegemöglichkeiten eine abweichende interne Struktur in Form des Subnetzes.

Ferner werden bei jedem Simulationsschritt von `QueueSimulation` bzw. `QSim` die Funktionen `beforeSimStep()`, `doSimStep()` und `afterSimStep()` aufgerufen. Dabei

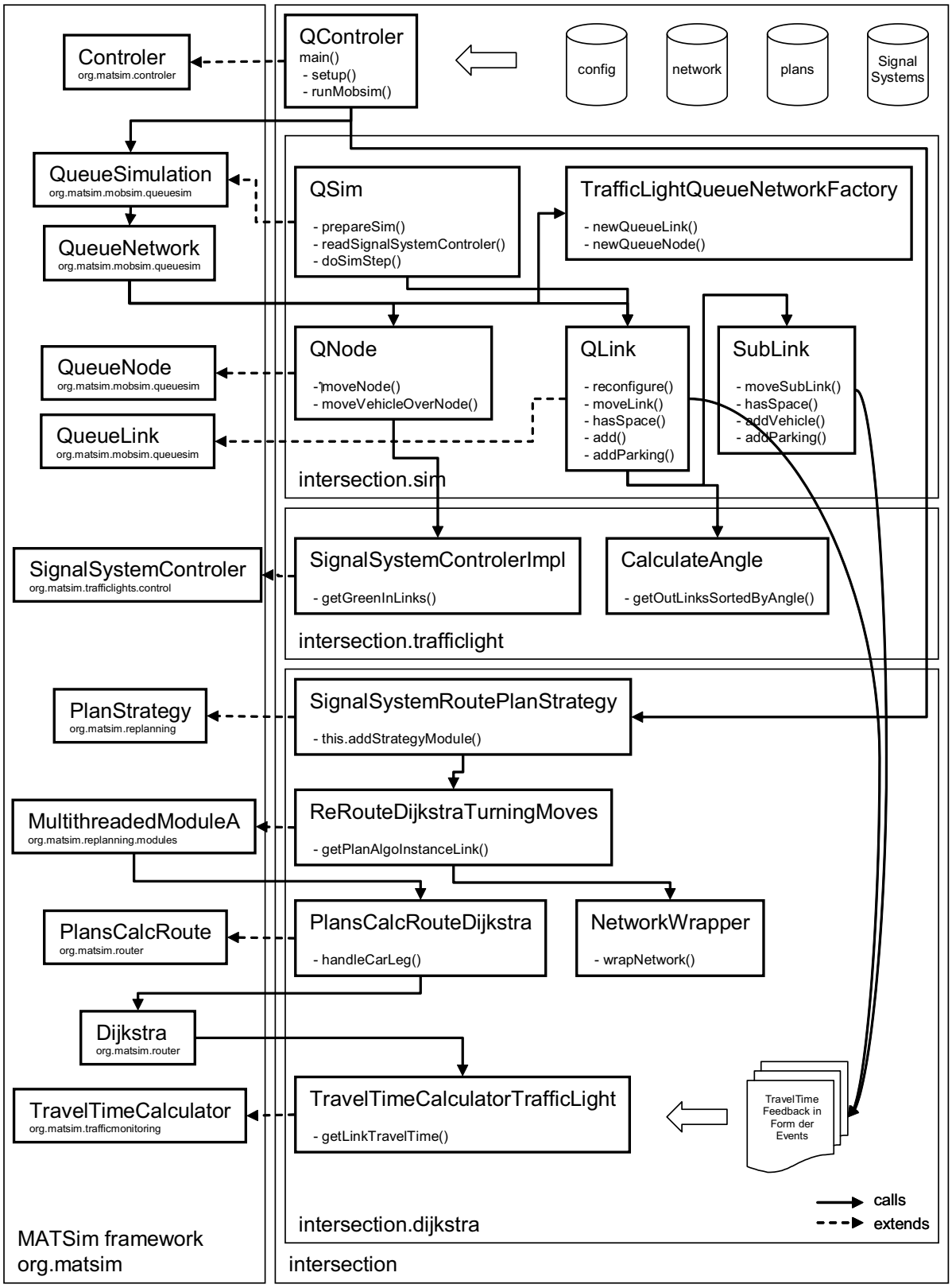


Abbildung 17: Klassendiagramm des nach Kapitel 3 umgesetzten Modells

ruft `doSimStep()` die beiden `QueueNetwork`-Funktionen `moveLinks()` und `moveNodes()` auf. Diese wiederum geben den Aufruf an die bereits erwähnten Methoden `moveLink()` bei den Kanten und `moveNode()` bei den Knoten weiter. Wird `moveLink()` aufgerufen so leitet `QLink` den Aufruf wie in Abschnitt 3.3 beschrieben weiter an alle Links des Subnetzes, welche dann ihre eigene Methode `moveSubLink()` aufrufen.

Paket trafficlight Die Lichtsignalanlage ist vollständig über den Knotenpunkt implementiert. Beim Aufruf von `moveLink()` besteht nach der vollständigen Generierung des Subnetzes seitens der Kanten kein Unterschied zwischen einer Kante mit Lichtsignalanlage an deren Ende und einer ohne LSA. Der Unterschied besteht einzig innerhalb der `QNode`-Klasse. Diese hat Zugriff auf den `SignalSystemControlerImpl` und fragt ihn während `moveNode()` nach den Schaltzuständen der einzelnen Signalgruppen.

Zur Identifikation der für das Wenden relevanten Fahrspur und einer Anordnung der *LaneLink*s entsprechend ihrer Anordnung im Lageplan existiert die Hilfsklasse `CalculateAngle`. Sie wird zu Beginn der Simulation einmal für jeden Link während `QLink.reconfigure()` aufgerufen.

Paket dijkstra Der Kern dieses Paketes ist die Klasse `PlansCalcRouteDijkstra`. Dessen `handleCarLeg()` wird aufgerufen sobald ein Agent eine neue Route sucht, wird also während des Replannings genutzt. Als Replanning-Modul verankert ist sie über die beiden Klassen `SignalSystemRoutePlanStrategy` und `ReRouteDijkstraTurningMoves`. `ReRouteDijkstraTurningMoves` stellt auch das für den Aufruf des Dijkstra-Algorithmus benötigte invertierte Netzwerk über die Hilfsklasse `NetworkWrapper` bereit.

Da der Dijkstra-Algorithmus neben dem Netzwerk auch Kosten zu den Kanten benötigt, sammelt `TravelTimeCalculatorTrafficLight` die relevanten Events während der Ausführung, bereitet diese auf und stellt sie dem Dijkstra für das Replanning zur Verfügung.

5 Resultate

In diesem Kapitel soll das implementierte Subnetzmodell auf seine Funktionstüchtigkeit getestet werden. Dazu werden die in Abschnitt 2.3 vorgestellten Bewertungen anhand der Knotenpunktkapazität und -qualität auf einfache Netzwerke angewendet und die dabei erzielten Ergebnisse vorgestellt. Zusätzlich wird geprüft, ob die in Kapitel 3 formulierten Anforderungen vom Modell erfüllt werden. Begonnen wird mit einem möglichst einfachen, allgemein gültigen Netz, welches MATSim darzustellen vermag. Anschließend werden die generierten Ergebnisse mit denen der alten Implementierung verglichen. Den Abschluss bilden Tests zum Replanning-Prozess und die Simulation eines an einen Planungsentwurf angelehnten kompletten Knotenpunktes.

5.1 Zweiarmiger Knotenpunkt

Der zweiarmige Knotenpunkt besteht aus einer zuführenden und einer wegführenden Kante. Er stellt den einfachsten sinnvollen Fall der Anwendung einer Lichtsignalanlage dar. Ein Beispiel aus der Realität ist ein mit einer Lichtsignalanlage gesicherter Fußgängerüberweg.

Aufbau

Betrachtet man einen Knotenpunkt als ein abstraktes Gebilde, so wird der einfachste Fall durch einen einarmigen Knotenpunkt dargestellt oder, allgemein ausgedrückt, durch eine Sackgasse. In Unterabschnitt 3.4.2 wurde bereits angesprochen, dass die Kanten in MATSim unidirektional sind und jede Kante eines Netzwerkes von jeder anderen Kante aus erreichbar sein sollte, es aber nicht sein muss. Als Folge besteht zwar die Möglichkeit in Sackgassen endende Einbahnstraßen in MATSim darzustellen, dies sollte aber für einen allgemein gültigen Testfall vermieden werden. Anders ausgedrückt verfügt im Modell eines realen Netzes jeder Knoten über mindestens einen *outLink* und einen *inLink*. Bezogen auf den abstrakten Knotenpunkt stellt der zweiarmige Knotenpunkt den für Testfälle anwendbaren Minimalfall dar. Ein solcher Knotenpunkt ist in Abbildung 18 der Knoten 2. Kante 1 ist dabei der *inLink* und Kante 2 der *outLink*.

Der in Abbildung 18 dargestellte Knoten 2 wurde zusätzlich um die Kanten 3, 4 und 5 ergänzt. Kante 4 ist für den Test nicht zwingend erforderlich, sorgt aber dafür, dass jede Kante von jeder anderen aus erreicht werden kann. Lediglich Knoten 2 ist mit einer Lichtsignalanlage ausgestattet. Bei den Knoten 1, 3, 4 und 5 handelt es sich um konventionelle Knoten ohne LSA. Die eigentliche Teststrecke besteht aus den unidirektionalen

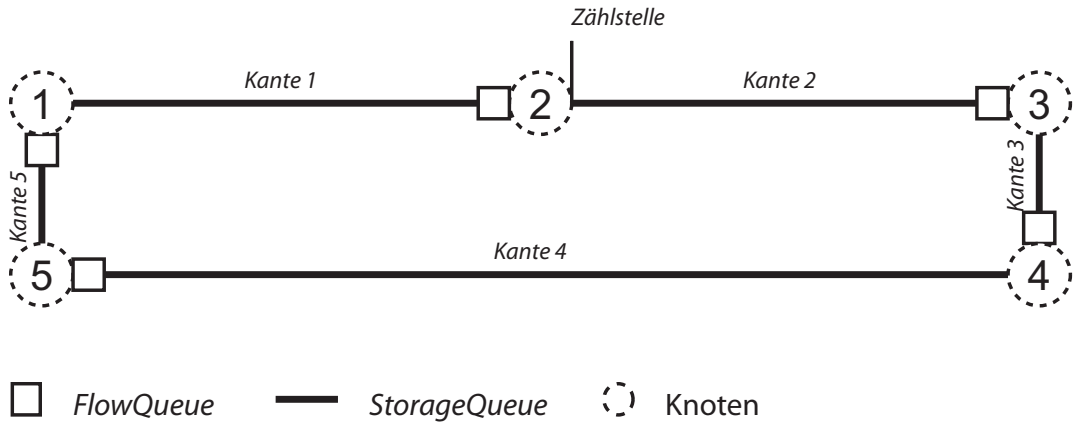


Abbildung 18: Kapazitätstest an einem zweiarmigen Knotenpunkt - Aufbau
Das Netz besteht aus 5 Knoten und 5 Kanten, verknüpft zu einem unidirektionalen Kreis. Agenten starten auf Kante 5 und haben Kante 3 zum Ziel.

Kanten 1 und 2 unterbrochen durch die Lichtsignalanlage an Knoten 2.

