Anpassung des Desynchronisationsalgorithmus DESYNC für VANETs

Bachelorarbeit

von

Andre Ippisch

aus

Düsseldorf

vorgelegt am

Lehrstuhl für Rechnernetze und Kommunikationssysteme
Prof. Dr. Martin Mauve
Heinrich-Heine-Universität Düsseldorf

Juli 2012

Betreuer:
Daniel Baselt M. Sc.

Zusammenfassung

Im Rahmen dieser Arbeit wird der Desynchronisationsalgorithmus DESYNC vorgestellt, an Fahrzeug-Ad-hoc-Netzwerke (VANETs) angepasst und im Kontext der Fahrzeug-zu-Fahrzeug-Kommunikation ausgewertet.

DESYNC wurde als TDMA-Protokoll an der Harvard-Universität für kabellose stationäre Sensornetzwerke entwickelt. Der rundenbasierte Algorithmus weist jedem Knoten des Netzwerks den gleichen Zeitabschnitt der Runde zum Senden von Daten zu und passt sich dynamisch der Anzahl der Knoten im Netzwerk an. Deshalb ist er als Sicherungsschichtprotokoll für Fahrzeug-Ad-hoc-Netzwerke gut geeignet.

Der DESYNC-Algorithmus wurde von den Entwicklern in einer stationären Umgebung getestet. Die folgenden Untersuchungen und Auswertungen des an VANETs angepassten DESYNC-Algorithmus stellen fest, dass sich dieser auf einem verlustbehafteten, also VANET-ähnlichen, Kanal ähnlich gut verhält, wie es der Fall war, als der ursprüngliche Algorithmus in der noch nicht angepassten Form und Umgebung getestet wurde.

Es hat sich herausgestellt, dass sowohl das Beaconing als auch die Desynchronisation in Hinsicht auf die Fahrzeug-zu-Fahrzeug-Kommunikation sehr gut funktionieren. Für Kolonnenfahrten kommen Beacons schnell an, Fahrzeuge können sich schnell erkennen und der Verlust von Beacons ist bei geeigneter Rundenlänge und Fahrzeuganzahl gering. Eine kumulative Dichtefunktion zeigt, dass wenige Millisekunden ausreichen, um direkte Fahrzeugnachbarn mit Informationen zu versorgen. Die Anforderung von VANETs, mit einer 10-Hertz-Rate Informationen zu verbreiten, funktioniert mit der Rundenlänge von 100 ms gut, mit einer Rundenlänge von 50 ms kann sogar mit einer 20-Hertz-Rate gesendet werden und es werden nur minimal schlechtere Werte erzielt.

Neben dem Versand von Beacons wurde auch der Versand von Anwendungsdaten untersucht. Der von DESYNC vorgesehene Weg, Daten zu versenden, ist aufgrund des Verlusts auf dem Kanal nicht erfolgreich und stört zusätzlich das Beaconing. Es werden alternative Methoden vorgestellt, Anwendungsdaten zu versenden.

Danksagung

Besonderen Dank kommt meinem Betreuer Daniel Baselt zu Gute, der sich viel Zeit für mich genommen und unterstützt hat, wann und wo er nur konnte.

Und auch großen Dank an meine Freunde aus allen Ecken der Universität und an alle Kollegen im Rechnerlabor, die mir bei Fragen immer zur Verfügung standen, und natürlich an meine Freundin für die viele Motivation.

Für die Ausarbeitung dieser Arbeit wurde hauptsächlich freie Software genutzt, deshalb geht ein großer Dank an die Entwickler von ns-3, geany, LATEX, Ubuntu, Linux, gnuplot, inkscape und dia.

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

Nach Angaben des statistischen Bundesamts[1] gab es im Jahr 2011 über 2.3 Millionen Unfälle, 300000 davon mit Personenschaden, 4000 Menschen verloren ihr Leben. Entwickler auf der ganzen Welt arbeiten an intelligenten Transportsystemen für Autos, sodass der Verkehr nicht nur sicherer wird, sondern den Fahrern auch beim Fahren geholfen wird.

Hierbei soll die Fahrzeug-zu-Fahrzeug-Kommunikation helfen, bei der es wichtig ist, dass Fahrzeuge Informationen über sich, zum Beispiel ihren Standort oder ihre aktuelle Fahrtrichtung und Geschwindigkeit, an ihre Umgebung verbreiten, sodass sich alle anderen Fahrzeuge in der Nähe ein Bild dieser Umgebung machen können. Anwendungen können sich dieses Umgebungsbild zu Nutze machen und den Fahrer vor Gefahren warnen, auf einen Stau hinweisen und allgemein den Verkehr sicherer und effizienter machen. Ein wichtiges Thema im Bereich dieser kabellosen Netzwerke ist die Desynchronisierung von Knoten in einem Kanal, um einen kollisionsfreien Versand von Nachrichten zu ermöglichen. Diese Arbeit beschäftigt sich mit dem Algorithmus DESYNC, der diese Desynchronisierung in Fahrzeug-Ad-Hoc-Netzwerken (VANETs) ermöglichen soll.

In Kapitel 2 wird der Algorithmus DESYNC im Allgemeinen erklärt und es wird betrachtet, welche Auswertungen bisher gemacht worden sind. Kapitel 3 zeigt die Ände-

[1] www.destatis.de, Juni 2012

rungen, die notwendig sind, um DESYNC für VANETs anzupassen. In Kapitel 4 wird die Implementierung von DESYNC und des Kanals, auf welchem DESYNC laufen wird, erläutert. In Kapitel 5 werden die Auswertungen der Tests gezeigt, die mit DESYNC in VANET-Umgebungen durchgeführt worden sind. Kapitel 6 gibt eine Zusammenfassung der Arbeit und eine Aussicht auf zukünftige Arbeit.

1.1 Verwandte Arbeiten

V-DESYNC Der Algorithmus V-DESYNC [SCIR12] wurde entwickelt und auf der VTC Spring 2012 vorgestellt. Das Paper beschäftigt sich ebenfalls mit dem Problem der Desynchronisierung des Beaconing in VANETs und der wichtigen Anpassungseigenschaften an ein solches dynamisches Netzwerk. Zu diesem Zweck wurde der Algorithmus V-DESYNC entwickelt, welcher die Anzahl der Kollisionen beim Beaconing um 85% reduziert, ohne die Senderate zu reduzieren. Zum Zeitpunkt dieser Arbeit war das Paper noch nicht öffentlich verfügbar.

TRAMA Das Protokoll TRAMA [ROGLA06], ein energieeffizientes, kollisionsfreies Kanalzugriffsverfahren für kabellose Netzwerke, wurde parallel zu dieser Arbeit an VANETs angepasst [Bia12].

Sicherheit in kabellosen verteilten Systemen In seiner Dissertation [TM07] schreibt Marc Torrent Moreno über den Sicherheitsaspekt in VANETs. Hier werden Anforderungen der Anwendungsschicht an die Sicherungsschicht geschildert und verdeutlicht. Für die Kommunikation zwischen hintereinanderfahrenden Autos, am Beispiel der Situation, dass das vorherfahrende Fahrzeug abbremsen könnte, ist das Paper [BSM11] von Daniel Baselt, Björn Scheuermann und Martin Mauve der Univerität Düsseldorf eine interessante Lektüre für das Verständnis der Auswertungen in dieser Arbeit.

Kapitel 2

Der Algorithmus DESYNC

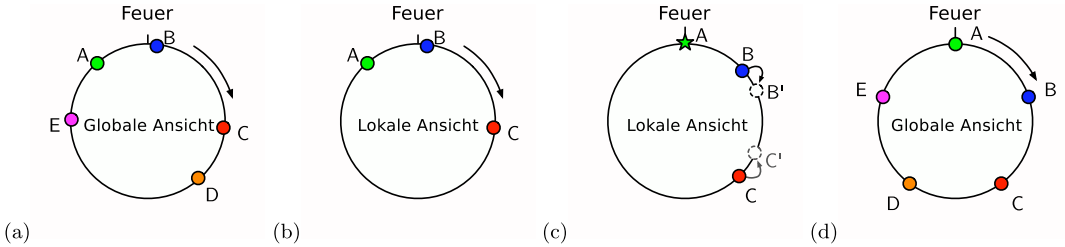
DESYNC [DRPN07] ist ein Desynchronisationsalgorithmus, welcher von Julius Degesys, Ian Rose, Ankit Patel und Radhika Nagpal[1] entwickelt wurde, um ein kollisionsfreies Versenden von Nachrichten in einem drahtlosen Netzwerk zu ermöglichen. Es ist ein Zeitmultiplexverfahren (TDMA, Time Division Mulitple Access), bei welchem die Daten periodisch in einem Rundlaufverfahren verschickt werden. Das rundenbasierte Verfahren funktioniert ohne globale Uhr und bezieht sein Wissen über andere Knoten in der Umgebung nur durch periodisch verschickte Nachrichten. Es passt sich der Anzahl der Knoten auf dem Kanal an, sodass jeder Knoten die gleiche Zeitspanne zum Senden von Daten hat, ohne Kollisionen zu verursachen.

2.1 Funktionsweise von DESYNC

Wir bezeichnen das Verschicken einer für den Algorithmus zur Desynchronisation wichtigen Nachricht als "Feuern", die Nachricht selbst als Beacon.

In jeder Runde feuert ein Knoten ein Beacon ab, welches alle anderen Knoten im Netzwerk hören können. Ein Knoten merkt sich die Ankunftszeit des vorherigen Beacons eines Knotens und des nachfolgenden Beacons eines Knotens und berechnet daraus den

[1] Division of Engineering and Applied Sciences, Harvard University, ssr@eecs.harvard.edu



(a) (b) (c) (d)

DESYNC Algorithmus: (a) Globale Ansicht von fünf Knoten, die noch nicht desynchronisiert sind. (b) Die lokale Nachbarschaftsansicht von Knoten B. (c) Wenn A feuert, dann weiß der Knoten, der unmittelbar zuvor gefeuert hat, Knoten B, jetzt die Position seiner beiden Nachbarn, C hat schon gefeuert und A feuert gerade. Knoten B kann nun also berechnen, wo er im Idealfall hätte positioniert sein müssen, B', und setzt sich an diese Stelle. Knoten C hat sich aber mittlerweile an Position C' positioniert, dies wissen die übrigen Knoten allerdings noch nicht. (d) Das System ist desynchronisiert. Alle Knoten sind in der Mitte ihrer Nachbarn, demnach müssen keine Berechnungen mehr gemacht werden, das System ist stabil.

Abbildung 2.1: DESYNC Algorithmus, sinngemäß aus dem DESYNC Paper übertragen

optimalen Zeitpunkt, an dem er selbst in der nächsten Runde feuern sollte. Der optimale Zeitpunkt ist dabei die Mitte aus den Zeiten der beiden Feuer-Nachrichten, wobei die Anpassung gewichtet, aus dem eigenen Feuerzeitpunkt und dem berechneten Mittelwert, erfolgt. Außerdem wird die Start- und Endzeit eines Zeitslots berechnet, in welchem der Knoten in der nächsten Runde andere Daten als das Beacon senden darf.

Abbildung 2.1 ist dem DESYNC-Paper [DRPN07] entnommen und ins Deutsche übersetzt worden und beschreibt die Desynchronisierung in einem System mit fünf Knoten.

2.2 Ursprüngliche Tests mit DESYNC

Die Entwickler des DESYNC-Algorithmus erbrachten den Beweis, dass für eine Knotenzahl kleiner als 500 die Desynchronisierung erfolgreich abläuft. Die 500 Knoten wurden lediglich als Grenze festgelegt, da der Beweis computergestützt durchgeführt wurde. Bei ausreichender Rundenlänge ist davon auszugehen, dass auch bei höherer Knotenzahl der gleiche Erfolg bei der Desynchronisierung erzielt wird.

Außerdem haben Auswertungen mit MATLAB eine quadratische Laufzeit belegt, die auch für DESYNC in der an VANETs angepassten Version gelten, da diese nichts an der Art der Berechnung ändert.

Die Entwickler des DESYNC-Algorithmus haben in Testdurchläufen die Länge einer Runde auf eine Sekunde gesetzt und feste Standorte für ihre Knoten genutzt. Als Knoten fungierten in den Tests kleine Rechner mit einem Transceiver. Getestet wurden nur die Desynchronisierung der Knoten und der Durchsatz der Anwendungsdaten. In dieser Arbeit werden diese Tests als die ursprünglichen Tests bezeichnet.

Bei der Desynchronisierung der Knoten, wie sie die Entwickler beobachtet haben, fällt auf, dass sich das System perfekt verhält. Der Verlust von Beacons ist nicht zu erkennen und das System läuft dauerhaft stabil, sobald es einmal desynchronisiert ist. Es ist nichts über die Entfernung zwischen den Knoten angegeben, allerdings erkennt man an den Messwerten, dass fast kein Verlust auf dem Kanal zu verzeichnen ist, weswegen eine geringe Distanz zwischen den Knoten angenommen werden kann.