Für den Test wurden 5000 Agenten mit 5000 Fahrzeugen generiert. Diese haben Kante 5 als Start- und Kante 3 als Zielkante. Alle Fahrzeuge haben die gleiche Abfahrtzeit, so dass alle zu Beginn der Simulation starten. Kante 1 und 5 verfügen über eine *FlowCapacity* von 2000 Fz/h. Die restlichen Kanten besitzen eine leicht erhöhte *FlowCapacity* von 2500 Fz/h. Damit kann für die Dauer einer Stunde ein permanenter Zustrom von 2000 Fz/h am Knoten 2 garantiert werden. Des Weiteren ist ein ungehinderter Abfluss über Kante 2 möglich.

Resultate

In einem ersten Schritt konnte gezeigt werden, dass alle Fahrzeuge ihr Ziel erreichen. Es konnte ferner gezeigt werden, dass Fahrzeuge sich auf Kante 1 stauen, wenn die *FlowCapacity* von Kante 5 ebenfalls auf 2500 Fz/h erhöht wird, da dann die *FlowCapacity* von Kante 1 geringer ist als die der zuführenden Kante 5.

In einem zweiten Schritt wurde die Kapazität am Knoten 2 untersucht. Die Knotenpunktkapazität wird mit Hilfe einer Zählstelle direkt hinter dem Knoten am Anfang von Kante 2 bestimmt. Dabei wurden die für diese Kante relevanten *LinkEnter*-Events ausgewertet und folgende Werte festgehalten:

- Zeitpunkt des Eintreffens des ersten Agenten zur Ermittlung eines geeigneten Zeitpunktes für den Start der Zählung
- Anzahl der Agenten, die die Kante betreten

- Anzahl der Agenten, die die Kante innerhalb einer Stunde vom Beginn der Zählung an betreten haben

Der Startzeitpunkt der Zählung wurde so gewählt, dass er erstens zu Beginn eines Umlaufs liegt und zweitens ein Stau vor dem Knoten garantiert werden kann. Mit der Zählung wurde demnach circa acht Minuten nach Passieren des ersten Fahrzeuges begonnen.

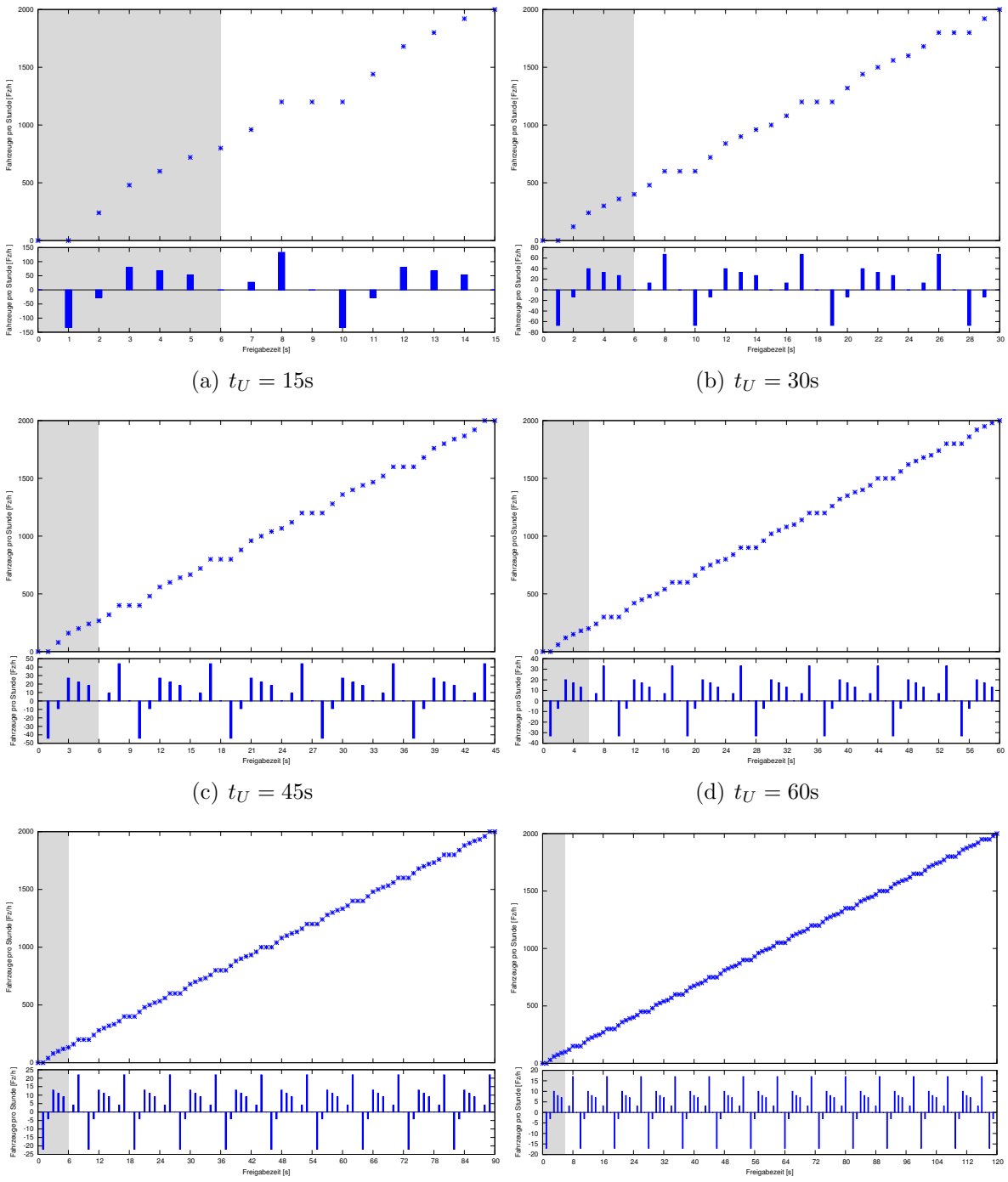
Bei ungehinderter Fahrt über den Knoten 2, $t_F = t_U$, sollten auch die maximal möglichen 2000 Fz/h den Knoten passieren können. Es konnte gezeigt werden, dass 2000 Fahrzeuge innerhalb der ersten Stunden die Zählstelle passieren und die verbliebenen innerhalb der folgenden eineinhalb Stunden. Für $t_F = 0\,\mathrm{s}$ sollte dagegen kein Fahrzeug passieren können. Auch das konnte gezeigt werden.

Der Test wurde im Folgenden erweitert und für verschiedene Umlaufzeiten und Freigabezeiten durchgeführt. Dabei wurden für jede Umlaufzeit alle Freigabezeiten von 1 s bis zum maximal möglichen $t_F = t_U$ simuliert. Die Freigabezeit von 0 Sekunden beschreibt einen Sonderfall, da in diesem Fall kein Fahrzeug den Knotenpunkt passieren kann und somit das hier beschriebene Messverfahren nicht greift. Die Daten für $t_F = 0\,\mathrm{s}$ wurden gesondert durch visuelle Kontrolle während der Simulation ermittelt.

Die Ergebnisse der Tests für eine LSA-Implementierung der Variante I wie in Unterabschnitt 3.4.3 beschrieben sind für alle Umlaufzeiten in Abbildung 19 dargestellt.

Auf der Abszisse ist jeweils die Freigabezeit t_F in Sekunden abgetragen. Deren Maximalwert ist abhängig von der Umlaufzeit t_U. Auf der Ordinate ist der Fluss in Fahrzeugen je Stunde abgetragen. Jeder Graph ist zweigeteilt. Im oberen Abschnitt ist die durch die Zählstelle innerhalb der ersten Stunde ermittelte absolute Anzahl der Fahrzeuge dargestellt. Der untere Graph stellt die Differenz der absolut gezählten Fahrzeuge zu der nach HBS-Rechnung erwarteten Anzahl dar. Die Rechnung nach HBS wurde nach Gleichung 5 auf Seite 27 durchgeführt. Zugrunde liegt dabei ebenfalls eine Kapazität von 2000 Fz/h, was einem Zeitbedarfswert t_B von konstant 1,8 s entspricht. Die graue Fläche kennzeichnet den Bereich der Freigabezeiten von unter 6 Sekunden. Für den motorisierten Individualverkehr empfiehlt das HBS 6 s als minimale Freigabezeit. Zeiten darunter sollten nur in Ausnahmefällen verwendet werden. Die Untersuchungen hier konzentrieren sich daher auf den Bereich ab 6 Sekunden.

Zu erkennen ist, dass für alle Umlaufzeiten bei einer Freigabezeit von 0 Sekunden kein Fahrzeug den Knoten passiert hat. Bei einer maximalen Freigabezeit von $t_F = t_U$ wurde die volle *FlowCapacity* von Kante 1 ausgeschöpft. In Bezug auf die Kapazität verhält sich damit das Modell wie das bisher implementierte ohne Lichtsignalanlage.



(a) $t_U = 15$s

(b) $t_U = 30$s

(c) $t_U = 45$s

(d) $t_U = 60$s

(e) $t_U = 90$s

(f) $t_U = 120$s

Abbildung 19: Knotenpunktkapazität mit LSA-Implementierung der Variante I

Oberer Graph: Anzahl der Fahrzeuge, die innerhalb einer Stunde den Knotenpunkt passiert haben, abgetragen über die Freigabezeit t_F

Unterer Graph: Differenz zwischen der gemessenen Anzahl und der nach HBS berechneten Anzahl der Fahrzeuge (gemessen - berechnet)

Zwischen diesen beiden Extremwerten kommt es immer wieder zu Abweichungen, die sich periodisch wiederholen. Die größte Abweichung liegt für alle Umlaufzeiten bei $t_F = 8\,\mathrm{s}$. Alle anderen Abweichungen sind vom Betrag her gleich groß oder liegen darunter. Die Abweichungen liegen damit abhängig von der Umlaufzeit zwischen $\pm 133\,\mathrm{Fz/h}$ für $t_U = 15\,\mathrm{s}$, d.h. bezogen auf die Kapazität von $2000\,\mathrm{Fz/h}$ $\pm 6{,}65\,\%$ und $\pm 17\,\mathrm{Fz/h}$ für eine Umlaufzeit von $t_U = 120\,\mathrm{s}$, d.h. $\pm 0{,}85\,\%$.

Die Abstufung, die für $t_U = 15\,\mathrm{s}$ zwischen $t_F = 8\,\mathrm{s}$ bis $t_F = 10\,\mathrm{s}$ auftritt, wird verursacht durch die verbliebene Restkapazität eines Zeitschrittes. Da nur ganze Fahrzeuge in einem Zeitschritt bewegt werden und die *FlowCapacity* meist durch eine gebrochen rationale Zahl dargestellt wird, verbleibt in einem Zeitschritt eine Restkapazität. Diese kann nicht sofort genutzt werden. In den nachfolgenden Zeitschritten wird diese Restkapazität aufsummiert bis sich wieder ein ganzes Fahrzeug ergibt. Das Aufsummieren wird dann unterbrochen, wenn die Restkapazität mindestens ein Fahrzeug ergibt.

Durch die Verwendung einer Lichtsignalanlage, wird der Fahrzeugstrom regelmäßig unterbrochen. Das Aufsummieren geschieht jedoch auch während der Rotphase, also in den Zeitschritten in denen eigentlich kein Fahrzeug abfließen kann. Als Folge steht am Anfang der nächsten Grünphase eine Restkapazität größer gleich eins zur Verfügung und im ersten Zeitschritt kann sofort ein Fahrzeug abfließen.

Bei einer Freigabezeit von $t_F = 7\,\mathrm{s}$ können 4 Fahrzeuge je Umlauf passieren, also insgesamt 960 Fahrzeuge. Bei $t_F = 8\,\mathrm{s}$ sind es jedoch bereits 5 Fahrzeuge, da die akkumulierte Restkapazität gerade soweit ausreicht, um in Sekunde 7 ein zusätzliches Fahrzeug passieren zu lassen. In der anschließenden Rotphase steigt die Restkapazität dann wie bei $t_F = 7\,\mathrm{s}$ wieder auf größer gleich eins an. Anschließend wiederholen sich die Werte für die Restkapazität mit jedem Umlauf. Als Folge passieren in jedem weiteren Umlauf 5 Fahrzeuge den Knotenpunkt, also insgesamt 1200 Fahrzeuge in der Stunde.

Für $t_F = 9\,\mathrm{s}$ können ebenfalls konstant 5 Fahrzeuge pro Umlauf, also 1200 Fahrzeuge in der Stunde passieren. Für $t_F = 10\,\mathrm{s}$ tritt der umgekehrte Fall von $t_F = 8\,\mathrm{s}$ auf. Die verbliebene Restkapazität in Sekunde 9 reicht gerade nicht aus um ein weiteres Fahrzeug passieren zu lassen. Die Restkapazität füllt sich in Sekunde 10 sofort wieder auf, jedoch kann dieses Fahrzeug dann nicht mehr in diesem Zeitschritt passieren, sondern muss den nächsten Umlauf abwarten. Anschließend wiederholt sich die Höhe der zu Beginn eines Umlaufs zur Verfügung stehenden Restkapazität. Es können dann in jedem Umlauf 5 Fahrzeuge passieren, also insgesamt wiederum 1200 Fahrzeuge. Die zusätzliche Grünzeit von einer Sekunde wirkt sich hier nicht über eine höhere Restkapazität aus.

Die Abstufungen treten nur bei einem bestimmten Verhältnis von t_F/t_U zu *Flow-Capacity* auf. Die Variante I der LSA-Implementierung hat gezeigt, dass die Ergebnisse abhängig von der Umlaufzeit mehr oder weniger von der Theorie abweichen. Dies hat zu einer leichten Überarbeitung des Modells geführt. Die Ergebnisse der in Unterabschnitt 3.4.3 vorgestellten erweiterten LSA-Implementierung der Variante II werden im Folgenden vorgestellt.

Abbildung 20 zeigt die erzielten Ergebnisse bei Verwendung der Variante II. Diese berücksichtigt den momentanen Schaltzustand der Lichtsignalanlage. Die verbliebene Restkapazität einer Kante wird demnach nur während der Grünphase erhöht. Besitzt ein Knotenpunkt keine Lichtsignalanlage so wird in jedem Zeitschritt geprüft, ob die Restkapazität erhöht werden muss.

Im Vergleich zu Abbildung 19 fällt auf, dass bei der erweiterten Variante II die Abweichungen von der Theorie viel geringer ausfallen. Die Abweichung beträgt hier maximal $\pm 1\,\mathrm{Fz/h}$ und ist unabhängig von der Umlaufzeit. Bezogen auf die Kapazität von $2000\,\mathrm{Fz/h}$ ist dies weniger als ein Promille Abweichung.

5.2 Vergleich mit dem bisher implementierten Modell

Dieser Test soll zeigen, inwieweit die Ergebnisse der Implementierung des Subnetzmodells von denen der bisherigen Implementierung abweichen. Sollten die Ergebnisse identisch sein, so kann das Subnetzmodell nicht nur bei Netzen mit Lichtsignalanlagen verwendet werden, sondern kann auch das bisherige Modell bei Netzen mit ausschließlich unsignalisierten Knotenpunkten ersetzen. Dazu müssen die Ergebnisse jedoch exakt übereinstimmen.

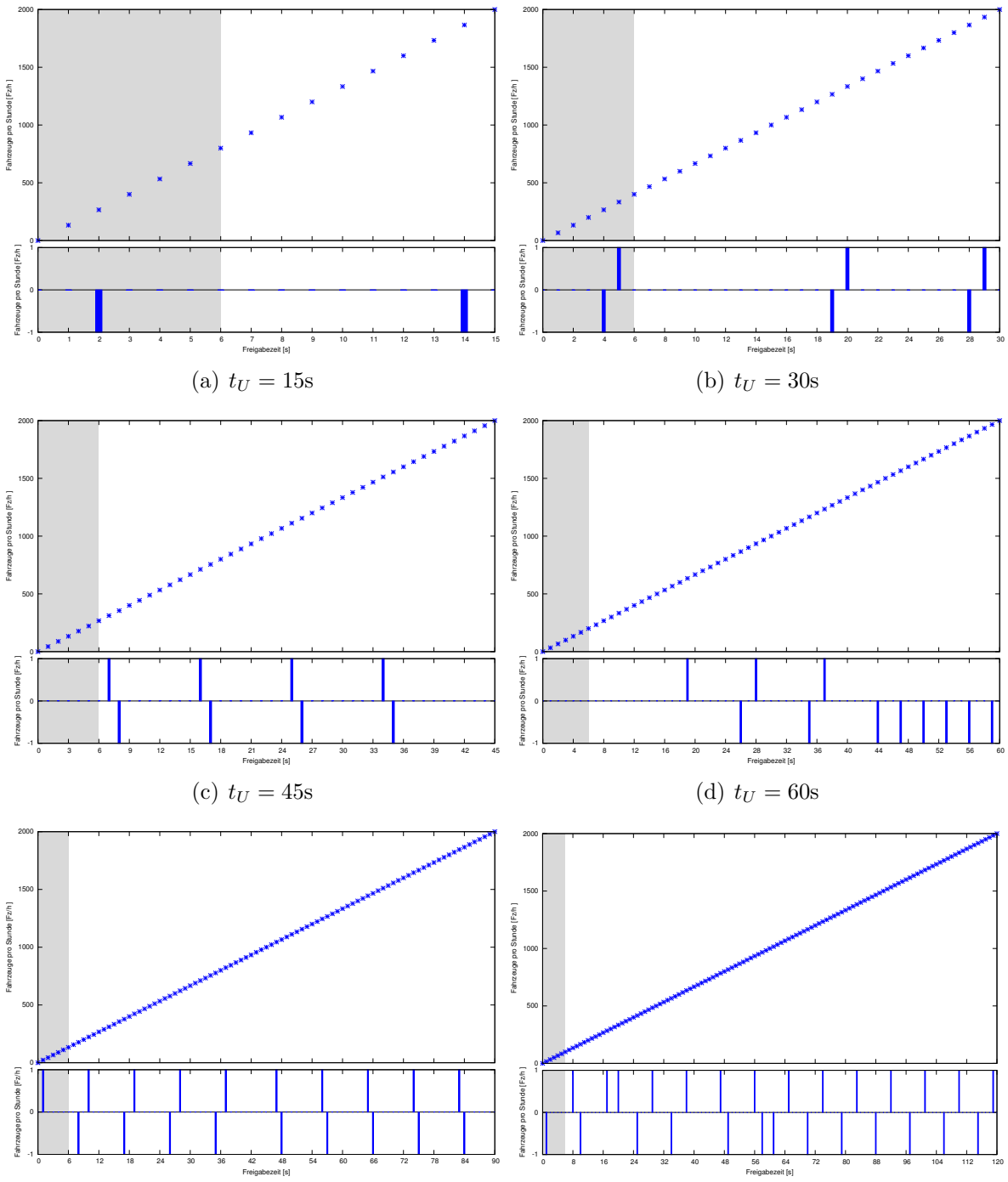
5.2.1 Teil 1 - Bei Verwendung eines einfachen Netzes

Aufbau

Für den Test werden beide Implementierungen mit den gleichen Eingangsdaten simuliert. Als Eingangsdaten dienen Netzwerk und Pläne aus Abschnitt 5.1. Im Anschluss an die Simulation können die dabei generierten Ergebnisse in Form der Events miteinander verglichen werden.

Resultate

In einem ersten Versuch wurde keine Lichtsignalanlage definiert und die neue Implementierung lief ohne die für die LSA relevanten Definitionsdateien. Demnach sollte in der



(a) $t_U = 15$s (b) $t_U = 30$s

(c) $t_U = 45$s (d) $t_U = 60$s

(e) $t_U = 90$s (f) $t_U = 120$s

Abbildung 20: Knotenpunktkapazität mit LSA-Implementierung der Variante II

Oberer Graph: Anzahl der Fahrzeuge, die innerhalb einer Stunde den Knotenpunkt passiert haben, abgetragen über die Freigabezeit t_F

Unterer Graph: Differenz zwischen der gemessenen Anzahl und der nach HBS berechneten Anzahl der Fahrzeuge (gemessen - berechnet)

neuen Implementierung ein ähnlich strukturiertes internes Netzwerk generiert werden, wie bei der bisherigen Implementierung es der Fall ist. Der Vergleich der Events hat ergeben, dass die Ergebnisse der beiden Implementierungen identisch sind.

In einem zweiten Versuch soll geprüft werden, ob die Definition einer Lichtsignalanlage auch dann einen Einfluss auf das Simulationsergebnis hat, wenn diese eine Grünzeit in Höhe der Umlaufzeit aufweist. Dazu wurde eine Lichtsignalanlage am Knoten 2 definiert. Deren Umlaufzeit betrug 60 s und die Freigabezeit wurde ebenfalls auf 60 s festgelegt. Durch die Definition einer Lichtsignalanlage wird die interne Struktur der Kante 1 verändert. Die Fahrzeuge müssen zusätzlich zu der *StorageQueue* und *FlowQueue* des *RoadLink*s die Queues des *LaneLink*s durchfahren. Durch die resultierende Grünphase in Höhe der Umlaufzeit sollten dennoch keine spürbaren Auswirkungen auf das Ergebnis der Simulation festzustellen sein. Diese sind auch nicht aufgetreten. Die Ergebnisse beider Implementierungen waren auch hier identisch.

In einem letzten Versuch wurde die vorhandene Lichtsignalanlage auf andere Knoten umgesetzt und zusätzliche Lichtsignalanlagen definiert. Die Ergebnisse waren auch hier identisch. Es kann festgehalten werden, dass die Dateien mit den Ergebnissen beider Simulationen für dieses einfache Netzwerk auf das Bit genau übereinstimmen.

5.2.2 Teil 2 - Bei Verwendung komplexer Netze

Da das in Teil 1 verwendete Netzwerk in seiner Struktur sehr einfach ist und zum Beispiel keine Knotenpunkte mit mehr als einer Abbiegebeziehung aufweist, soll der Test noch einmal mit einem komplizierteren Netzwerk durchgeführt werden.

Aufbau

Für den Test werden beide Implementierungen wiederum mit den gleichen Eingangsdaten simuliert. Es wurde das in Abbildung 21 dargestellte Netz verwendet. Die zugehörigen Eigenschaften der einzelnen Kanten können Tabelle 2 entnommen werden. Diese enthalten Extremwerte wie kurze Steckenabschnitte oder geringe Kapazitäten, um auch Grenzfälle in der Netzwerkdefinition abzudecken.