Für die Auswertungen in dieser Arbeit wurde zum Vergleich mit den ursprünglichen Tests neben dem Kanal, der die Umgebung einer Autobahnfahrspur simuliert, ein zusätzlicher Kanal implementiert, auf welchem kein Verlust durch ein zu schwaches Signal auftritt, Paketkollisionen aber weiterhin möglich waren.

Kapitel 3

Anpassung von DESYNC für VANETs

Ein mobiles Ad-Hoc-Netzwerk ist ein Netzwerk, in welchem die Knoten sich selbst integrieren und konfigurieren, für die Fahrzeug-zu-Fahrzeug-Kommunikation wird dieses als VANET (Vehicular Ad Hoc Network) bezeichnet. Ziel dieser Arbeit ist es den Algorithmus DESYNC für VANETs anzupassen.

3.1 Anforderungen an DESYNC zur Anpassung für VANETs

Das Primärziel von VANETs ist der Austausch von Informationen über die Fahrzeuge. Die wichtigsten Daten, dazu gehören die aktuelle Position oder die Geschwindigkeit, werden zusammengefasst als Paket an alle sich in der Umgebung befindlichen Fahrzeuge geschickt und werden als Beacon bezeichnet.

Dabei sollen die Fahrzeuge ihre Informationen mindestens mit einer 10-Hertz-Rate verteilen, damit sich die Fahrzeuge in der Nähe immer ein aktuelles Bild ihrer Umgebung machen können. Eine hohe Rate ist wichtig für bestimmte Anwendungen, die auf dieses Umgebungsbild zurückgreifen, wie zum Beispiel ein Bremsassistent, der anhand der Informationen des vorherfahrenden Fahrzeugs selbstständig abbremst.

3.2 Erstellen eines Kanals

Um eine verkehrsähnliche Umgebung zu schaffen, nutzen wir den WLAN-Standard 802.11p, eine Erweiterung von 802.11a, welche aus 802.11 entstanden ist. 802.11p ist in [80210] definiert und bestimmt einige der in diesem Kapitel verwendeten Werte. 802.11p ist entwickelt worden, um WLAN in Fahrzeugen verfügbar zu machen und Fahrzeug-zu-X-Kommunikation zu ermöglichen bzw. zu verbessern.

In Europa wird für Fahrzeug-zu-Fahrzeug-Kommunikation ein 70 Mhz breites Frequenzband genutzt, zwischen 5855 und 8925 Mhz [80210]. Dieses ist unterteilt in 7 gleich große Bänder, von denen dieses Protokoll eines nutzen wird. Als Übertragungsgeschwindigkeit wird 6 Mbit/s ($= 6 * 10^6$ Bit/s) gewählt.

Ein Paket kommt bei einem Empfänger an, wenn die Signalstärke des Pakets groß genug ist und es nicht mit einem anderen Paket kollidiert. Die folgenden beiden Unterkapitel beschreiben das Verlustmodell auf dem verwendeten Kanal.

3.2.1 Entfernungsbedingter Verlust auf dem Kanal

Der entfernungsbedingte Verlust auf dem Kanal hängt von der Empfangssignalstärke der Pakete ab, die auf dem Kanal versendet werden. Fahrzeuge die im Frequenzband von 5.9 GHz senden, dürfen, sofern sie nicht Einsatzfahrzeuge sind, mit 33dbm (2 Watt) Signalstärke senden. Die Antenne eines Empfängers (Atheros-Transceiver [1]) kann Signalstärken über -91dbm empfangen, benötigt aber mindestens -71dbm Signalstärke (20dbm höher [80210]), um das Signal vom Grundrauschen unterscheiden zu können. Die Signalstärke nimmt bei zunehmender Entfernung zum Sender ab. Viele Faktoren, dazu gehören Hindernisse, Störgeräusche und Wetterverhältnisse, verschlechtern oder verbessern die Signalstärke zusätzlich. Entfernungsbedingter Verlust auf dem Kanal findet statt, wenn die Signalstärke eines Pakets unterhalb von -71dbm fällt.

[1] www.atheros.com, Juni 2012

Als Verlustmodell wurde ein Distanzmodell genutzt, welches auf einer Formel von Harald T. Friis basiert, kombiniert mit einem Fadingmodell, das die Nakagami-m Verteilung von M. Nakagami benutzt. Diese beiden Modelle sind zwei von mehreren Verlustmodellen, die im ns3-Simulator verfügbar sind.

Friis-Abschwächungs-Modell Das Friis-Modell [Fri46] berechnet die Stärke des Signals beim Empfänger des Pakets bei einer direkten Sichtverbindung, es werden Wellenausbreitung und Freiraumdämpfung berücksichtigt. Für die Berechnung der Signalstärke wird die Wellenlänge des Signals benötigt. Die Wellenlänge berechnet sich aus der Frequenz des Kanals und der Lichtgeschwindigkeit durch Luft.

$$\text{Wellenlänge: } \lambda = \frac{299708516 \frac{m}{s}}{5890000000 \frac{1}{s}} = 0.050884298\ m \approx 5.1\ cm$$

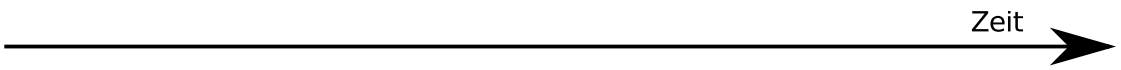
Das Friis-Verlustmodell berechnet die Signalstärke beim Empfänger nach der Distanz zwischen ihm und dem Sender des Pakets auf dem freien Kanal ohne Störungen.

Nakagami-Wahrscheinlichkeits-Modell Das Nakagami-Verlustmodell [YN00] ist eine Wahrscheinlichkeitsverteilung, die dem Verlustmodell von Friis Signalverstärkungen und Signalabschwächungen hinzufügt. Das Nakagami-Modell ist aus Versuchsergebnissen zusammengestellt worden und berücksichtigt Störungen wie Objekte im Sichtfeld, Überlagerungen, Mehrwegeempfang und Doppler-Effekt.

3.2.2 Frame Capture

Frame Capture, auch als Capture Effect bekannt, bezeichnet die Möglichkeit des Empfängers bei einer Kollision zweier Pakete jenes mit der höheren Signalstärke vollständig zu empfangen und nur das schwächere Paket zu verlieren, anstatt beide. Der Effekt wurde in [LKL+07] untersucht, wenn auch nur für 802.11a. Untersuchungen für 802.11p existieren bisher noch nicht.

Fünf Fälle werden in folgendem Abschnitt betrachtet.

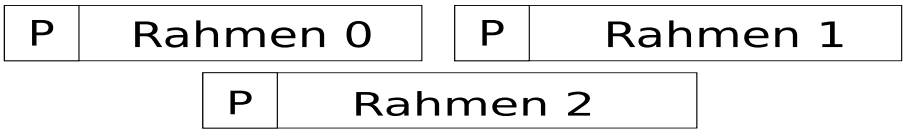


(a) Rahmen 1 zuerst, vor der Präambel von Rahmen 2

(b) Rahmen 1 zuerst, während der Präambel von Rahmen 2

(c) Rahmen 1 zuletzt, während der Präambel von Rahmen 2

(d) Rahmen 1 zuletzt, nach der Präambel von Rahmen 2, Empfänger fixiert auf Rahmen 2



(e) Rahmen 1 zuletzt, nach der Präambel von Rahmen 2, Empfänger nicht fixiert auf Rahmen 2

Abbildung 3.1: Verschiedene Fälle des Empfangs zweier Pakete bei Betrachtung von Rahmen 1

Für den Fall in Abbildung 3.1a, bei welchem die Präambel von Rahmen 1 schon empfangen wurde, bevor Rahmen 2 ankommt, muss Rahmen 1 mit höherer Signalstärke ankommen als Rahmen 2, damit es aufgegriffen werden kann. Für die Fälle 3.1b und 3.1c, bei denen zur Zeit der Präambelübertragung ein neuer Rahmen ankommt, kann Rahmen 1 aufgegriffen werden, wenn dieser mit einer 10 dB höheren Signalstärke ankommt. Für den Fall 3.1d, bei welchem der Empfänger schon auf das Empfangen von Rahmen 2 fixiert ist, muss der Rahmen 1 mit einer 10 dB höheren Signalstärke ankommen, damit er statt Rahmen 2 aufgegriffen wird. Ist der Empfänger noch nicht auf das Empfangen von Rahmen 2 fixiert, siehe Fall 3.1e, braucht Rahmen 1 eine 24 dB höhere Signalstärke, um aufgegriffen zu werden.

3.3 Anpassung von DESYNC

Wir betrachten den fünfschichtigen Internet-Protokollstapel, bestehend aus der Anwendungsschicht, Transportschicht, Netzwerkschicht, Sicherungsschicht und der Bitübertragungsschicht. Der Algorithmus DESYNC sorgt für den geordneten Austausch von Daten auf der Sicherungsschichtebene, darunter befindet sich der für VANETs erstellte Kanal. Die höheren Schichten werden in dieser Arbeit nicht näher betrachtet.

Da kein applikationsunabhängiges neues Protokoll für die Sicherungsschichtebene entwickelt werden soll, sondern eines speziell für VANETs, kann für die Grundinformationen der Fahrzeuge, also diejenigen, welche an alle Fahrzeuge in der Nähe übermittelt werden müssen, die Transport- und Netzwerkschicht übergangen werden, und das Beacon, das in der Sicherungsschichtebene für DESYNC genutzt wird, direkt mit den Fahrzeuginformationen versehen werden (cross-layer).

Weitere Daten, die Fahrzeuge austauschen wollen, Daten mit geringerer Priorität, sollen dennoch wie gewohnt über den Protokollstapel an ihr Ziel verschickt werden. Auch hier wird getestet, wie gut dies in der VANET-Umgebung funktioniert.

Die Rundenlänge muss von einer Sekunde auf höchstens 100 ms gesenkt werden, damit eine 10-Hertz-Rate erreicht werden kann.

Kapitel 4

Implementierung von DESYNC im ns3-Simulator

Der DESYNC-Algorithmus, angepasst an VANETs, wurde im ns-3-Netzwerksimulator implementiert. Zur Anpassung wurden einige Variablen, die in der ursprünglichen Implementierung oder den Tests genutzt wurden, angepasst, um die Situation innerhalb der Fahrzeug-zu-Fahrzeug-Kommunikation zu simulieren.

4.1 ns3

Die Simulationen werden mit Hilfe von ns-3 [ns3] durchgeführt. ns-3 ist ein ereignisorientierter Netzwerksimulator, implementiert in C++ und Python, und wurde in der Version 3.13 genutzt. ns-3 ist ein freies, quelloffenes Projekt und wird von Entwicklern auf der ganzen Welt weiterentwickelt und verbessert. Der ns-3 Simulator gibt Entwicklern die Möglichkeit, Netzwerkforschung in einer einfachen Umgebung zu simulieren.

Parameter	Ursprünglicher Wert	Genutzte(r) Wert(e)
Paketgröße	35 Byte	100 Byte
Präambellänge	(nicht genutzt)	32 µs / 24 Byte
Datenrate	250 kb/s	6 Mbit/s
Frequenzband	2.4 GHz	5.9 GHz
Übertragungsprotokoll	802.15.4	802.11p
Rundenlänge	1000 ms	1000, 500, 100, 50, 25 ms
alpha	0.95	0.95, 1.0, 0.75, 0.5, 0.25

Tabelle 4.1: Variablen in der ursprünglichen und aktuellen Implementierung

4.2 Variablenanpassung

Paketgröße Pakete in den ursprünglichen Tests waren 35 Byte groß, für VANETs müssen der aktuelle Ort als GPS-Koordinaten und der Status übertragen werden, die Beacons sind deshalb 100 Byte groß gewählt. Die 100 Byte beinhalten sowohl die Präambel als auch den Header des Pakets.

Präambel 802.11p schreibt eine Präambellänge von 32 µs vor. Das entspricht 24 Byte bei einer Datenrate von 6 Mbit/s.

Header In der Praxis wird im Header eines Beacons eine eindeutige Identifizierungsnummer des Fahrzeugs mitgeschickt, um dieses jederzeit wiedererkennen und von anderen unterscheiden zu können, dazu kann die MAC-Adresse verwendet werden. Innerhalb der Simulation steckt diese Information in den 100 Bytes des Pakets.

Rundenlänge Eine Runde bezeichnet die Zeitspanne, in welcher sich die Knoten desynchronisieren. Jeder Knoten erhält einen fairen Teil der Runde für seinen Slot und den Zeitpunkt zu dem er das Beacon sendet. Eine Runde war in den ursprünglichen Tests eine Sekunde lang. Im Kontext der Fahrzeug-zu-Fahrzeug-Kommunikation wird 10-Hertz-Beaconing angestrebt, deshalb wurde mit einer Sekunde sowie 500, 100, 50 und 25 Millisekunden getestet.

alpha Ein Knoten setzt sich nicht genau in die Mitte seiner Phasennachbarn sondern gewichtet mit seinem eigenen Feuern, auf diese Weise ist die Verschiebung nicht zu groß, sollte ein Paket zu spät ankommen oder verloren gehen. In den ursprünglichen Tests wurde 0.95 als Wert für alpha verwendet, das bedeutet, dass mit dem Faktor 0.95 die exakte Mitte der Phasennachbarn, nur mit Faktor 0.05 das eigene Feuern gewertet wird. Die Simulationen wurden mit dem Wert 0.95 durchgeführt, in Kapitel A.3 wurde der Einfluss verschiedener Werte von alpha auf die Messergebnisse untersucht.