Es wurden für 2000 Agenten Pläne und damit 2000 Fahrzeuge definiert. 1000 davon befahren die Route von Kante 1 nach Kante 6, die restlichen von Kante 2 nach Kante 6. Alle Agenten besitzen die gleiche gewünschte Abfahrtzeit. Da beide Routen die gleichen Eigenschaften besitzen, treffen die Fahrzeugströme von Kante 1 und Kante 2 kommend am Knoten 5 aufeinander. Bedingt durch die identische Abfahrt- und Reisezeit begegnen sich zwei Fahrzeuge „auf“ dem Knoten jeweils im gleichen Zeitschritt. Entscheidend ist

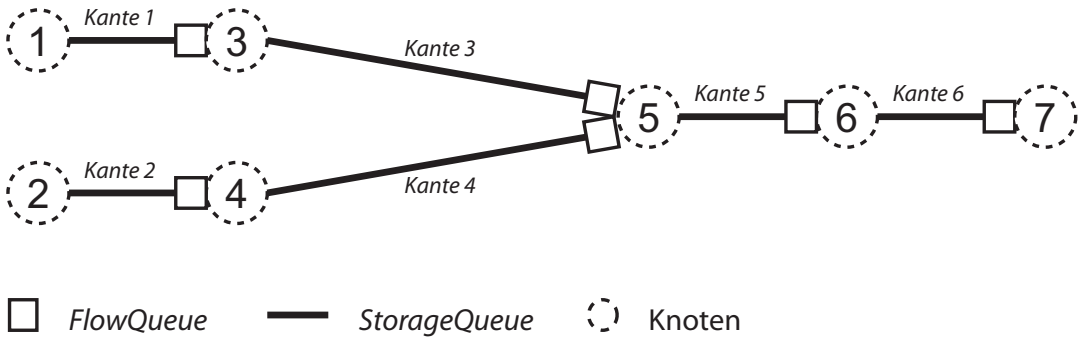


Abbildung 21: Netzwerk für den Vergleich mit der bisherigen Implementierung Dargestellt ist der für den Test relevante Ausschnitt. Die rückführenden Kanten von Knoten 7 zu Knoten 1 bzw. 2 sind nicht dargestellt. Eigenschaften der Kanten können Tabelle 2 entnommen werden.

Tabelle 2: Eigenschaften der in Abbildung 21 dargestellten Kanten
Angabe der Länge l, der Anzahl der Spuren n, der daraus resultierenden *StorageCapacity* SC für jede *StorageQueue* SQ und der zusätzlich verwendeten *FlowCapacity* FC aller *FlowQueue*s FQ

Kante [ ]	*l* [m]	*n* [ ]	*SC* [Fzg]	*v* [m/s]	tt_0 [s]	*FC* [Fzg/h]	*FC* [Fzg/s]
1	60,0	1	8	13,88	4,32	1.514	0,421
2	60,0	1	8	13,88	4,32	1.514	0,421
3	594,5	1	79	23,12	25,71	1.637	0,455
4	594,5	1	79	23,12	25,71	1.637	0,455
5	30,4	1	4	5,40	5,63	973	0,270
6	60,0	1	8	13,88	4,32	1.800	0,500

an dieser Stelle welcher von den beiden *inLink*s 3 und 4 innerhalb der `moveLinks()`-Methode als erstes aufgerufen wird. Dessen Fahrzeug wird auch als erstes auf den *outLink* 5 gesetzt.

Für eine exakte Übereinstimmung der Simulationsergebnisse müssen beide Fahrzeuge nicht nur im gleichen Zeitschritt den Knoten 5 passieren, sondern es muss auch in beiden Simulationen in der gleichen Reihenfolge geschehen. Andernfalls kann es Auswirkungen auf die Reisezeit der beiden Agenten und den Verkehrsfluss der nachfolgenden Kanten haben.

Resultate

In einem ersten Versuch wurden keine Lichtsignalanlagen definiert. Die Resultate haben gezeigt, dass die Ergebnisse beider Simulationen identisch sind. Es kann daraus geschlos-

sen werden, dass beide Simulationen Knoten ohne Lichtsignalanlagen gleich behandeln.

In einem zweiten Versuch wurde eine Lichtsignalanlage an Knoten 5 definiert. Diese regelt den Verkehr sowohl von Kante 3 nach Kante 5 als auch den Verkehr von Kante 4 nach Kante 5. Die Freigabezeit beider Signalgruppen beträgt 60 s bei einer Umlaufzeit von ebenfalls 60 s. Die Resultate haben gezeigt, dass in diesem Versuch die Ergebnisse nicht übereinstimmen. Zwar sind die Ergebnisse anfangs identisch, später divergieren diese jedoch bezüglich der Reihenfolge, mit der die Fahrzeuge von Kante 3 oder 4 auf Kante 5 gesetzt werden. Variationen in den Kanteneigenschaften haben gezeigt, dass diese Abweichungen nicht zwangsläufig auftreten müssen, aber sich mit Fortschreiten der Simulation die Wahrscheinlichkeit dafür erhöht.

Der entscheidende Unterschied zu Teil 1 dieses Tests liegt im Aufbau des Netzes. In Teil 1 spielt die Reihenfolge der *inLink*s an Knoten 2 keine Rolle, da nur ein *inLink* vorhanden ist. Es können also durch die in Unterabschnitt 3.4.4 beschriebenen Vorgänge keine Abweichungen entstehen. Das Netz im zweiten Teil des Tests besitzt dagegen an Knoten 5 zwei *inLink*s.

Es kann abschließend festgehalten werden, dass die neue Implementierung für unsignalisierte Netze auf das Bit genau die gleichen Resultate liefert wie die bisherige Implementierung. Sie ist daher in der Lage, deren Funktionalität mit zu übernehmen. Für den seltenen Fall von mit Dauergrün definierten Lichtsignalanlagen, können die Ergebnisse bitgenau übereinstimmen, müssen es aber nicht. Auf jeden Fall sind die Fahrtzeiten auch hier bei beiden Implementierungen identisch. Ist dagegen ein Teil der Knoten mit Lichtsignalanlagen ausgestattet und besitzen diese einen realistischen Signalzeitenplan, so wird erwartet, dass die Simulationsergebnisse beider Implementierungen auf jeden Fall divergieren. Dies wurde an dieser Stelle nicht explizit getestet, kann aber prinzipbedingt abgeleitet werden.

5.3 Unter Verwendung des Replanning-Algorithmus

In diesem Beispiel geht es darum zu zeigen, dass der bisherige Replanning-Prozess auch mit dem neuen Modell unverändert funktioniert. Um das gewährleisten zu können, muss wie im vorangegangenen Test sowohl das Einlesen aller Dateien wie Netzwerk, initiale Pläne und LSA-Beschreibungen wie auch das Verarbeiten der darin enthaltenen Informationen funktionieren.

Neu ist die in Unterabschnitt 3.4.2 beschriebene Einbindung der Routensuche. Es wird konkret getestet, ob das eingelesene Netzwerk korrekt invertiert wird, ob die darauf berechneten Routen in ihrer Distanz und ihrem Zeitbedarf stimmen und ob die berechneten

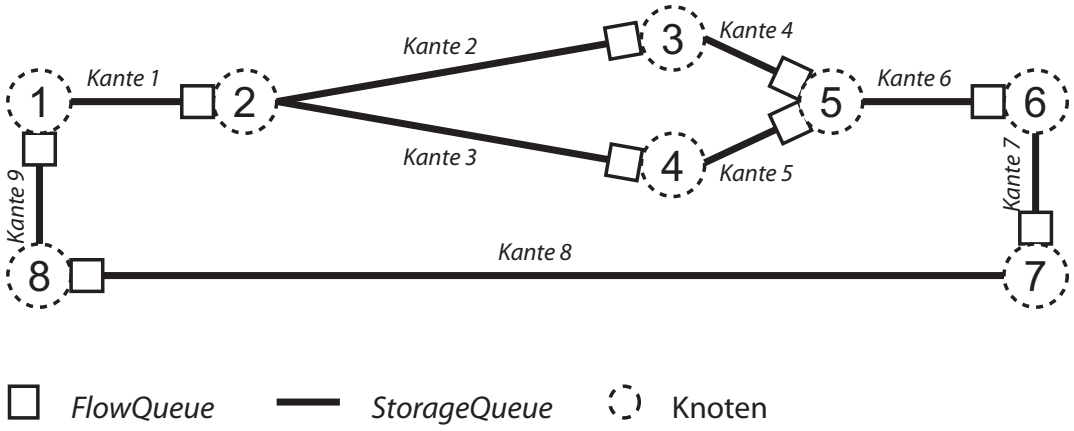


Abbildung 22: Zwei Routen Netzwerk - Aufbau
Erweiterung des Netzes aus Abbildung 18 um eine Alternativroute durch Einfügen der Knoten 3, 4 und 5 und der dazugehörenden Kanten 3, 4 und 5

Tabelle 3: Zwei Routen Netzwerk - Eigenschaften der Kanten
Dargestellt sind für jede Kante: Länge l, Anzahl der Spuren n, zulässige Höchstgeschwindigkeit v, davon abgeleitete $FreeLinkTravelTime$ tt_0 und $FlowCapacity$ FC.

Kante [ ]	*l* [m]	*n* [ ]	*SC* [Fzg]	*v* [m/s]	tt_0 [s]	*FC* [Fzg/h]	*FC* [Fzg/s]
1	100	1	13	13,88	7,20	1.800	0,500
2	700	1	93	13,88	50,43	1.800	0,500
3	700	1	93	13,88	50,43	1.800	0,500
4	100	1	13	13,88	7,20	1.800	0,500
5	100	1	13	13,88	7,20	1.800	0,500
6	100	1	13	13,88	7,20	1.800	0,500
7	500	1	67	13,88	36,02	1.800	0,500
8	1.000	1	133	13,88	72,05	1.800	0,500
9	500	1	67	13,88	36,02	1.800	0,500

Routen korrekt auf das ursprüngliche Netz zurück konvertiert werden können. Soll das Verhalten der Agenten analysiert werden, so benötigt man ein Netzwerk mit zwei verschiedenen Routen zum Ziel.

5.3.1 Teil 1 - Zwei getrennte Lichtsignalanlagen

Aufbau

Das Netzwerk basiert auf dem des ersten Tests. Es wird erweitert durch eine Alternativroute wie in Abbildung 22 dargestellt. Die Kapazitäten werden wie in Tabelle 3 dargestellt angepasst.

Bis auf Knoten 3 und Knoten 4 sind alle Knoten unsignalisiert. Die beiden signalisierten Knoten haben beide eine Umlaufzeit von $t_U = 60\,\mathrm{s}$. Knoten 3 wird von Sekunde 0 bis 40 grün signalisiert und Knoten 4 in den anschließenden 20 Sekunden. Durch die Signalisierung wird die *FlowCapacity* der beiden Knoten von 1800 Fz/h auf 1200 Fz/h bzw. auf 600 Fz/h herabgesetzt. Zusammengenommen verfügen beide Routen dennoch über eine Gesamtkapazität von 1800 Fz/h. Es ist somit theoretisch möglich, dass der maximale Zustrom auf den Engpass diesen auch ohne Behinderung passieren kann. Dazu müssen jedoch alle ankommenden Fahrzeuge ideal auf die beiden Routen verteilt werden. Andernfalls treten Rückstaueffekte auf und ein Teil der Kapazität auf der wegführenden Kante 6 bleibt ungenutzt. Die durch die Events generierten Reisezeiten wurden in Zeitintervallen zu je 15 Minuten zusammengefasst.

Für den Test wurden 450 Agenten und damit 450 Fahrzeuge generiert. Alle besitzen die gleiche gewünschte Abfahrtzeit. Durch die Begrenzung der *FlowCapacity* von Kante 1 auf 1800 Fz/h, kann mit Erreichen der Abfahrtzeit für die Dauer von 15 Minuten ein konstanter Strom an Fahrzeugen garantiert werden.