Antennenabstände Bei zwei Fahrspuren sind die Antennen zweier nebeneinanderstehender Autos mindestens eine Fahrspurbreite entfernt, diese beträgt auf Autobahnen circa 3.5 Meter. Der Antennenabstand zweier Fahrzeuge auf einer Fahrspur wurde im Stau auf zehn Meter gesetzt, beim Fahren auf Sicherheitsabstand plus fünf Meter.

4.3 Implementierung

Für die Implementierung in ns3 wurden die beiden Klassen `SimpleNetDevice` und `SimpleChannel` geändert, erweitert und als Modul `desync` zu ns3 hinzugefügt, wobei das `Simple` zu `Desync` umbenannt wurde. `DesyncNetDevice` entspricht der MAC-Schicht und implementiert den DESYNC-Algorithmus. `DesyncChannel` entspricht dem kabellosen Kanal, auf welchem Pakete übertragen von Fahrzeug zu Fahrzeug verschickt werden. Weitere Klassen in diesem Modul sind `DesyncPhy`, welches der physikalischen Schicht des Protokollstapels entspricht und die Verbindung der MAC-Schicht mit dem Kanal ist, `DesyncMacState`, welche den Sende- und Empfangsstatus der physikalischen Schicht verwaltet und `DataTag`, welche für das Verschicken von Paketen genutzt wird. Neben der Hilfsdatei `desync-const.h`, in welcher die Konstanten definiert werden, wurde die Hilfsklasse `DesyncHelper` geschrieben, welche die obenstehenden Klassen miteinander verbindet. Ein vereinfachtes Klassendiagramm findet sich in Abbildung 4.1.

Die ausführbaren Hauptdateien befinden sich im Ordner `scratch` des Hauptverzeichnisses von `ns-3.13` und enthalten alle für die Auswertungen wichtigen Szenarien. Die-

se Dateien lauten `main.cc`, `loss.cc`, `maxcars.cc`, `twocolsmall.cc` und `twocol.cc`.

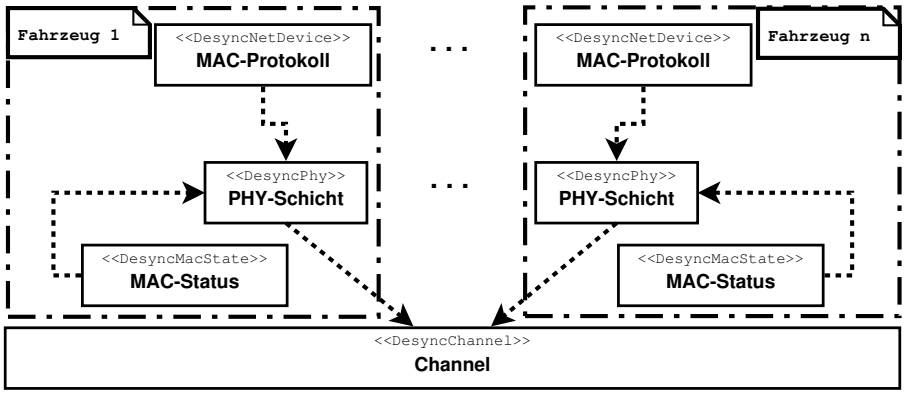


Abbildung 4.1: Klassendiagramm (vereinfacht)

4.3.1 MAC-Schicht

Auf der Sicherungsschicht ist der DESYNC-Algorithmus nach dem Pseudocode in Abbildung A.1, entnommen aus dem DESYNC-Paper [DRPN07], implementiert. Die Beschreibung des Algorithmus ist in 2.1 zu finden.

4.3.2 PHY-Schicht

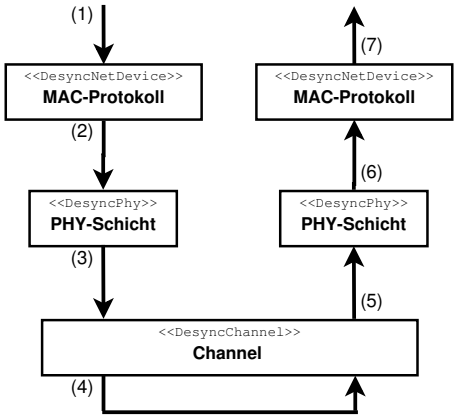
Die physikalische Schicht überträgt Pakete von der MAC-Schicht auf den Kanal oder vom Kanal auf die MAC-Schicht. Beim Übertragen der Pakete vom Kanal wird überprüft, ob das Paket eine Signalstärke von mehr als -71dbm aufweist, außerdem wird bei Empfang von meheren Paketen gleichzeitig mit Hilfe des Capture Effects, siehe 3.2.2, ermittelt, ob eines dieser Pakete empfangen wird.

4.3.3 Kanal

Der Kanal wurde entsprechend der Anpassungen des Kapitels 3.2 implementiert. Auf dem Kanal wird die Entfernung zwischen den Fahrzeugen und die Paketgröße benutzt, um die Übertragungs- und Ausbreitungsverzögerung zu berechnen. Die Empfangssignalstärke wird mit Hilfe der Verlustmodelle ermittelt, sodass die PHY-Schicht bestimmen kann, ob ein Paket ankommt oder nicht. Der Kanal kann auch so eingestellt werden, dass es zu keiner Signalabschwächung kommt und damit der Kanal verlustfrei ist. Trotzdem kann es auf der physikalischen Schicht noch zu Kollisionen kommen.

4.3.4 Versenden von Informationen

Das folgende Diagramm zeigt den Weg von Fahrzeuginformationen vom Sender zu jedem Empfänger.



(1) Die MAC-Schicht erhält die aktuellen Fahrzeuginformationen und verpackt sie in ein Paket.
(2) Die MAC-Schicht berechnet den Feuerzeitpunkt und reicht die Daten an die PHY-Schicht weiter.
(3) Die PHY-Schicht überträgt das Paket auf den Kanal, hier wird die Übertragungsverzögerung berechnet.
(4) Der Kanal berechnet für den Empfänger die Ausbreitungsverzögerung und die Signalabschwächung.
(5) Die PHY-Schicht nimmt das Paket entgegen, wenn die Signalstärke groß genug ist.
(6) Die PHY-Schicht gibt das Paket an die MAC-Schicht weiter, wenn es zu keiner Kollision in der PHY-Schicht kam.
(7) Die MAC-Schicht nutzt die Ankunftszeit des Pakets zum Setzen der DESYNC-Timer und gibt die Informationen an die höheren Schichten weiter.

Abbildung 4.2: Versenden von Informationen

Kapitel 5

Auswertung von DESYNC in VANETs

In diesem Kapitel werden die Tests, die mit dem angepassten DESYNC-Algorithmus durchgeführt wurden, vorgestellt. Die Auswertungen wurden in mehrere Teile gegliedert. In Kapitel 5.2 wird der Verlust auf dem Kanal in Abhängigkeit von der Entfernung zwischen zwei Fahrzeugen untersucht. Kapitel 5.3 zeigt die Desynchronisierung von Fahrzeugen auf diesem Kanal, Kapitel 5.4 das Beaconing von Kolonnen und Kapitel 5.5 das Beaconing zwischen zwei Fahrzeugen. Kapitel 5.6 untersucht den Anwendungsdatenversand zwischen Fahrzeugen und dessen Auswirkungen auf das Beaconing.
Eine Beschreibung, wie die Boxplot-Grafiken dieses Kapitels zu lesen sind, liefert das Kapitel A.2.

5.1 Verwendete Szenarien

Für das Auswertungskapitel werden folgende Szenarien betrachtet, die standardmäßig in einem verlustbehafteten Kanal stattfinden.

5.1.1 Eine-Kolonne-Szenario

Eine Anzahl von Fahrzeugen fährt auf einer Spur einer Autobahn mit einer konstanten Geschwindigkeit von 100 km/h. Die Fahrzeuge halten jeweils einen Sicherheitsabstand von 50 Metern zum vorausfahrenden Fahrzeug ein.

5.1.2 Zwei-Kolonnen-Szenario

Zwei Kolonnen mit je der gleichen Anzahl von Fahrzeugen fahren auf einer Spur der Autobahn. Die vordere Kolonne fährt mit 100 km/h, die hintere Kolonne fährt zu Beginn mit 250 km/h. Die Kolonnen sind zu Beginn 1500 Meter voneinander entfernt. Wenn die hintere Kolonne die vordere erreicht, nach circa 35 Sekunden, bremsen die Fahrzeuge auf 100 km/h ab und es entsteht eine große Kolonne. Alle Fahrzeuge halten zu jeder Zeit den Sicherheitsabstand zu ihren Vorgängern ein.

5.2 Verlustwahrscheinlichkeit durch Signalabschwächung

Je größer die Entfernung zweier Fahrzeuge ist, umso kleiner wird die Signalstärke des Pakets und umso größer ist die Wahrscheinlichkeit, dass dieses Paket verloren geht, welches von einem Knoten zum anderen geschickt wird. Die Grafik 5.1 zeigt die Wahrscheinlichkeit von Verlust in Abhängigkeit von der Entfernung bei Verwendung des Friis-Nakagami-Verlustmodells (siehe 3.2.1).

Wie zu erwarten steigt die Verlustwahrscheinlichkeit mit Zunahme der Entfernung an und konvergiert gegen 100 Prozent. Dennoch liegt die Empfangswahrscheinlichkeit bei einer Entfernung der Fahrzeuge von 1000 Metern noch über zehn Prozent.

Das Rauschen erklärt sich durch das Nakagami-Modell, welches dem Verlustmodell von Friis noch zufällige Abweichungen hinzufügt. Der Sprung bei 80 Metern Entfernung

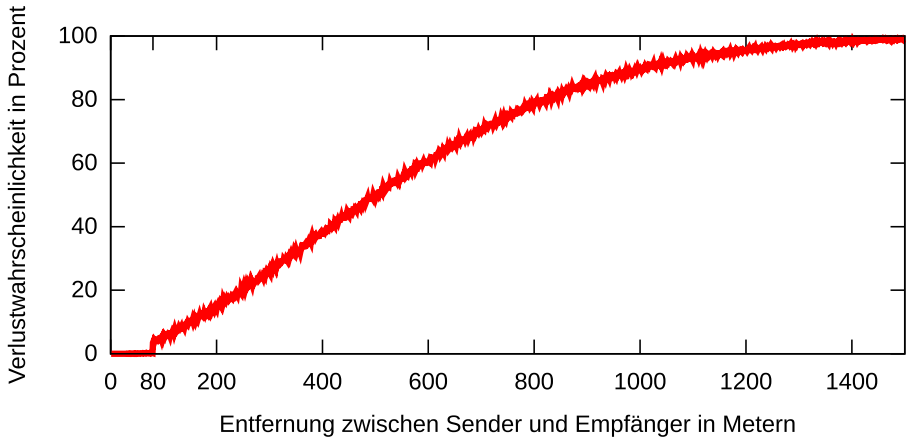


Abbildung 5.1: Auswertung der Verlustwahrscheinlichkeit von Paketen durch Signalabschwächung

zwischen den Fahrzeugen lässt sich durch einen Parameter des Nakagami-Modells erklären, der unterhalb einer 80-Meter-Entfernung einen anderen Wert besitzt als überhalb.

5.3 Desynchronisierung von Fahrzeugen

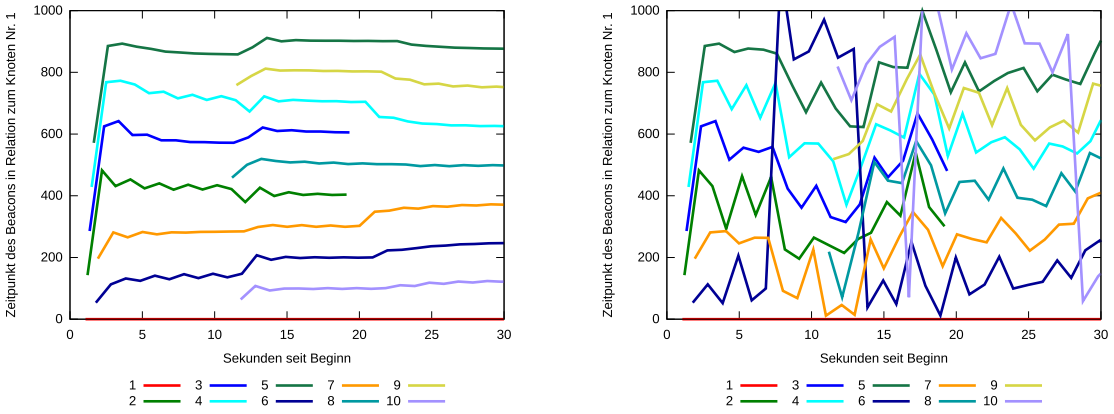
In diesem Abschnitt wird die Desynchronisierung von Fahrzeugen betrachtet. Das Desynchronisieren ist der Hauptbestandteil vom Algorithmus DESYNC, weshalb es wichtig ist zu sehen, ob dies auch im Fahrzeugkontext funktioniert.