Alle Fahrzeuge starten mit der gleichen initialen Route, welche über die Knoten 1, 2, 3, 5 und 6 führt. Diese Verteilung der Fahrzeuge auf die beiden Routen ist offensichtlich nicht optimal. Im Folgenden wird die Simulation 1000 Mal iteriert. 10 % der Agenten dürfen sich im Anschluss an eine Iteration eine neue Route suchen. Der Rest wählt den bei der letzten Iteration besten Plan aus.

Resultate

Die theoretische minimale Reisezeit auf einer realen Straße beträgt bei einer zu bewältigenden Distanz von 1500 m und einer *FreeLinkTravelTime* von 13,88 m/s circa 108 s. Innerhalb der Simulation benötigt ein Agent für eine 100 m lange Kante exakt 7,2 s. Die Simulation rechnet jedoch nur in ganzen Zeitschritten. Die benötigte Reisezeit für die Kante wird auf 7 s abgerundet. Für das Passieren des anschließenden Knotens muss der nächste Zeitschritt abgewartet werden und es wird eine weitere Sekunde benötigt, so dass die Fahrtzeit vom Betreten bis zum Verlassen der Kante insgesamt 8 s beträgt. Als Folge muss zusätzlich zu den in Tabelle 3 aufgelisteten Reisezeiten für jede passierte Kante eine zusätzliche Sekunde aufgeschlagen werden. Da die Agenten stets am Ende einer Kante ihre Route beginnen oder beenden, setzt sich die Gesamtreisezeit aus den abgerundeten Reisezeiten der Kanten 1, 2, 4, 6 und 7 sowie aus dem Überqueren der Knoten 1, 2, 3, 5 und 6 zusammen. Die in der Simulation minimal realisierbare Reisezeit

beträgt demnach 112 s. In der ersten Iteration hat der als erstes abfahrende Agent eine Reisezeit von 112 s realisieren können.

Gleichfalls existiert eine minimal realisierbare Reisezeit für den letzten Agenten. Bei idealer Verteilung der Agenten auf beide Routen steht zwischen Start- und Zielkante eine Gesamtkapazität von 1800 Fz/h zur Verfügung. Daraus resultiert eine minimale Reisezeit für den letzten Agenten von 1012 s, die sich aus der Minimalreisezeit des ersten Agenten von 112 s und der aus der Kapazitätsbeschränkung resultierenden Zeit bis zum Passieren aller Agenten von 15 min (900 s) zusammensetzt.

In der ersten Iteration nutzen alle Agenten noch die initiale Route über Knoten 3. Der letzte Agent benötigt hier 1450 s. In den späteren Iterationen nähert sich dieser Wert dem Optimum von 1012 s. In der letzten Iteration, Iteration 1000, beträgt er 1068 s.

Eine visuelle Kontrolle der Simulation hat gezeigt, dass bereits nach den ersten zehn Iterationen Ansätze von Fahrzeugpulks zu erkennen sind, die zeitlich mit den Grünphasen der Lichtsignalanlage übereinstimmen. Die einzelnen Pulks erreichen den Knotenpunkt mit Beginn der Grünphase und enden mit Beginn der Grünphase der Alternativroute. Während eines Umlaufs bleiben jeweils nur wenige Fahrzeuge am Knotenpunkt „hängen“. Die Kapazität der abfließenden Kante 6 wird bis auf vereinzelte Lücken vollständig genutzt.

5.3.2 Teil 2 - Eine kombinierte Lichtsignalanlage

Der vorangegangene Test betrachtet zwei Routen mit jeweils einer Lichtsignalanlage auf jeder Route. Beide Lichtsignalanlagen sitzen auf separaten Knoten und regeln jeweils genau eine Abbiegebeziehung und damit eine Route. Es konnte damit nicht überprüft werden, ob verschiedene Abbiegebeziehungen auf einer Kante auch verschiedene Reisezeiten generieren und diese den Agenten korrekt zur Verfügung stellen.

Deshalb soll hier untersucht werden, ob das System auch funktioniert, wenn eine Lichtsignalanlage beide Routen regelt, sprich zwei Abbiegebeziehungen steuert. Dazu muss die Simulation für jede Abbiegebeziehung unterschiedliche Daten genieren, vorhalten und auswerten können.

Aufbau

Es wird das gleiche Netzwerk wie im vorangegangenen Test verwendet. Auch die initialen Pläne der Agenten sind gleich. Alle 450 Agenten benutzen anfangs die Route über die Knoten 1-2-3-5-6. Geändert hat sich die Anordnung der Lichtsignalanlagen. Die beiden LSA an den Knoten 3 und 4 werden durch eine einzelne am Knoten 2 ersetzt. Diese

regelt die beiden Abbiegebeziehungen von Kante 1 nach Kante 2 bzw. nach Kante 3. Die Umlaufzeit beträgt wiederum 60 s. Von Kante 1 nach Kante 2 wird die ersten 40 s Grün geschaltet, von Kante 1 nach Kante 3 die restlichen 20 s. Als Resultat besitzen beide Routen die gleichen Kapazitäten wie im vorangegangenen Test. Die Gesamtkapazität beider Routen beträgt bei optimaler Ausnutzung wiederum 1800 Fz/h.

Resultate

Die Resultate haben gezeigt, dass der erste Agent in der ersten Iteration wiederum 112 s und der letzte Agent wiederum eine Reisezeit von 1450 s benötigt hat. In Iteration 1000 weist der letzte Agent eine Reisezeit von 1145 s auf. Diese liegt leicht höher als beim vorangegangenen Test (1068 s).

Zurückzuführen ist dies auf die Kapazität der beiden *LaneLink*s von Kante 1. Diese sind standardmäßig 45 m lang und fassen jeweils 6 Fahrzeuge. Da alle Agenten anfangs die gleiche Route besitzen, sind Überstauungen unvermeidlich. Eine Folge ist, dass sich die Reisezeit auch für die eigentlich ungestaute Richtung erhöht und somit das Lernen über die Iterationen hinweg verlangsamt wird. Es werden demzufolge mehr Iterationen benötigt, um auf ein ähnliches Resultat wie im vorangegangenen Test zu kommen.

Auch in diesem Test konnte eine Pulkbildung der Fahrzeuge beobachtet werden.

5.3.3 Teil 3 - Vergleich mit dem bisher implementierten Modell

Dieser Test ergänzt die Tests in Abschnitt 5.2 und soll zeigen, ob die Dateien mit den Ergebnissen unter Verwendung des Replanning-Algorithmus ebenfalls auf das Bit genau identisch sind.

Aufbau

Da das in Abschnitt 5.2 beschriebene Netzwerk keine Routenwahl zulässt, wurde stattdessen das Netzwerk zusammen mit den zugehörigen Plänen aus den vorangegangenen Tests dieses Abschnitts verwendet. Lichtsignalanlagen wurden jedoch nicht definiert. Durch den Wegfall der Lichtsignalanlagen existieren auch keinerlei Kapazitätsengpässe mehr und der Anreiz zur Suche einer alternativen Route entfällt. Zwecks Kompensation wurde daher das Netzwerk leicht modifiziert und die Kapazität der Kanten 4 und 5 in Abbildung 22 von 1800 auf 800 Fz/h gesenkt. Sowohl die bisherige wie auch die Implementierung des Subnetzmodells haben als Eingangsdaten die gleichen Pläne, das gleiche Netzwerk, wie auch die gleiche Konfigurationsdatei erhalten.

In einem ersten Versuch wurde der bisher implementierte Replanning-Algorithmus verwendet. Anschließend wurde der unter Verwendung eines invertierten Netzwerkes arbeitende neue Replanning-Algorithmus getestet und dessen Ergebnisse miteinander verglichen. Beide Versuche hatten einen Umfang von 10 Iterationen, wobei sich nach jeder Iteration 10 % der Agenten eine neue Route suchen konnten.

Resultate

Der erste Versuch hat gezeigt, dass der alte Replanning-Algorithmus mit beiden Implementierungen funktioniert. Die Dateien mit den Plänen und die aus der nachfolgenden Simulation resultierenden Dateien mit den Events waren in jeder Iteration auf das Bit identisch. Der zweite Versuch hat die gleichen Ergebnisse geliefert. Auch hier waren die Ergebnisse der beiden Implementierungen untereinander identisch.

Unterschiede gab es im Vergleich der beiden Replanning-Algorithmen untereinander. Diese lieferten unterschiedliche Längenangaben für die neu ermittelten Routen und demzufolge auch abweichende Reisezeiten. Der Grund liegt jedoch darin, dass der Umfang der Route in MATSim nicht genau spezifiziert ist. Die Länge einer Route im neuen Replanning-Algorithmus setzt sich aus allen Kanten zusammen, beginnend bei der ersten Kante im Anschluss der Startkante und endend inklusive der Zielkante. Begründet wird diese Approximation damit, dass der Agent die Startkante nahezu sofort verlässt, die Zielkante jedoch nahezu komplett abfahren muss. Die alte Implementierung berücksichtigt jedoch weder die Startkante noch die Zielkante, was mit der Kompatibilität zu anderen Simulationen als der Queue-Simulation begründet wird.

Zusammenfassend kann festgestellt werden, dass sowohl der neue als auch der alte Replanning-Algorithmus identische Ergebnisse liefert, solange beide Implementierungen jeweils denselben Algorithmus verwenden. Es konnte gezeigt werden, dass die erste Anforderung aus Kapitel 3 vom Subnetzmodell vollständig erfüllt wird.

5.4 Vierarmiger Knotenpunkt

Dieser Test verwendet einen vierarmigen Knotenpunkt. Er dient dazu, die komplette Infrastruktur beginnend mit dem Definieren von Netzwerk und Signalzeitenplänen bis hin zur Auswertung der Simulationsergebnisse an einem realen Beispiel zu testen.

5.4.1 Aufbau

Der verwendete Knotenpunkt entstammt einem Planungsentwurf. Es handelt sich dabei um einen realen Knotenpunkt. Ein an den Entwurf angelehnter Lageplan mit allen verwendeten Bezeichnern ist in Abbildung 23 dargestellt. Der Knotenpunkt besitzt 4 Arme. Die Arme A und C sind zu- und wegführend zweispurig ausgeführt und besitzen eine Gesamtkapazität von jeweils 3600 Fz/h. Innerhalb des Kreuzungsbereich weiten sich die zuführenden Strecken auf vier Fahrstreifen auf. Dabei steht den Abbiegebeziehungen 1, 3, 7 und 9 jeweils ein separater Fahrstreifen zur Verfügung. Für den geradeaus führenden Verkehr (2 und 8) werden jeweils zwei Fahrstreifen vorgehalten. Somit beträgt die Gesamtkapazität für die Arme A und C innerhalb des Kreuzungsbereichs jeweils 7200 Fz/h.

Die Arme B und D besitzen jeweils einen zu- und einen wegführenden Fahrstreifen mit einer Kapazität von 1800 Fz/h. Diese weiten sich für den zuführenden Fahrzeugstrom innerhalb des Kreuzungsbereichs auf zwei Fahrstreifen auf, so dass eine Gesamtkapazität von 3600 Fz/h bereit steht. Dabei stehen den links abbiegenden Verkehrsströmen 4 und 10 jeweils ein separater Fahrstreifen mit einer Kapazität von 1800 Fz/h zur Verfügung. Die beiden restlichen Verkehrsströme 5 und 6 bzw. 11 und 12 müssen jeweils einen Fahrstreifen mit einer Kapazität von ebenfalls 1800 Fz/h gemeinsam nutzen.