Im ersten Unterabschnitt wird betrachtet, wie sich die Fahrzeuge desynchronisieren, wenn einzelne Fahrzeuge zu einer Kolonne stoßen oder wenn Fahrzeuge die Kolonne verlassen. Im zweiten Abschnitt wird betrachtet, wie sich zwei Kolonnen desynchronisieren, die aufeinandertreffen.

5.3.1 Desynchronisierung beim Hinzufügen oder Entfernen eines Fahrzeugs

Im Gegensatz zu den ursprünglichen Tests mit DESYNC ist es nicht wichtig, wie schnell sich Knoten zum Start der Simulation desynchronisieren, sondern wie sich ein schon desynchronisiertes System verhält, wenn sich diesem System Knoten anschließen oder es verlassen. Dazu wird das Eine-Kolonne-Szenario 5.1.1 betrachtet. Dabei sind sieben Fahrzeuge zu Beginn in der Kolonne, drei Fahrzeuge kommen hinzu, nachdem sich die sieben Fahrzeuge desynchronisiert haben. Nachdem die zehn Fahrzeuge desynchronisiert sind, verlassen zwei Fahrzeuge die Simulation.

Bei der Simulation mit einer Sekunde Rundenlänge treten nach zwölf Sekunden drei neue Fahrzeuge dem Kanal bei, nach 18 Sekunden verlassen zwei Fahrzeuge den Kanal.



(a) 1000 ms Rundenlänge, in verlustfreiem Kanal

(b) 1000 ms Rundenlänge, in verlustbehaftetem Kanal

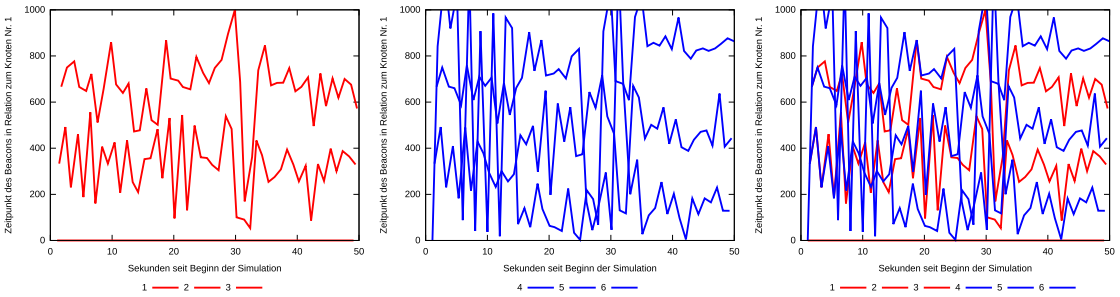
Abbildung 5.2: Desynchronisierung beim Hinzufügen oder Entfernen eines Fahrzeugs, 1000 ms Rundenlänge

Die Simulation wurde in einem verlustfreien Kanal und in einem verlustbehafteten Kanal durchgeführt. In beiden Fällen sieht man eine erfolgreiche Desynchronisierung. Die Ausschläge in Grafik 5.2b lassen sich dadurch erklären, dass bei Paketverlust die Desynchronisierung kurzzeitig gestört ist. Die Desynchronisierung erfolgt dennoch, wenn auch nicht so gleichmäßig wie in dem verlustfreien Kanal.

Der gleiche Test wurde auch für eine Rundenlänge von 100 ms und 50 ms durchgeführt. Grafiken siehe Anhang in A.4.1.

5.3.2 Desynchronisierung beim Zusammentreffen von zwei Fahrzeugkolonnen

Die Desynchronisierung wurde auch beim Zusammentreffen von zwei Fahrzeugkolonnen mit drei Fahrzeugen pro Kolonne, wie im Zwei-Kolonnen-Szenario 5.1.2 beschrieben, getestet.



(a) Betrachtung der ersten Kolonne (b) Betrachtung der zweiten Kolonne (c) Betrachtung beider Kolonnen

Abbildung 5.3: Desynchronisierung zweier Kolonnen bei einer Rundenlänge von 1000 ms

In den ersten 35 Sekunden kommen sich die Kolonnen immer näher, bis sie dann gemeinsam als eine Kolonne weiterfahren. In den Grafiken 5.3a und 5.3b sieht man, dass sich die Fahrzeuge in den ersten Sekunden nach Simulationsbeginn innerhalb ihrer Kolonne desynchronisieren, allerdings Beacons der jeweils anderen Kolonne erhalten und die Desynchronisierung nach und nach auf die andere Kolonne ausgeweitet wird. Je näher sich die Kolonnen kommen, umso mehr Beacons kommen an, sodass schon nach 18 Sekunden sehr gut erkennbar ist, dass sich alle sechs Fahrzeuge desynchronisieren.

Der gleiche Test wurde auch für eine Rundenzeit von 50 ms durchgeführt. Grafik siehe Anhang in A.4.4.

5.4 Beaconing

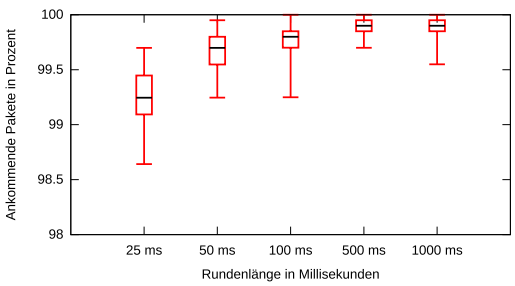
Unter Beacons versteht man die Datenpakete, die für die Berechnung der Desynchronisierung innerhalb des Algorithmus DESYNC genutzt werden. Der Algorithmus berechnet dem Fahrzeug einen Zeitslot, in welchem es theoretisch noch andere, nicht priorisierte Anwendungsdaten, versenden kann. Die wichtigsten Daten, die ein Fahrzeug anderen Fahrzeugen mitteilen muss, werden im Beacon, nicht als Anwendungsdaten, verschickt. Deshalb wird in den folgenden Abschnitten zunächst das Beaconing betrachtet und das Verschicken der Anwendungsdaten erst in Kapitel 5.6.

Unterabschnitt 5.4.1 beschäftigt sich mit dem Beaconing in einer Kolonne, Unterabschnitt 5.4.2 mit dem Beaconing zweier Kolonnen, die aufeinandertreffen, und 5.4.3 mit dem Beaconing in Stausituationen.

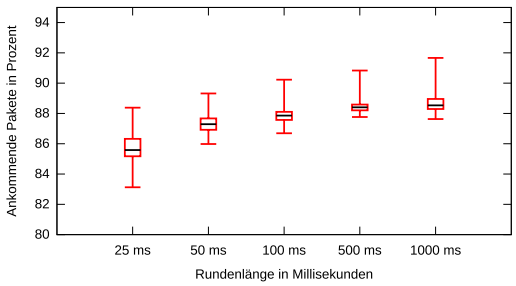
5.4.1 Auswertung des Beaconing in einer Fahrzeugkolonne

Betrachtet wird das Eine-Kolonne-Szenario 5.1.1 mit 50 Fahrzeugen.

Bei gleicher Anzahl von Runden, aber verschiedenen Rundenlängen, wird gemessen, wie viele Pakete bei Empfängern in bis zu 100 und bis zu 300 Metern Entfernung ankommen. Die Grafiken 5.4 zeigen den prozentualen Anteil der verschickten Pakete, die beim Empfänger angekommen sind.



(a) 100 Meter Empfangsreichweite



(b) 300 Meter Empfangsreichweite

Abbildung 5.4: Auswertung des Beaconing einer Kolonne bei Empfangsreichweite von 100 und 300 Metern

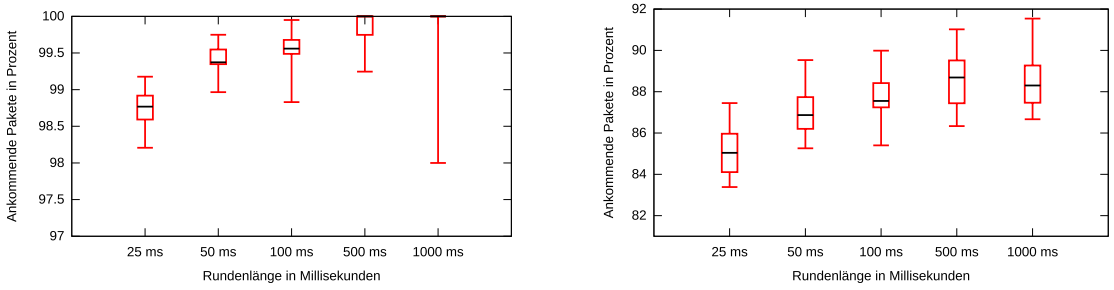
Wie in den Grafiken zu sehen ist, ist die Wahrscheinlichkeit, dass ein Beacon ankommt, umso größer, je länger die Runde und je kleiner die Entfernung zwischen Sender und Empfänger ist. Größere Rundenzeiten, also größere Sendeintervalle, bedeuten weniger Kollisionen auf dem Kanal.

Bei 300 Meter Empfangsreichweite, siehe Abbildung 5.4b, und 50 ms Rundenlänge liegt der maximale Prozentwert bei 89, der minimale bei 86 Prozent. Das bedeutet, dass mindestens einer von 50 Sendern nur 86 Prozent seiner Pakete erfolgreich an Empfänger in 300 Meter Entfernung verschickt hat, mindestens einer von 50 Sendern 89 Prozent seiner Pakete erfolgreich verschickt hat. Die Prozentwerte der übrigen Sender liegen dazwischen, wobei der Median bei 87 Prozent liegt.

Der gleiche Test wurde auch für Entfernungen bis 500 Meter und 1000 Meter durchgeführt. Grafiken siehe Anhang in A.4.2.

5.4.2 Auswertung des Beaconing beim Zusammentreffen zweier Kolonnen

Die gleiche Auswertung wie zuvor wurde auch für das Zwei-Kolonnen-Szenario 5.1.2 mit zehn Fahrzeugen pro Kolonne durchgeführt.



(a) 100 Meter Empfangsreichweite (b) 300 Meter Empfangsreichweite

Abbildung 5.5: Auswertung des Beaconing zweier Kolonnen bei Empfangsreichweite von 100 und 300 Metern

Die Werte sind ähnlich zu denen, die die Auswertung des Beaconings in einer Fahrzeugkolonne geliefert hat. Das ist damit zu erklären, dass sich zwei Kolonnen, wenn

sie langsam aufeinandertreffen, schnell desynchronisieren und beim Erreichen der Empfangsreichweite schon nah beieinander sind.

Der gleiche Test wurde auch für Entfernungen bis 500 Meter und 1000 Meter durchgeführt. Grafiken siehe Anhang in A.4.3.

5.4.3 Auswertung des Beaconing in Stausituationen

Die folgende Auswertung zeigt, wie sich das Protokoll verhält, wenn sich eine große Anzahl von Fahrzeugen nah beieinander aufhält, wie es bei einem Stau der Fall wäre. Ausgehend von einem Sicherheitsabstand von 5 Metern und einer Fahrspurbreite von 3.5 Metern wird ein Stau auf einer vierspurigen Autobahn simuliert. Bei einer Anzahl von 300 Fahrzeugen entsteht ein Stau von 740 Metern Länge, bei 500 Fahrzeugen ein Stau von 1240 Metern Länge und bei 1000 Fahrzeugen ein Stau von 2490 Metern Länge.

Für verschiedene Rundenlängen wurden die ankommenden Beacons im Vehältnis zu den gesendeten gemessen.

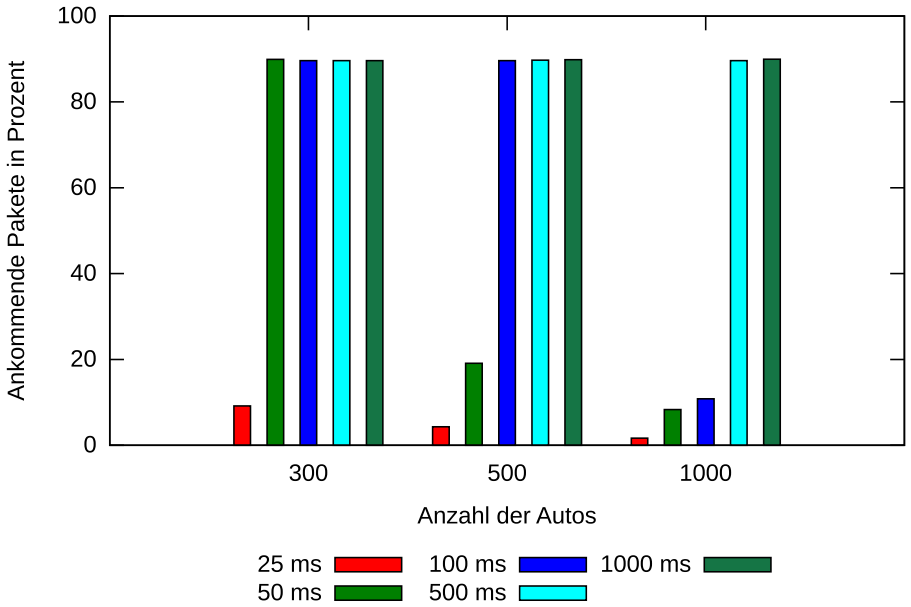


Abbildung 5.6: Auswertung des Beaconing in Stausituationen

Bei gleicher Rundenlänge erkennt man, dass weniger Beacons ankommen, wenn mehr Fahrzeuge beieinander stehen. Dieser negative Effekt wird dadurch gelindert, dass bei zunehmender Autozahl das durchschnittliche Tempo sinkt und die Fahrzeuge weniger schnell ihre Informationen aus der Umgebung benötigen.