Zusätzlich stehen die folgenden Daten zur Verfügung:

- Die Belastungen in Kfz/h für alle Abbiegebeziehungen, dargestellt in Abbildung 24.
- Ein nach den „Richtlinien für Lichtsignalanlagen“, RiLSA (1992), optimierter Signalzeitenplan, dargestellt in Abbildung 25.

Von den im Signalzeitenplan beschriebenen Signalgebern werden in dieser Arbeit nur die Signalgeber K1, K2, K3, K4, K5 und K6 verwendet. Die restlichen Signalgeber betreffen entweder Radfahrer oder Fußgänger. Beide Gruppen von Verkehrsteilnehmern werden in MATSim nicht abgedeckt und bleiben daher unberücksichtigt.

Dem Signalzeitenplan ist zu entnehmen, dass es sich um ein 3-Phasen-System handelt. In Phase I bekommen die Abbiegebeziehungen 2, 3, 8 und 9 eine Freigabezeit von 17 s. In Phase II können die Linksabbieger der Knotenarme A (1) und C (7) fahren. Die

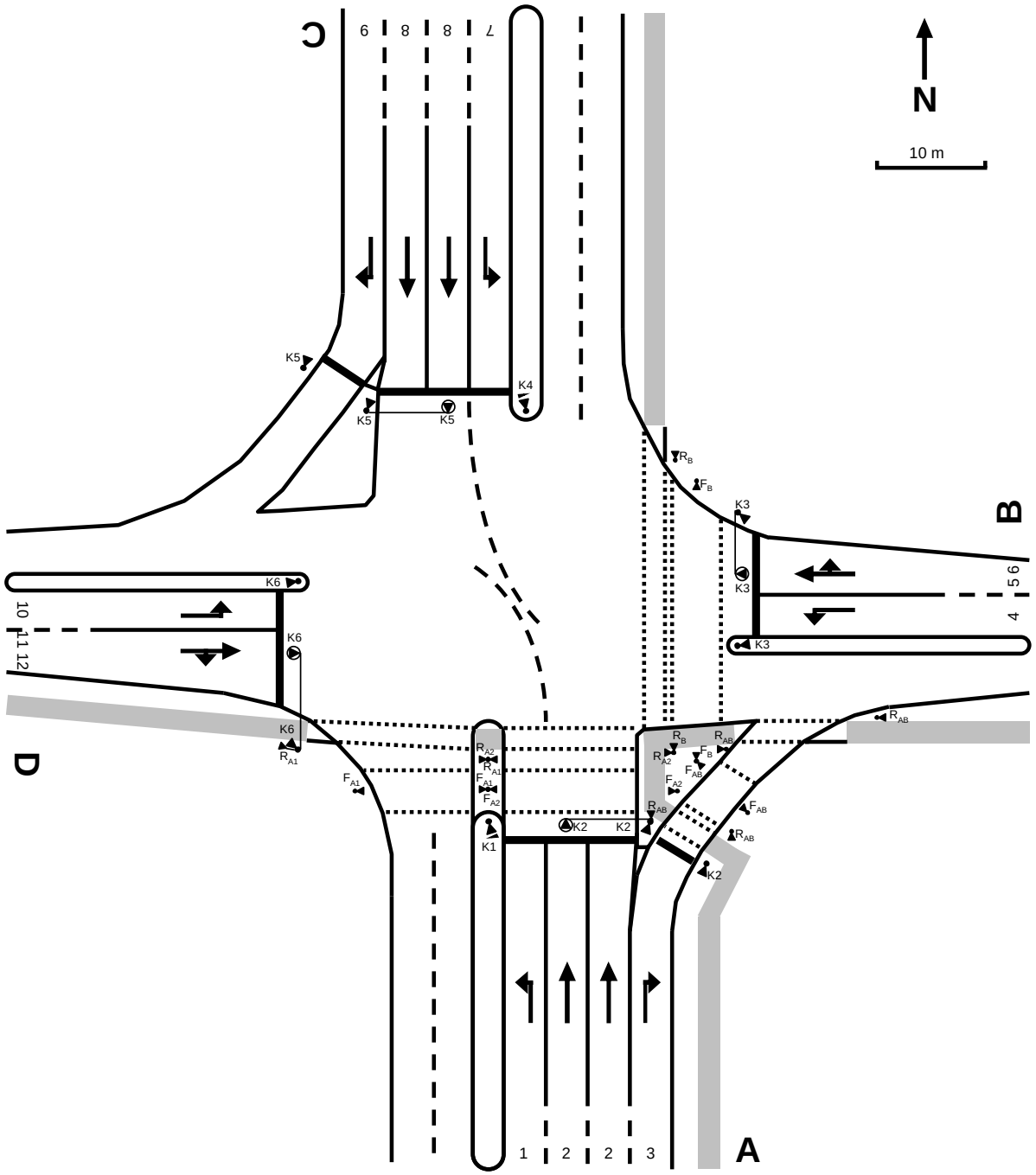


Abbildung 23: LSA-Lageplan aus dem Planungsentwurf
Dargestellt ist die komplette Knotenpunktgeometrie mit der Anordnung der Fahrspuren und der Positionen aller Signalgeber. Die Knotenarme sind mit A, B, C und D beschriftet, die Abbiegebeziehungen mit 1 bis 12. Von den eingezeichneten Signalgebern finden in dieser Arbeit lediglich die für den Kfz-Verkehr relevanten Signalgeber K1 bis K6 Verwendung.

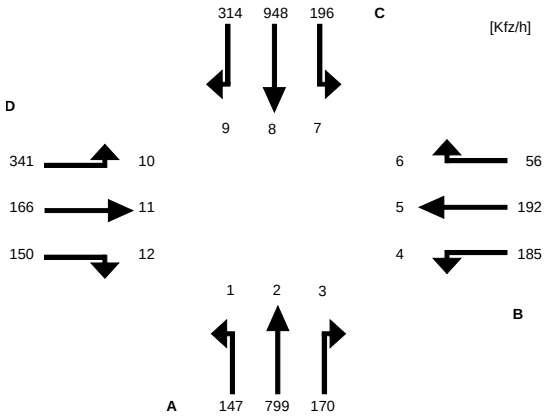


Abbildung 24: Zugrunde liegende Belastungen am vierarmigen Knotenpunkt
Innen: Bezeichnung der möglichen Abbiegebeziehungen
Außen: Belastungen in Kfz/h

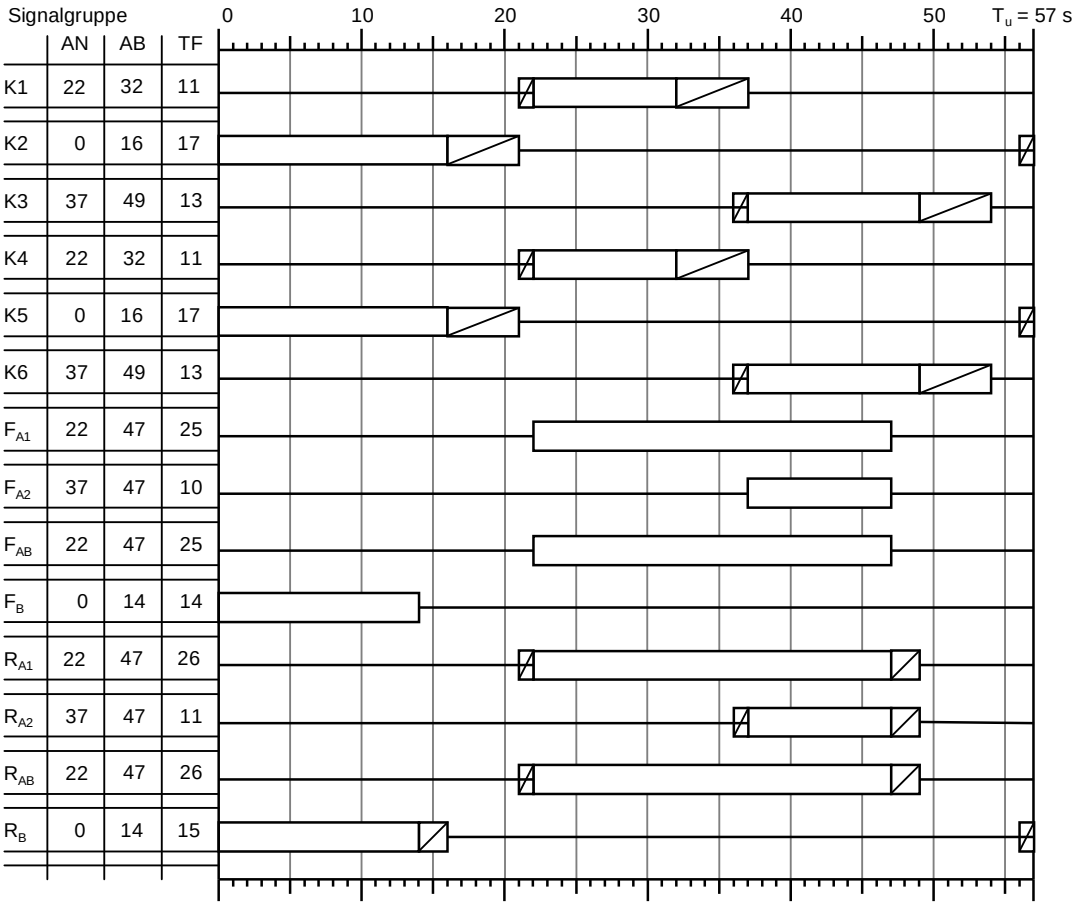


Abbildung 25: Signalzeitenplan aus dem Planungsentwurf
In dieser Arbeit finden nur die für den Kraftfahrzeugverkehr relevanten Signalgeber K1 bis K6 Verwendung.

Freigabezeit beträgt 11 s. Die Freigabezeit der dritten Phase beträgt 13 s und gilt für die Abbiegebeziehungen 4, 5, 6, 10, 11 und 12. Zuzüglich 19 Sekunden für die Zwischenzeiten beträgt die Umlaufzeit 57 Sekunden.

5.4.2 Übertragung auf das Modell

Da es sich um ein 3-Phasen-System handelt und alle Abbiegebeziehungen zugelassen sind, existiert demzufolge eine ungesicherte Abbiegebeziehung. Es handelt sich dabei um die Linksabbieger der Arme B und D. Ungesicherte Abbiegebeziehungen werden vom LSA-Modell dieser Arbeit nicht abgedeckt. Es werden keine Konflikte konkurrierender Ströme gelöst. Als Folge ist für die Abbiegebeziehungen 4 und 10 im Vergleich zum Planungsentwurf eine leicht erhöhte Kapazität zu erwarten, da beide bei grüner Signalisierung ungehindert den Knotenpunkt passieren dürfen.

Abbildung 26 zeigt die Modellierung des Lageplans mit dem Subnetzmodell. Die genauen Daten können Tabelle 4 entnommen werden. Die Entscheidungspunkte im Anschluss an die *FlowQueue*s M, O, Q und S liegen 45 m vom Ende der Kante entfernt. Daraus ergibt sich eine *LaneStorageCapacity* von 6 Fahrzeugen für die einspurigen *LaneLink*s bzw. von 12 Fahrzeugen für die zweispurigen *LaneLink*s. Die vom Knotenpunkt wegführenden *StorageQueue*s sind alle 300 m lang und fassen 40 bzw. 80 Fahrzeuge. Die auf den Knotenpunkt zuführenden *StorageQueue*s 100, 200, 300 und 400 besitzen eine um 45 m gekürzte Länge und können daher nur 34 bzw. 68 Fahrzeuge fassen.