5.5 Beaconing zwischen zwei Fahrzeugen

Dieses Kapitel führt die Auswertungen des Beaconing aus dem letzten Kapitel fort, wobei die Betrachtung auf den Austausch von Beacons zwischen zwei Fahrzeugen beschränkt wird.

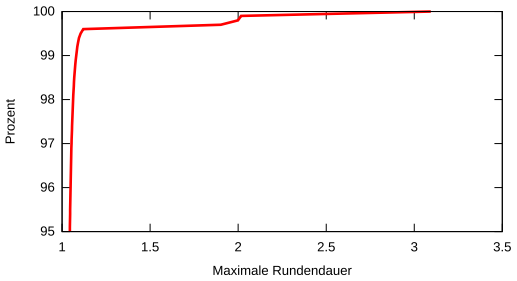
Die Unterkapitel 5.5.1 und 5.5.2 beschäftigen sich mit der Anzahl von Runden bis ein Beacon beim Empfänger angekommen ist. Unterkapitel 5.5.3 untersucht die benötigte Rundenlänge, sodass wenig Paketverlust bei hintereinanderfahrenden Fahrzeugen auftritt. Unterkapitel 5.5.4 betrachtet erneut die Situation zweier Kolonnen die aufeinander treffen und zeigt, wie schnell diese sich erkennen.

5.5.1 Rundenanzahl bis zur Ankunft eines Beacons

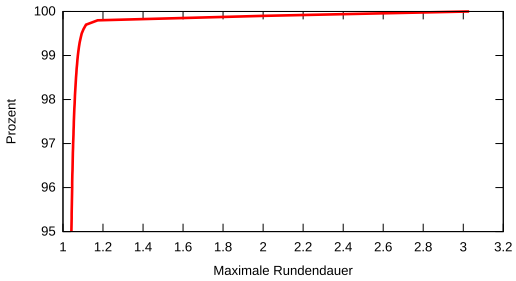
Durch den verlustbehafteten Kanal ist es bei jedem Sendeereignis möglich, dass ein Paket, welches das sendende Fahrzeug verschickt, nicht bei einem bestimmten Empfänger ankommt.

In dieser Simulation werden aus dem Ein-Kolonnen-Szenario 5.1.1 mit 50 Fahrzeugen nur diejenigen betrachtet, die hinter sich ein Fahrzeug haben. Betrachtet wird, wie viele Runden ein Beacon braucht, um beim Nachfolger anzukommen. Dabei benötigt es zum Beispiel zwei Runden, wenn ein Beacon verloren gegangen ist, drei Runden, wenn zwei Beacons hintereinander verloren gingen.

Die Auswertung ist als kumulative Dichtefunktion angegeben. So kann anhand des geforderten Prozentwerts die maximale Rundenanzahl abgelesen werden, die zum erfolgreichen Beaconaustausch benötigt wird.



(a) Kumulative Dichtefunktion bei einer Rundenlänge von 50 ms



(b) Kumulative Dichtefunktion bei einer Rundenlänge von 100 ms

Abbildung 5.7: Kumulative Dichtefunktion der Rundenanzahl bis ein Beacon ankommt

Will der Betrachter zu 99 Prozent sicher sein, dass ein Beacon ankommt, so muss bei 50 ms Rundenlänge 1.08053 Runden, also 54 ms gewartet werden, bei 100 ms Rundenlänge 1.07414 Runden, also 107.4 ms.

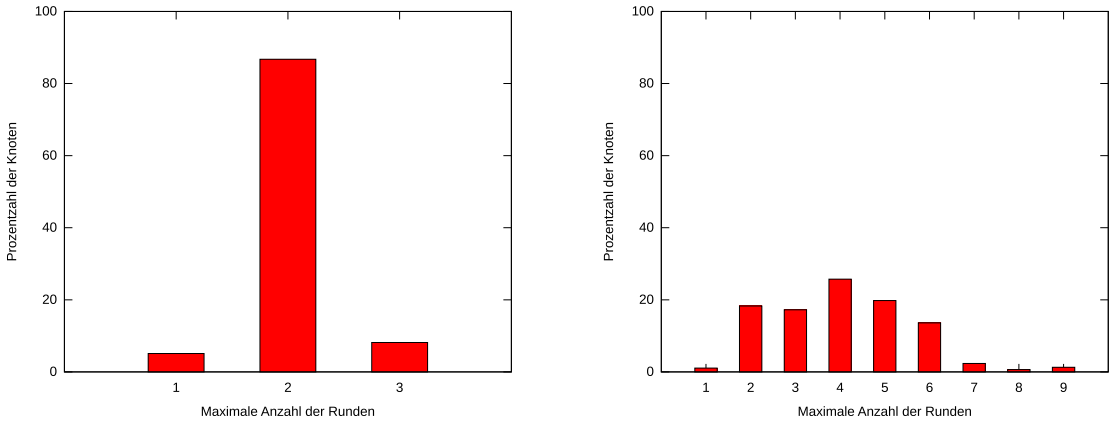
Dies bedeutet, dass ein schneller und erfolgreicher Informationsaustausch zwischen den Fahrzeugen gewährleistet ist.

Der gleiche Test wurde auch für Rundenlängen von 25ms, 500ms und 1000ms durchgeführt. Grafiken siehe Anhang in A.4.6.

5.5.2 Maximum der Rundenanzahl bis zur Ankunft eines Beacons

Ebenfalls von Interesse ist wie viele Runden es maximal dauert, bis ein Beacon bei einem Empfänger ankommt, der zuvor durch einen oder mehrere Paketverluste kein Beacon erhalten hat. Betrachtet wird das Ein-Kolonnen-Szenario 5.1.1 mit 50 Fahrzeugen.

Die Grafiken 5.8 zeigen für verschiedene Empfangsradien die maximale Rundenanzahl, die es gedauert hat, bis ein Beacon, das ein Fahrzeug gesendet hat, beim Empfänger angekommen ist.



(a) 100 Meter Empfangsreichweite (b) 300 Meter Empfangsreichweite

Abbildung 5.8: Auswertung der maximalen Runden, bis der Empfang eines Paketes eintritt, bei 50 ms Rundenlänge

Grafik 5.8a zeigt, dass 90 Prozent der Fahrzeugpaare maximal zwei Runden benötigen, um Beacons untereinander auszutauschen. 3 Prozent der Fahrzeugpaare benötigen nur eine Runde, 7 Prozent brauchen maximal drei Runden.

Für hintereinander fahrende Fahrzeuge sind diese Werte sehr gut, da gesichert ist, dass diese Fahrzeuge die wichtigsten Informationen ihrer Umgebung schnell erhalten. Ein höherer Empfangsradius von 300 Metern bewirkt, dass bei manchen Fahrzeugpaaren acht Runden hintereinander ein Beaconverlust auftritt, wodurch die maximale Rundenanzahl bis zum erfolgreichen Empfangen auf neun Runden steigt.

Der gleiche Test wurde auch für Rundenlängen von 25 ms und 100 ms durchgeführt. Grafiken siehe Anhang in A.4.5.

5.5.3 Geeignete Rundenlängen für das Beaconing

Betrachtet wird ein modifiziertes Ein-Kolonnen-Szenario 5.1.1 mit 50 Fahrzeugen, bei welchem der Sicherheitsabstand a zwischen den Fahrzeugen in Abhängigkeit von der Rundenlänge r abhängt. Die Geschwindigkeit der Fahrzeuge beträgt v = 100 km/h.

$$\text{Sicherheitsabstand: } a = r\ [ms] * v\ [\frac{m}{s}] = r\ [ms] * 27.78\ \frac{m}{s}$$

Der so berechnete Sicherheitsabstand a würde zum Abbremsen ausreichen, wenn ein Beacon nicht ankommt. Untersucht wird, wie viel Prozent der Beacons beim nachfolgenden Fahrzeug ankommen, wenn der Sicherheitsabstand in Abhängigkeit von der Rundenlänge gewählt ist.

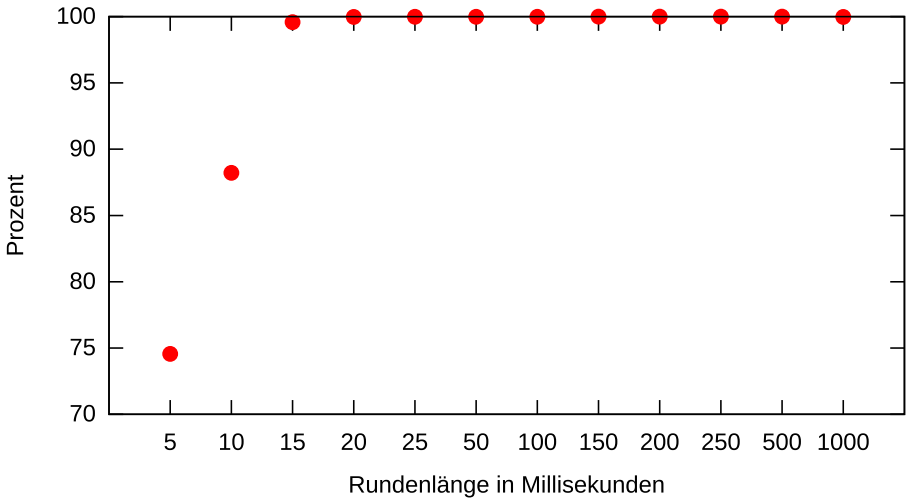


Abbildung 5.9: Auswertung von geeigneten Rundenlängen für das Beaconing

Ab einer Rundenlänge von 15 Millisekunden, was einem Sicheheitsabstand von nur 42 cm entspricht, kommen die Beacons zu über 99 Prozent beim nachfolgenden Fahrzeug an.

5.5.4 Kolonnenerkennen

Betrachtet wird das Zwei-Kolonnen-Szenario 5.1.2 mit zehn Fahrzeugen pro Kolonne.

Die Grafik 5.10 zeigt die Pakete, die das erste Fahrzeug der hinteren Kolonne vom letzten Fahrzeug der vorderen Kolonne erfolgreich empfängt, in Abhängigkeit von der Entfernung der beiden Fahrzeuge, für verschiedene Rundenlängen.

Bei einer Rundenlänge von 1000 ms erkennt das hintere Fahrzeug das vordere, wenn es noch knapp 1000 Meter entfernt ist, 22 Sekunden vor einem möglichen Aufprall. Bei niedrigeren Rundenlängen ist zwar die Verlustwahrscheinlichkeit höher, da mehr Kollisionen auftreten können, aber durch die größere Anzahl versendeter Beacons in der

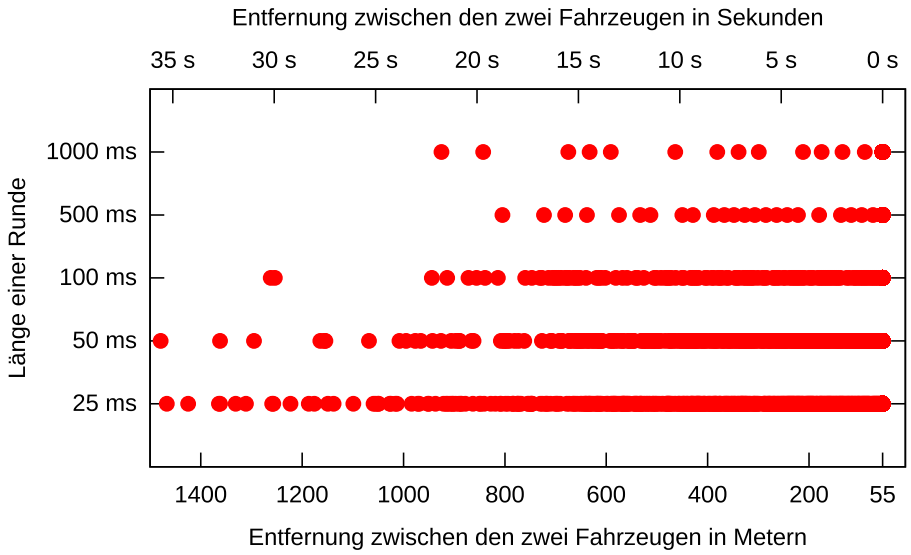


Abbildung 5.10: Erkennen von Fahrzeugen beim Zusammentreffen von Kolonnen

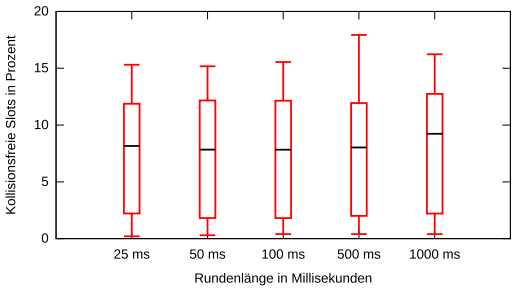
selben Zeit ist die Wahrscheinlichkeit, dass ein Paket durchkommt, teilweise höher. Es ist gut zu sehen, dass ein vorausfahrendes Fahrzeug schnell erkannt wird.

5.6 Slotting und Anwendungsdatenversand

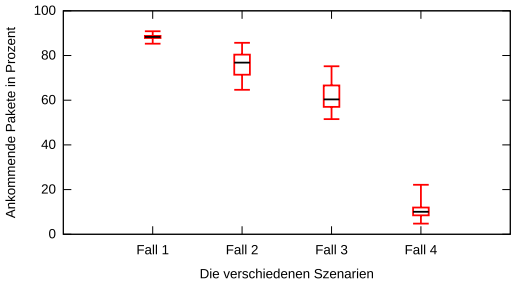
Der Algorithmus DESYNC ist ursprünglich darauf ausgelegt sich über das Beaconing zu desynchronisieren und Anwendungsdaten in bestimmten Zeitslots zu versenden, die er für jede Runde berechnet. In den ursprünglichen Tests von DESYNC wurde sowohl Beaconing als auch das Verschicken von Anwendungsdaten in einem verlustfreien Kanal ausgeführt. Hier wird nun das Verschicken der Anwendungsdaten in einem verlustbehaftetem Kanal simuliert.

Es wurde erneut die Kolonnenfahrt mit 50 Autos auf der Autobahn simuliert, sowohl im verlustfreien Kanal, als auch im verlustbehafteten.

Im verlustfreien Kanal überschneiden sich keine Slots, alle Slots können komplett zum Datenversand verwendet werden. Mit Prozentwerten unter 20 Prozent ist das Slotting in einem verlustbehaftetem Kanal, siehe Abbildung 5.11a, nicht gut ausgefallen.



(a) Auswertung des Slottings



(b) Auswertung des Beaconing bei Datenversand auf dem Kanal

Abbildung 5.11: Auswertung des Datenversands

Alle bisherigen Simulationen in dieser Arbeit wurden ohne Datenversand innerhalb der Slots durchgeführt. Falls aber Anwendungsdaten in diesen Slot versendet werden, kommt es nicht nur zu Kollisionen der Anwendungsdaten, sondern auch zu Kollisionen dieser mit Beacons. Durch den daraus resultierenden Verlust an Beacons wird das Desynchronisieren gestört und die Werte der Auswertungen in den letzten Kapiteln werden nicht mehr erreicht. Zur Verdeutlichung wurde das Eine-Kolonne-Szenario mit 50 Fahrzeugen durchgeführt. Anwendungsdaten wurden zusätzlich zu den Beacons auf verschiedene Arten mitverschickt. Wir betrachten in Abbildung 5.11b das Beaconing zwischen Fahrzeugen bei 50 ms Rundenlänge und einer Empfangsreichweite von 300 Metern. Fall 1 zeigt das Beaconing, wenn keine Anwendungsdaten verschickt werden bei einer Beacongröße von 100 Byte. Fall 2 und 3 zeigt das Beaconing, wenn die Fahrzeuge ihre Anwendungsdaten im Beacon mitschicken, dabei wurden in Fall 2 500 Byte Daten mitgeschickt, in Fall 3 wurden 1000 Byte Daten mitgeschickt. Fall 4 zeigt das Beaconing der Fahrzeuge, wenn alle Fahrzeuge innerhalb ihrer Slotzeiten auch Anwendungsdaten verschicken.

Je größer die Beacon-Größe gewählt ist, umso weniger Beacons kommen an, es kommt zu mehr Kollisionen. Wird der Datenversand über das Slotting durchgeführt, kommen nur noch unter 20 Prozent der Beacons an. Der Datenversand über Nutzdaten in Beacons ist vorzuziehen.

Kapitel 6

Fazit und Ausblick

6.1 Fazit

Ziel der vorangegangenen Untersuchungen und Auswertungen war es festzustellen, wie sich der an VANETs angepasste DESYNC-Algorithmus auf einem verlustbehafteten Kanal verhält. Dabei ist es wichtig, dass das Beaconing und die Desynchronisation ähnlich gut funktionieren wie es der Fall war, als der ursprüngliche Algorithmus in der noch nicht angepassten Form und Umgebung getestet wurde.

Es hat sich herausgestellt, dass sowohl das Beaconing als auch die Desynchronisation im Hinblick auf die Fahrzeug-zu-Fahrzeug-Kommunikation sehr gut funktionieren. Für Kolonnenfahrten kommen Beacons schnell an, Fahrzeuge können sich schnell erkennen und der Verlust von Beacons ist bei geeigneter Rundenlänge und Fahrzeuganzahl gering. Die kumulative Dichtefunktion zeigt, dass wenige Millisekunden ausreichen, um direkte Fahrzeugnachbarn mit Informationen zu versorgen.

Die Anforderung, in einer 10-Hertz-Rate Informationen zu verbreiten, funktioniert mit der Rundenlänge von 100 ms gut, mit einer Rundenlänge von 50 ms kann sogar mit einer 20-Hertz-Rate gesendet werden und es werden nur minimal schlechtere Werte erzielt. Für viele Autos in naher Umgebung, wie es bei einem Stau der Fall wäre, sind lange Rundenzeiten geeigneter, damit es beim Beaconing zu weniger Kollisionen kommt. Durch

den geringeren Bedarf an Beacons während eines Staus kann aber auch hier eine Rundenlänge von 50 ms gewählt werden.

Der Versand von Anwendungsdaten, die nicht in Beacons enthalten sind, war hingegen nicht erfolgreich, da sich die dafür verwendeten Zeitslots überschnitten und so gegenseitig störten. Dadurch wird gleichzeitig auch das Beaconing gestört. Möglichkeiten dies zu ändern werden im Absatz 6.2.2 erläutert.

6.2 Ausblick

6.2.1 Praxisversuche

Praxisversuche sind teuer, genau deshalb werden Simulationen gemacht. Dennoch müssen einige Testläufe in der Praxis durchgeführt werden. 802.11p ist bisher fast gar nicht praxisbezogen ausgewertet worden.

6.2.2 Datenversand, nicht Beaconing

DESYNC wurde auf der MAC-Schicht implementiert und erstellt dort die Beacons mit den neuesten Informationen vom Fahrzeug. Die Schnittstelle zur Netzwerkschicht und die Implementierung vom Versand der Daten, die von der Netzwerkschicht kommen, sind teilweise vorhanden, wurden aber nur teilweise genutzt. Die Auswertungen in dieser Arbeit zum allgemeinen Datenversand bezogen sich nur auf die Berechnungen der Slots, in denen DESYNC senden darf. Wie man in Kapitel 5.6 sieht, klappt das Slotting nicht sehr gut, wenn der Kanal nicht verlustfrei ist.

Obwohl das Slotting und der Datenversand nicht Bestandteil dieser Arbeit waren, wurden zwei Möglichkeiten überlegt, Datenversand in das Protokoll miteinzubauen.

Beacon-Größe Ein Beacon enthält, neben der Präambel und dem Header, die notwendigen Informationen der Fahrzeuge, die direkt in die Sicherungsschicht gespeist werden. Über die Anwendungsschicht übermittelte Daten könnte man in einem zusätzlichen Datenteil des Beacons auf dem Kanal übertragen. Kapitel 5.6 zeigt, dass größere Beacons zu mehr Kollisionen führen und dadurch das Beaconing schlechter funktioniert.

Zweiter Knoten Ein Fahrzeug kann sich nicht nur als ein Knoten ausgeben, sondern als zwei Knoten. Bekommt das MAC-Schicht-Protokoll Daten von der Anwendungsschicht, dann schickt es pro Runde zwei Beacons. Eines mit den priorisierten Fahrzeugdaten und ein weiteres mit den Anwendungsdaten. Für andere Fahrzeuge auf dem Kanal funktioniert die Desynchronisierung wie zuvor, für diese sieht es so aus, als wäre ein zusätzliches Fahrzeug unterwegs. Anhand des Headers wissen die Fahrzeuge aber, dass sie den zusätzlichen Knoten nicht weiter betrachten müssen.

Literaturverzeichnis

[80210] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. In: *IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, and IEEE Std 802.11w-2009)* (2010), 15, S. 1 –51. http://dx.doi.org/10.1109/IEEESTD.2010.5514475. DOI 10.1109/IEEESTD.2010.5514475.

[Bia12] BIALON, Raphael: *Anpassung von TRAMA für VANETs*, Heinrich-Heine-Universität in Düsseldorf, Bachelorarbeit, Juni 2012

[BSM11] BASELT, D.; SCHEUERMANN, B.; MAUVE, M.: A top-down approach to inter-vehicle communication (Poster). In: *Vehicular Networking Conference (VNC), 2011 IEEE* IEEE, 2011, S. 206–213.

[DRPN07] DEGESYS, Julius; ROSE, Ian; PATEL, Ankit; NAGPAL, Radhika: Desync: Selforganizing desynchronization and tdma in wireless sensor networks. In: *In Proc. Conference on Information Processing in Sensor Networks*, 2007.

[Fri46] FRIIS, H.T.: A Note on a Simple Transmission Formula. In: *Proceedings of the IRE* 34 (1946), may, Nr. 5, S. 254 – 256. http://dx.doi.org/10.1109/JRPROC.1946.234568. DOI 10.1109/JRPROC.1946.234568. ISSN 0096–8390.

[LKL+07] LEE, Jeongkeun; KIM, Wonho; LEE, Sung-Ju; JO, Daehyung; RYU, Jiho; KWON, Taekyoung; CHOI, Yanghee: An experimental study on the capture effect in 802.11a networks. In: *Proceedings of the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*. New York, NY, USA : ACM, 2007 (WinTECH '07). ISBN 978–1–59593–738–4, 19–26.

[ns3] *ns-3*. `http://www.nsnam.org`, Abruf: 25. Juni 2012.

[ROGLA06] RAJENDRAN, Venkatesh; OBRACZKA, Katia; GARCIA-LUNA-ACEVES, J. J.: Energy-efficient, collision-free medium access control for wireless sensor networks. In: *Wirel. Netw.* 12 (2006), Februar, Nr. 1, 63–78. `http://dx.doi.org/10.1007/s11276-006-6151-z`. DOI 10.1007/s11276–006–6151–z. ISSN 1022–0038.

[SCIR12] SETTAWATCHARAWANIT, Tossaphol; CHOOCHAISRI, Supasate; INTANAGONWIWAT, Chalermek; ROJVIBOONCHAI, Kultida: V-DESYNC: Desynchronization for Beacon Broadcasting in Vehicular Networks. In: *Proceedings of the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*, 2012 (VTC75 '12).

[TM07] TORRENT MORENO, M.: *Inter-vehicle communications: achieving safety in a distributed wireless environment: Challenges, systems and protocols.* Univ.-Verl. Karlsruhe, 2007

[YN00] YIP, Kun-Wah; NG, Tung-Sang: A simulation model for Nakagami-m fading channels, m < 1. In: *Communications, IEEE Transactions on* 48 (2000), Feb, Nr. 2, S. 214 – 221. `http://dx.doi.org/10.1109/26.823554`. DOI 10.1109/26.823554. ISSN 0090–6778.

Anhang A

Anhang

A.1 Pseudocode des DESYNC-Algorithmus

```
init() {
  T = 1 second
  alpha = 0.95
  just_fired = False
  next_fire, prev_fire = 0
  last_fire = 0
  call SetFiringTimer(T)
}

on_firing_timer_expired() {
  call SendFiringMessage()
  just_fired = True
  my_fire = Now
  prev_fire = last_fire
  call SetFiringTimer(T)
}
```

```
on_receive_firing_message(msg_time) {
  last_fire = msg_time
  if (just_fired == True) {
    just_fired = False
    next_fire = msg_time
    slot_start = T + (prev_fire + my_fire) / 2
    slot_end = T + (next_fire + my_fire) / 2
    goal_time = T + (1 - alpha) * my_fire
      + alpha * (prev_fire + next_fire) / 2
    call SetFiringTimer(goal_time - Now)
    call SetSlotStartTimer(slot_start - Now)
    call SetSlotEndTimer(slot_end - Now)
  }
}
```

Abbildung A.1: Pseudocode für DESYNC auf der MAC-Schicht

A.2 Grafiken lesen, Tutorial

Hier wird beschrieben, wie eine Boxplot-Grafik zu lesen ist. Für verschiedene Rundenlängen wird eine Grafik angegeben. Ein Boxplot besteht aus 5 Werten, minimaler und maximaler Wert, dem 0.25 und 0.75 Perzentil und dem Median. Dabei beschreibt die Box selber die mittleren 50 Prozent der Auswertung, vom 0.25 bis zum 0.75 Quantil, die

schwarze Markierung zeigt den Median (das 0.5 Quantil) und der minimale (0.0 Quantil) und maximale Wert (1.0 Quantil) sind als rote Markierungen angegeben.