Die *FlowCapacity* der einzelnen *FlowQueue*s A bis T, ist abhängig von der Anzahl der Spuren und kann ebenfalls der Tabelle 4 entnommen werden. Die *FlowQueue*s N, P, R und T gehören zu den vom Knotenpunkt wegführenden Kanten und sind in Abbildung 26 nicht abgebildet.

Im letzten Schritt wurden Pläne generiert, welche für die Dauer einer Stunde einen kontinuierlichen Fluss an Agenten generieren. Insgesamt wurden 3664 Agenten generiert, deren initiale Routenwahl sich entsprechend den Belastungen in Abbildung 24 verteilt. Die Abfahrtzeiten wurden zufällig variiert, so dass diese sich einer Gleichverteilung annähernd über die gesamte Stunde verteilen. Im Resultat entsteht für eine Stunde ein nahezu gleichmäßiger Fahrzeugstrom mit leichten Fluktuationen in der Belastung.

5.4.3 Resultate

Die simulierte Zeitdauer umfasste circa eine Stunde zuzüglich der Fahrtzeit des letzten Agenten durch das Netz. Es konnte gezeigt werden, dass sämtliche Fahrzeugströme mit

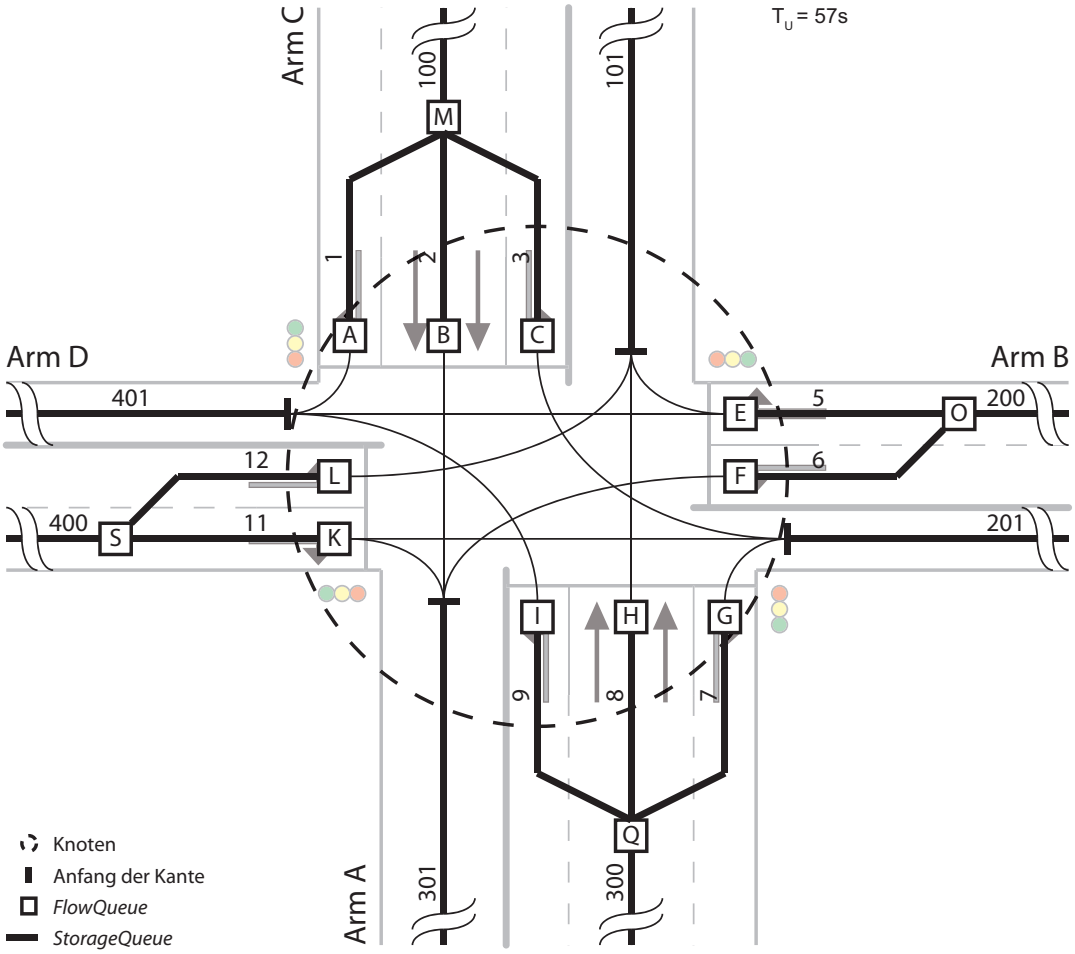


Abbildung 26: Vierarmiger Knotenpunkt, umgesetzt mit dem Subnetzmodell
Dargestellt sind die *LaneLink*s 1 bis 12 und die *RoadLink*s 100, 200, 300 und 400. Die *LaneLink*s 4 und 10 entfallen, da sie mit *LaneLink* 5 bzw. 11 zu einem gemischten Fahrstreifen zusammengefasst sind. Ebenfalls zusammengefasst sind die beiden geradeaus führenden Fahrspuren der Knotenarme A und C. Diese werden durch eine erhöhte Kapazität der *LaneLink*s 2 bzw. 8 repräsentiert.

Tabelle 4: Eigenschaften des in Abbildung 26 dargestellten Knotenpunktes
Angabe der Länge l, der Anzahl der Spuren n, der daraus resultierenden *StorageCapacity* SC für jede *StorageQueue* SQ und der zusätzlich verwendeten *FlowCapacity* FC aller *FlowQueue*s FQ

SQ [ ]	*l* [m]	*n* [ ]	*SC* [Fzg]	*FQ* [ ]	*FC* [Fzg/s]
1	45	1	6	A	0,500
2	45	2	12	B	1,000
3	45	1	6	C	0,500
4	entfällt			D	entfällt
5	45	1	6	E	0,500
6	45	1	6	F	0,500
7	45	1	6	G	0,500
8	45	2	12	H	1,000
9	45	1	6	I	0,500
10	entfällt			J	entfällt
11	45	1	6	K	0,500
12	45	1	6	L	0,500
100	255	2	68	M	1,000
101	300	2	80	N	1,000
200	255	1	34	O	0,500
201	300	1	40	P	0,500
300	255	2	68	Q	1,000
301	300	2	80	R	1,000
400	255	1	34	S	0,500
401	300	1	40	T	0,500

dem implementierten Signalzeitenplan bewältigt werden konnten und keine Rückstaueffekte auftraten. Dies ist vor allem darauf zurückzuführen, dass der Signalzeitenplan exakt auf die zugrunde liegenden Belastungen abgestimmt ist und ausreichend Reserven bereithält, um durch Schwankungen in der Nachfrage verursachte Stauungen bewältigen zu können. So beträgt der Sättigungsgrad g für die kritische Abbiegebeziehung 8 nach Gleichung 6 auf Seite 29 $g = 0,883$. Dieser Wert liegt knapp unter der im HBS angegebenen Grenze von 0,9, ab der mit einem Rückstau zu rechnen ist.

Es konnte in einem zweiten Simulationslauf gezeigt werden, dass die Verkürzung der Grünzeit für einen der kritischen Verkehrsströme (Abbiegebeziehungen 7, 8 und 10) zu Rückstaus führt. Diese blockieren nach Überstauung der zugehörigen Entscheidungspunkte M bzw. S auch die entsprechenden anderen Abbiegebeziehungen. Für die Abbiegebeziehung 8 war nur eine Verkürzung von einer Sekunde nötig, um temporäre Rückstaus mit unterschiedlicher Länge zu verursachen. Diese konnten zwischenzeitlich nahezu vollständig abgebaut werden. Die Beobachtung des Modells deckt sich auch hier

mit den Werten aus dem HBS. Durch die Verkürzung der Grünzeit von einer Sekunde steigt der Sättigungsgrad auf 0,938 und liegt damit zwischen 0,9 und 1 im Bereich kurzfristiger Überlastungen. Ein durch eine Verkürzung der Grünzeit von zwei Sekunden verursachter Stau konnte dagegen innerhalb der simulierten Stunde nicht wieder abgebaut werden. Nach HBS entspricht dies einem Sättigungsgrad von größer eins, der sich wiederum mit dem für Abbiegebeziehung 8 kalkulierten von $g = 1,001$ deckt.

In einem dritten Simulationslauf wurde untersucht, wie sich die Änderung der Umlaufzeit auswirkt. Trotz gleichen Verhältnisses von Freigabezeit t_F zu Umlaufzeit t_U wird angenommen, dass sich die Kapazität des Knotenpunktes bei sehr hohen Umlaufzeiten verringert und eine Freigabezeit von $t_F = 40\,\mathrm{s}$ bei einer Umlaufzeit von $t_U = 240\,\mathrm{s}$ in einer geringeren Knotenpunktkapazität resultiert als bei den Werten $t_F = 10\,\mathrm{s}$ zu $t_U = 60\,\mathrm{s}$. Dazu wurde unter Beibehaltung der Grünanteile die Umlaufzeit von 57 s auf 228 s vervierfacht.

Im Ergebnis konnte gezeigt werden, dass durch die längere Wartezeit die Wahrscheinlichkeit dafür steigt, einen Entscheidungspunkt zu überstauen. Die Folge ist das Blockieren einzelner Fahrspuren, was sich insbesondere bei den Armen mit getrennter Signalisierung (Arme A und C) bemerkbar macht. Ein ungestörtes Vorsortieren auf die einzelnen Fahrspuren ist dann nicht mehr möglich. Durch das Überstauen eines Entscheidungspunktes müssen Fahrzeuge während der Grünphase von den anschließenden *RoadLink*s 100, 200, 300 und 400 nachrücken. Diese verfügen über weniger Fahrspuren und damit über eine geringere Kapazität. Es können jedoch gegen Ende der Grünphase nur so viele Fahrzeuge nachrücken, wie die Höhe der Kapazität der *RoadLink*s es zulässt. Die Kapazität der *LaneLink*s direkt an der Haltelinie ist damit irrelevant und es geht Kapazität auf den blockierten *LaneLink*s verloren. Dieser Kapazitätsverlust kann auch bei gleichzeitiger Signalisierung aller Abbiegebeziehungen eines Armes nicht durch die verlängerte Grünzeit kompensiert werden.

Es konnte damit gezeigt werden, dass der erwartete Kapazitätsverlust auch im Modell abgebildet werden kann. Dieser Test hat darüber hinaus gezeigt, dass die zu Beginn in Kapitel 3 formulierten Forderungen 2 und 3 vom Modell vollständig erfüllt werden.

6 Zusammenfassung und Ausblick

Die Arbeit hat mit der Beschreibung des Warteschlangenmodells und dessen von Gawron und MATSim eingeführten Erweiterungen begonnen. Darauf aufbauend wurden die Anforderungen an ein erweiterndes Modell formuliert, welches auf dem Warteschlangenmodell aufbaut und in der Lage ist, Lichtsignalanlagen abzubilden. Von den anschließend vorgestellten beiden Ansätzen hat sich der des Subnetzmodells (Modifikation am Netzwerk zur Laufzeit der Simulation) gegenüber dem des Netzwerkmodifikationsmodells (Modifizierung der Eingangsdaten) als geeigneter für die Verwendung in MATSim herausgestellt. Entscheidend war die höhere Kompatibilität zu den vorhandenen Eingangsdaten, welche langfristig den Nachteil einer komplexeren Implementierung überwiegen werden. Daher wurde das Subnetzmodell im Detail spezifiziert und hinsichtlich seiner Erweiterbarkeit untersucht. Anschließend folgte eine Analyse, wie die bisherige Verkehrsflusssimulation in MATSim eingebunden ist und angesprochen werden kann. Darauf aufbauend wurde eine Beispielimplementierung des Subnetzmodells eingebunden und ausführlich getestet.