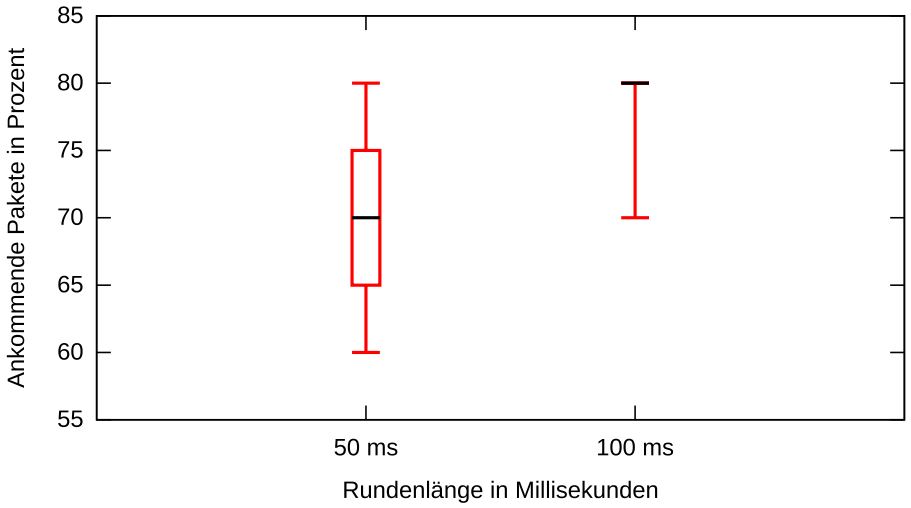


Abbildung A.2: Wie man eine Boxplot-Grafik liest

An diesem Beispiel liegt bei einer Rundenlänge von 50 Millisekunden der minimale Wert bei 60 Prozent, der maximale Wert bei 80 Prozent, der Median bei 70 Prozent und die mittlere Hälfte der geordneten Werte zwischen 65 und 75 Prozent. Beim Beispiel der Rundenlänge von 100 Millisekunden liegt der minimale Wert bei 70 Prozent, und die oberen 75 Prozent der Messwerte bei 80 Prozent.

A.3 DESYNC-Parameter alpha

Um den Zeitpunkt des Feuerns in der nächsten Runde zu berechnen, wird das letzte eigene Feuern und die Mitte der Nachbarfeuerzeitpunkte verwendet. Der Parameter alpha gewichtet diese beiden Werte, sodass der Algorithmus stabiler läuft.

Der Test aus 5.4.1 für 300 Meter Empfangsreichweite und 50 ms Rundenlänge wurde hier für verschiedene Werte für alpha wiederholt.

Es zeigt sich, dass verschiedene Werte für alpha keine große Auswirkung auf das Beaconing nehmen.

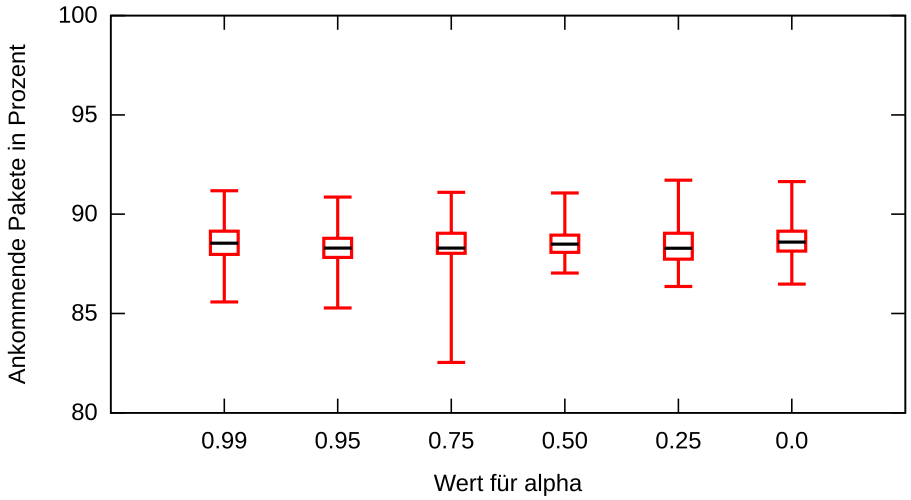


Abbildung A.3: Auswertung des Beaconing für verschiedene alpha-Gewichte

A.4 Zusätzliche Auswertungsgrafiken

A.4.1 Desynchronisierung beim Hinzufügen oder Entfernen eines Fahrzeugs

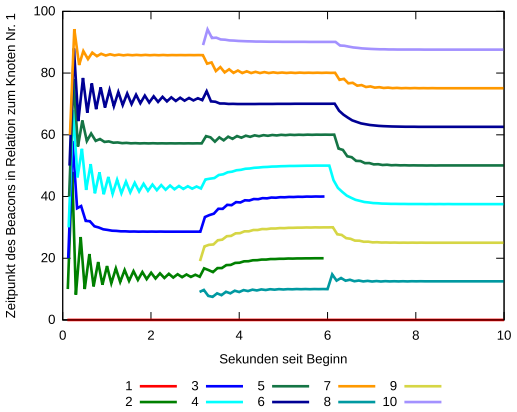
Hier finden sich zusätzliche Grafiken für die Auswertungen von 5.3.1.

Bei dem Durchlauf mit einer Rundenlänge von 100 ms, siehe Abbildung A.4, sind die drei zusätzlichen Fahrzeuge in der dritten Sekunde nach Simulationsbeginn dazugekommen, in der sechsten Sekunde haben zwei Fahrzeuge den Kanal verlassen.

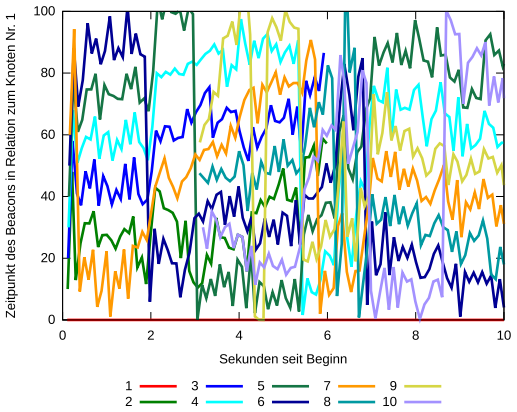
Im Durchlauf mit einer Rundenlänge von 50 ms, siehe Abbildung A.5, sind drei neue Fahrzeuge nach 2 Sekunden Simulationslaufzeit dazugekommen, nach weiteren zwei Sekunden haben zwei Fahrzeuge die Simulation verlassen.

A.4.2 Auswertung des Beaconing in einer Kolonne

Hier finden sich zusätzliche Grafiken für die Auswertungen von 5.4.1.

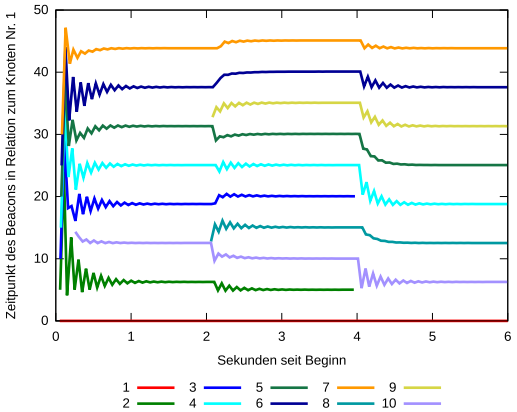


(a) 100 ms Rundenlänge, in verlustfreiem Kanal

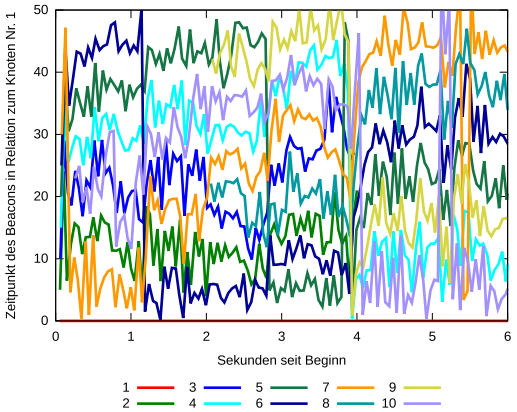


(b) 100 ms Rundenlänge, in verlustbehaftetem Kanal

Abbildung A.4: Desynchronisierung beim Hinzufügen oder Entfernen eines Fahrzeugs, 100 ms Rundenlänge



(a) 50 ms Rundenlänge, in verlustfreiem Kanal

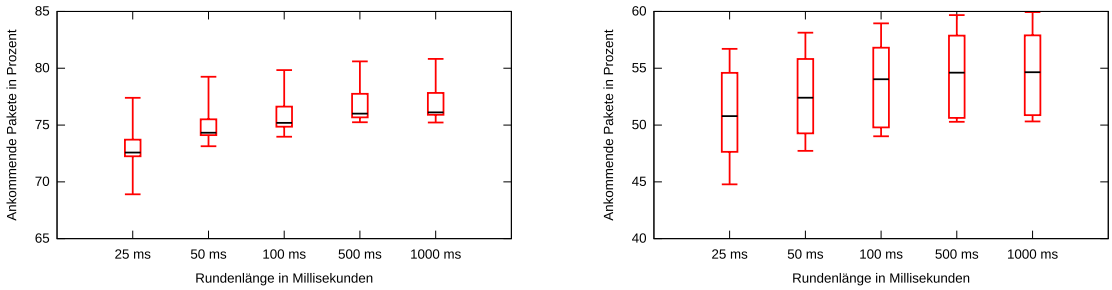


(b) 50 ms Rundenlänge, in verlustbehaftetem Kanal

Abbildung A.5: Desynchronisierung beim Hinzufügen oder Entfernen eines Fahrzeugs, 50 ms Rundenlänge

Beaconing vom Sender aus Die Simulation wurde auch mit Empfangsreichweiten von 500 Metern, siehe Abbildung A.6a, und 1000 Metern, siehe Abbildung A.6b, durchgeführt.

Beaconing beim Empfänger ankommend Ähnlich wird gemessen, wie viel Prozent der verschickten Pakete, die einen speziellen Empfänger erreichen sollen, ankommen.

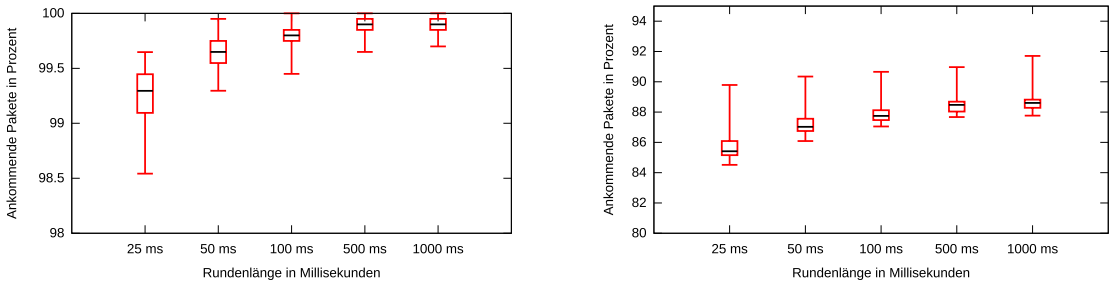


(a) Beaconing vom Sender, 500 Meter

(b) Beaconing vom Sender, 1000 Meter

Abbildung A.6: Auswertung des Beaconing bei Empfangsreichweite von 500 und 1000 Metern

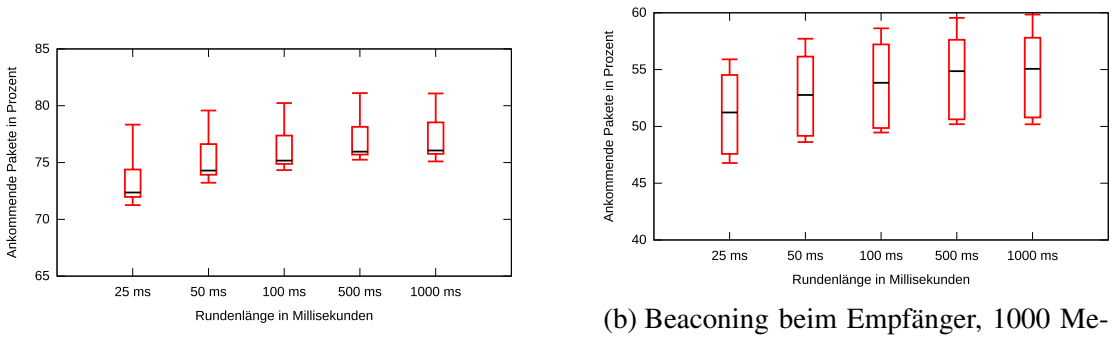
Der Unterschied liegt darin, dass das Verbreiten von Informationen eines Fahrzeugs eine andere Situation beschreibt, als das Benötigen von Informationen aus der Umgebung.