Zu den grundlegenden Forderungen an das erweiternde Modell gehörten:

- Generierung des gleichen Verkehrsflusses wie die bisherige Implementierung für den Fall, dass keine Lichtsignalanlagen definiert wurden:
  Diese Forderung wird vollständig erfüllt. Die bisherige Implementierung und die Implementierung des Subnetzmodells liefern den gleichen Verkehrsfluss. Dazu gehört insbesondere, dass die Kapazität der einzelnen Kanten und deren Mindestreisezeit identisch sind. Darüber hinaus ist es gelungen, die Kompatibilität zu der bisherigen MATSim-Implementierung so weit herzustellen, dass bei Simulationen mit ausschließlich unsignalisierten Knotenpunkten keinerlei Abweichungen im Ergebnis zu verzeichnen sind. Die Dateien mit den Ergebnissen beider Implementierungen sind auf das Bit identisch. Somit bietet sich das Subnetzmodell als Ersatz für die bisherige Implementierung an.

- Abbildung der für die Kapazität eines Knotenpunktes entscheidenden Eigenschaften wie Anzahl der zur Verfügung stehenden Fahrstreifen, deren Abbiegebeziehungen und Länge:
  Diese Forderung konnte vollständig erfüllt werden. Es ist möglich, an einem Knotenarm für jede Abbiegebeziehung getrennte Fahrstreifen einzurichten, mehrere Abbiegebeziehungen auf einem gemischten Fahrstreifen zusammenzufassen oder

beides zu kombinieren. Zusätzlich kann die Signalisierung unabhängig von der Aufteilung der Fahrstreifen erfolgen. So können mehrere Fahrstreifen von einer gemeinsamen Signalgruppe signalisiert werden oder es werden mehrere Signalgruppen auf einem gemischten Fahrstreifen definiert, die dann die einzelne Abbiegebeziehungen getrennt signalisieren.

- Ausbreitung des Rückstaus eines Fahrstreifens auf den gesamten Knotenarm und auf stromaufwärts anschließende Kanten:
  Auch die dritte Forderung wird vollständig erfüllt. Auftretende Belastungen oberhalb der Kapazität eines Fahrstreifens führen dazu, dass dieser sich beginnt zu stauen und wenn keine weiteren Fahrzeuge mehr aufgenommen werden können, sich die stromaufwärts liegenden Teile des auf der Kante liegenden virtuellen Netzes stauen. Dabei wird der Zugang zu eigentlich ungestauten Fahrstreifen blockiert, was sich auf die Gesamtkapazität eines Knotenarmes respektive auf die im Grundlagenkapitel beschriebene Knotenpunktqualität auswirkt. Es ist dabei so genau, dass Verkehrsbelastungen nahe an der Kapazitätsgrenze einer Lichtsignalanlage kurzfristige Verkehrsstaus verursachen, die bei leicht nachlassender Belastung teilweise wieder abgebaut werden können.

Wie erwartet, gab es Abweichungen bei den Ergebnissen zwischen neuer und alter Implementierung für den Fall der Simulation mit signalisierten Knotenpunkten. Die Tests der Beispielimplementierung haben gezeigt, dass das Subnetzmodell alle gestellten Forderungen vollständig erfüllt und darüber hinaus die Möglichkeit bietet, auch umfangreichere Knotenpunkte detailliert zu modellieren.

Ausblick

Das Subnetzmodell bietet viele Möglichkeiten für Erweiterungen. Mit der in dieser Arbeit vorgeschlagenen Erweiterung ist es möglich, Entscheidungspunkte metergenau auf der Kante zu verankern. Entscheidungspunkte müssen nicht zwangsläufig die von MATSim verwendeten Aktivitäten sein, sondern können auch von einer zukünftigen Implementierung eines Öffentlichen Personennahverkehrs in Form von Haltestellen genutzt werden.

Weiterhin ist es möglich, einzelne Bestandteile des Modells genauer zu modellieren. Beispielsweise kann die Kapazität der verschiedenen Fahrspuren an vorhandene Daten oder getreu nach einem Regelwerk wie dem HBS umgesetzt werden.

Das in der Beispielimplementierung praktizierte zweistufige Einlesen der LSA-relevanten Daten ermöglicht es, auf das Einlesen der eigentlichen Lichtsignalanlagen zu verzich-

ten und sich lediglich auf die Daten zu beschränken, welche die Fahrspuren beschreiben. Damit ist es möglich, Knotenpunkte mit einer unterschiedlichen Spuraufteilung zu simulieren, ohne dass diese zwangsläufig auch signalisiert sein müssen. Es wird erwartet, dass sich damit auch an unsignalisierten Knotenpunkten z.B. Kapazitätsengpässe aufgrund von Spurverengungen modellieren lassen.

Das größte Entwicklungspotential steckt jedoch in der geschaffenen Architektur mit der die Lichtsignalanlagen selbst in MATSim eingebunden wurden und angesprochen werden können. Das Subnetzmodell bildet die Grundlage für die Entwicklung und Erprobung von LSA-Steuerungen. Die bisherige Implementierung dient lediglich dem Auslesen der Signalzeitenpläne und der anschließenden direkten Weitergabe der für einen Knoten relevanten Informationen. Die Erweiterung des vorhandenen LSA-Controlers ist jedoch auch Grundlage für einer Kontrollstrategie folgende Manipulationen der Signalzeitenpläne und die Generierung neuer Signalzeitenpläne während der Simulation. Weiterhin ist die Koordinierung mehrerer LSA-Controler ebenso möglich wie eine zentrale Netzsteuerung oder das Testen adaptiver Ampelsteuerungen und dezentraler Steuerungsmechanismen.

Literaturverzeichnis

[Balmer u. a. 2008] Balmer, M. ; Rieser, M. ; Meister, K. ; Charypar, D. ; Lefebre, N. ; Nagel, K. ; Axhausen, K. W.: *MATSim-T: Architecture and Simulation Times*. Veröffentlicht unter http://fgvsp01.vsp.tu-berlin.de/biblio/297/, Eidgenössische Technische Hochschule Zürich, Technische Universität Berlin, 2008 - Zitiert auf Seite 55

[CCG 2008] CCG ; Teply, S. (Hrsg.) ; Allingham, D. I. (Hrsg.) ; Richardson, D. B. (Hrsg.) ; Stephenson, B. W. (Hrsg.): *Canadian Capacity Guide for Signalized Intersections*. Committee on Canadian capacity guide for signalized intersections, Institute of Transportation Engineering, District 7, Canada, 2008 - Zitiert auf den Seiten 26, 28 und 29

[CROW 2006] CROW: *Handboek verkeerslichtenregelingen*. Niederlande : Centrum voor Regelgeving en Onderzoek in de Grond-, Water- en Wegenbouw en de Verkeerstechniek - CROW, 2006 - Zitiert auf Seite 26

[Dijkstra 1959] Dijkstra, E. W.: A Note on Two Problems in Connexion with Graphs. In: *Numerische Mathematik* 1 (1959), Dezember, Nr. 1, S. 269–271 - Zitiert auf Seite 20

[Flötteröd 2008] Flötteröd, G.: *Traffic State Estimation with Multi-Agent Simulations*, Technische Universität Berlin, Diss., 2008 - Zitiert auf Seite 41

[Gawron 1998] Gawron, C.: An Iterative Algorithm to Determine the Dynamic User Equilibrium in a Traffic Simulation Model. In: *International Journal of Modern Physics C* 9 (1998), Nr. 3, S. 393–408 - Zitiert auf den Seiten 13, 14 und 17

[HBS 2001] HBS: *Handbuch für die Bemessung von Straßenverkehrsanlagen*. Köln : Forschungsgesellschaft für Straßen- und Verkehrswesen - FGSV, 2001 - Zitiert auf den Seiten 26, 27, 28 und 30

[HCM 2000] HCM: *Highway Capacity Manual: 2000*. Transportation Research Board - TRB, 2000 - Zitiert auf Seite 26

[IVT 2003] IVT: *Verkehrstechnik Grundzüge - Teil Individualverkehr*. Bd. 6 - Verkehrssteuerung mit Lichtsignalanlagen. Eidgenössische Technische Hochschule Zürich, Institut für Verkehrsplanung und Transportsysteme, 2003 - Zitiert auf Seite 28

[MATSim 2007] MATSim: *Multi-Agent Transport Simulation.* Internetpräsenz unter http://www.matsim.org/, Eidgenössische Technische Hochschule Zürich, Technische Universität Berlin, 2007 - Zitiert auf den Seiten 10 und 18

[Nagel u. Schreckenberg 1992] Nagel, K. ; Schreckenberg, M.: A Cellular Automaton Model for Freeway Traffic. In: *J. Phys. I France* 2 (1992), Nr. 2221, S. 222–235 - Zitiert auf Seite 12

[RAS 2001] RAS: *Richtlinien für die Anlage von Straßen.* Köln : Forschungsgesellschaft für Straßen- und Verkehrswesen - FGSV, 2001 - Zitiert auf Seite 26

[RiLSA 1992] RiLSA: *Richtlinien für Lichtsignalanlagen.* Köln : Forschungsgesellschaft für Straßen- und Verkehrswesen - FGSV, 1992 - Zitiert auf Seite 75

[RVS 2008] RVS: *Richtlinien und Vorschriften für den Straßenbau.* Wien : Österreichische Forschungsgesellschaft Straße - Schiene - Verkehr - FSV, 2008 - Zitiert auf Seite 26

[Simon u. Nagel 1998] Simon, P. M. ; Nagel, K.: Simplified Cellular Automaton Model for City Traffic. In: *Phys. Rev. E* 58 (1998), Aug, Nr. 2, S. 1286–1295 - Zitiert auf Seite 44

[Simon u. a. 1999] Simon, Patrice M. ; Esser, Jörg ; Nagel, Kai: Simple Queueing Model Applied to the City of Portland. In: *International Journal of Modern Physics C* 10 (1999), S. 941–960 - Zitiert auf Seite 16

[SN640022 1999] SN640022: *Leistungsfähigkeit, Verkehrsqualität, Belastbarkeit; Knoten ohne Lichtsignalanlagen.* Schweizer Verband der Straßen- und Verkehrsfachleute - VSS, 1999 - Zitiert auf Seite 26

[SN640023a 2008] SN640023a: *Leistungsfähigkeit, Verkehrsqualität, Belastbarkeit; Knoten mit Lichtsignalanlagen.* Schweizer Verband der Straßen- und Verkehrsfachleute - VSS, 2008 - Zitiert auf Seite 26

[Westumfahrung 2008] Westumfahrung: *Westumfahrung Zürich.* Internetpräsenz des Bauprojekts unter http://www.westumfahrung.ch/, Baudirektion Kanton Zürich, 2008 - Zitiert auf Seite 9

[Wiedemann 1974] Wiedemann, R.: *Simulation des Straßenverkehrsflusses*, Instituts für Verkehrswesen der Universität Karlsruhe, Diss., 1974. – Schriftenreihe des Instituts, Vol. 8 - Zitiert auf Seite 12

Die selbständige und eigenhändige Anfertigung
versichere ich an Eides statt.

Berlin, den 29. September 2008

_____

Andreas Neumann