(a) Beaconing beim Empfänger, 100 Meter

(b) Beaconing beim Empfänger, 300 Meter

Abbildung A.7: Auswertung des Beaconing bei Empfangsreichweite von 100 und 300 Metern



(a) Beaconing beim Empfänger, 500 Meter

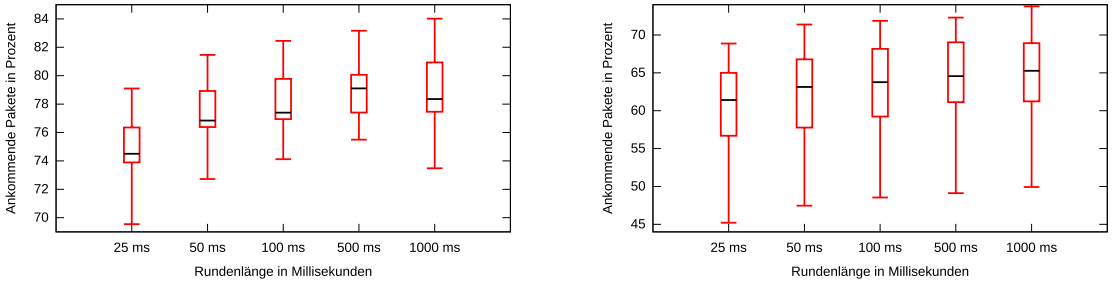
(b) Beaconing beim Empfänger, 1000 Meter

Abbildung A.8: Auswertung des Beaconing bei Empfangsreichweite von 500 und 1000 Metern

A.4.3 Auswertung des Beaconing vom Sender und beim Empfänger beim Zusammentreffen zweier Kolonnen

Hier finden sich zusätzliche Grafiken für die Auswertungen von 5.4.2.

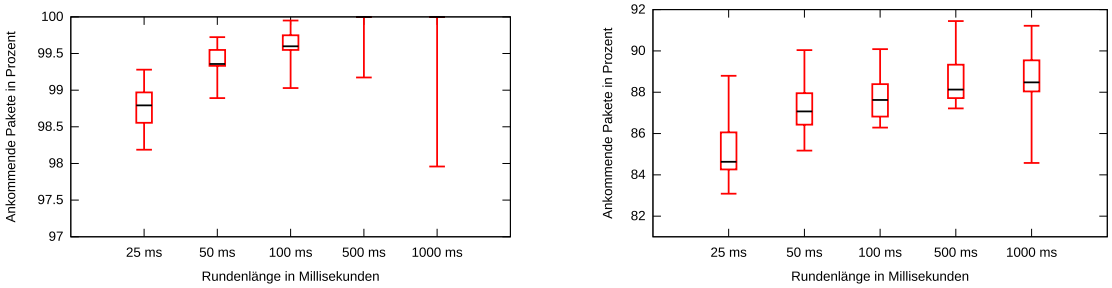
Beaconing vom Sender aus Die Simulation wurde auch mit Empfangsreichweiten von 500 Metern, siehe Abbildung A.9a, und 1000 Metern, siehe Abbildung A.9b, durchgeführt.



(a) Beaconing vom Sender, 500 Meter (b) Beaconing vom Sender, 1000 Meter

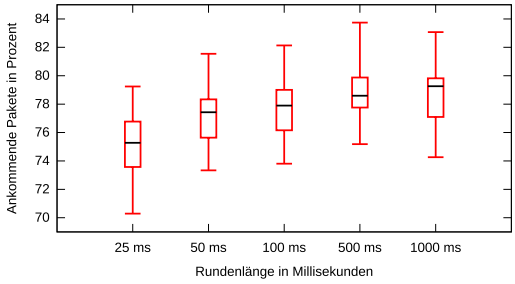
Abbildung A.9: Auswertung des Beaconing bei Empfangsreichweite von 500 und 1000 Metern

Beaconing beim Empfänger ankommend Für das Beaconing beim Empfänger wurden Empfangsreichweiten von 100, 300, 500 und 1000 Metern gewählt.

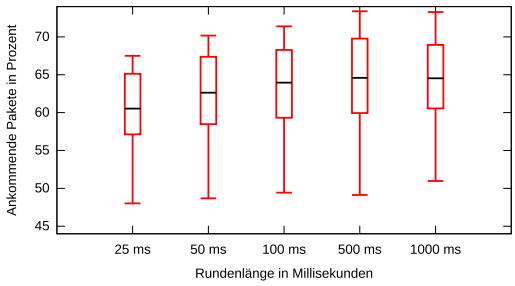


(a) Beaconing beim Empfänger, 100 Meter (b) Beaconing beim Empfänger, 300 Meter

Abbildung A.10: Auswertung des Beaconing bei Empfangsreichweite von 100 und 300 Metern



(a) Beaconing beim Empfänger, 500 Meter



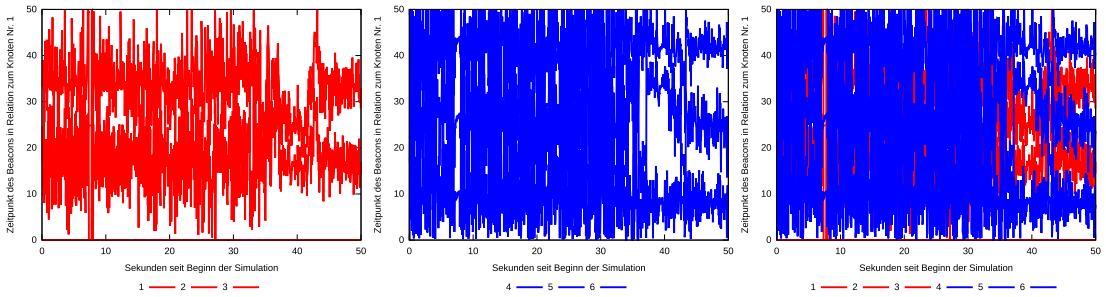
(b) Beaconing beim Empfänger, 1000 Meter

Abbildung A.11: Auswertung des Beaconing bei Empfangsreichweite von 500 und 1000 Metern

A.4.4 Desynchronisierung beim Zusammentreffen von zwei Fahrzeugkolonnen

Hier finden sich zusätzliche Grafiken für die Auswertungen von 5.3.2.

Die Simulation wurde auch mit einer Rundenlänge von 50 ms durchgeführt, siehe Abbildung A.12.



(a) Betrachtung der ersten Kolonne

(b) Betrachtung der zweiten Kolonne

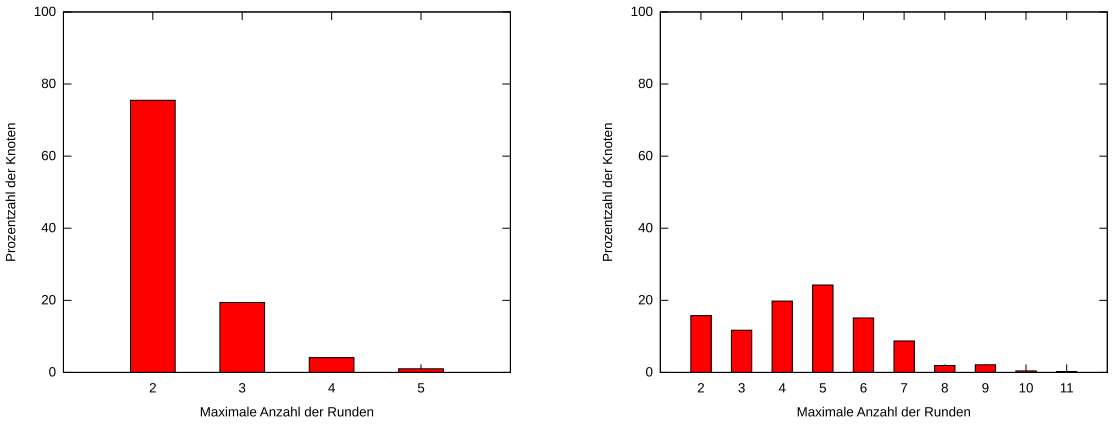
(c) Betrachtung beider Kolonnen

Abbildung A.12: Desynchronisierung zweier Kolonnen bei einer Rundenlänge von 50 ms

A.4.5 Maximale Runden, bis Paketverlust ausgeglichen ist

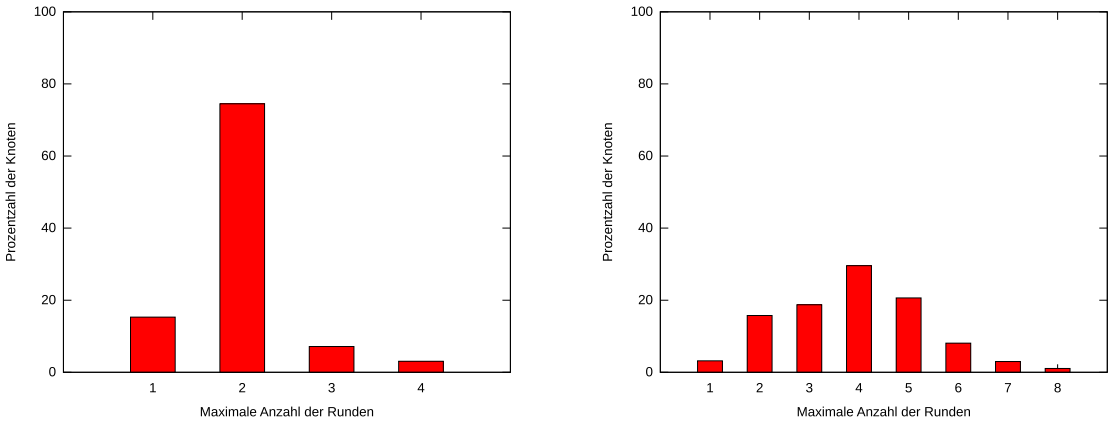
Hier finden sich zusätzliche Grafiken für die Auswertungen von 5.5.2.

Die Simulation wurde auch mit Rundenlängen von 25 ms, siehe Abbildung A.13, und 100 ms, siehe Abbildung A.14, durchgeführt.



(a) 100 Meter Empfangsreichweite (b) 300 Meter Empfangsreichweite

Abbildung A.13: Auswertung der maximalen Runden, bis der Empfang eines Paketes eintritt, bei 25 ms Rundenlänge



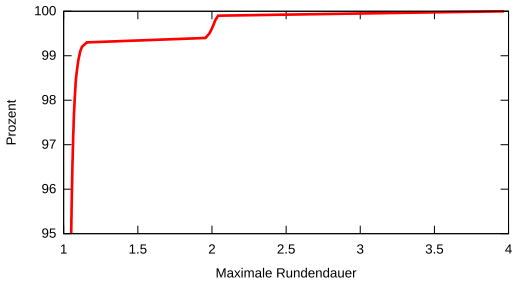
(a) 100 Meter Empfangsreichweite (b) 300 Meter Empfangsreichweite

Abbildung A.14: Auswertung der maximalen Runden, bis der Empfang eines Paketes eintritt, bei 100 ms Rundenlänge

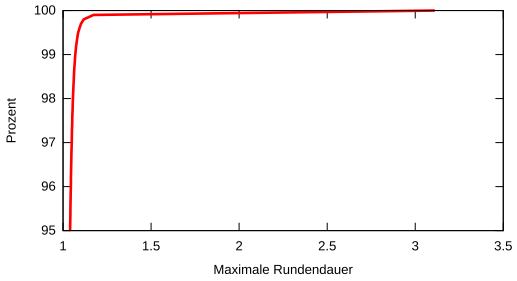
A.4.6 Dichtefunktion

Hier finden sich zusätzliche Grafiken für die Auswertungen von 5.5.1.

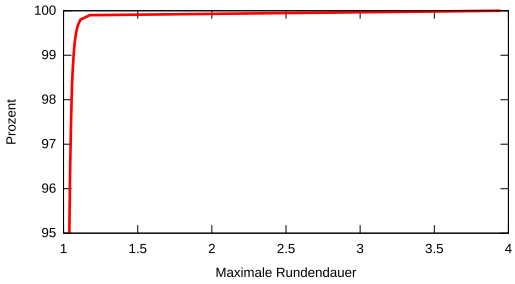
Die Simulation wurde auch mit Rundenlängen von 25 ms, siehe Abbildung A.15a, 500 ms, siehe Abbildung A.15b, und 1000 ms, siehe Abbildung A.15c, durchgeführt.



(a) Kumulative Dichtefunktion bei einer Rundenlänge von 25 ms



(b) Kumulative Dichtefunktion bei einer Rundenlänge von 500 ms



(c) Kumulative Dichtefunktion bei einer Rundenlänge von 1000 ms

Abbildung A.15: Kumulative Dichtefunktion der Rundenanzahl bis ein Beacon ankommt

Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Düsseldorf, 03. Juli 2012 Andre Ippisch