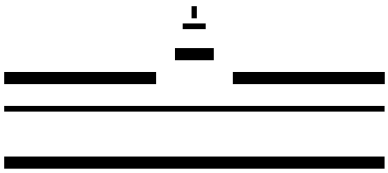

Westfälische
Wilhelms-Universität
Münster

Vergleich der Parametric Reachability Analysis und der Region-based Analysis zur Untersuchung von Hybrid Petri nets with a general one-shot Transition (HPnG) anhand eines Fallbeispiels

Bachelorarbeit
zur Erlangung des akademischen Grades
Bachelor of Science

Westfälische Wilhelms-Universität Münster
Fachbereich Mathematik und Informatik
Institut für Informatik

Betreuung:
Prof. Dr. Anne Remke
Jannik Hüls

Eingereicht von:
Thore Thießen

Münster, September 2016

Zusammenfassung

Das Ziel der vorliegenden Bachelorarbeit war ein Vergleich von zwei Verfahren zur Analyse von *Hybrid Petri nets with a single General one-shot transition* (HPnGs). HPnGs sind ein Formalismus zur Modellierung von hybriden Systemen mit einer Zufallskomponente. Die beiden Verfahren, *Parametric Reachability Analysis* und *Region-based Analysis*, ermöglichen das Bestimmen der Wahrscheinlichkeit, mit der bestimmte Zustände im Modell zu bestimmten Zeitpunkten erreicht werden. Verglichen wurden sowohl die Einschränkungen als auch die Laufzeit der Verfahren in Anwendung auf ein Fallbeispiel. Als ein Ergebnis der Arbeit konnte festgestellt werden, dass keines der beiden Verfahren für alle Modelle anwendbar oder bezüglich der Laufzeit optimal ist. Es konnten weiterhin Kriterien identifiziert werden, anhand derer für verschiedene Anwendungsfälle ein geeignetes Verfahren ausgewählt werden kann.

Eidesstattliche Erklärung

Hiermit versichere ich, Thore Thießen, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Gedanklich, inhaltlich oder wörtlich Übernommenes habe ich durch Angabe von Herkunft und Text oder Anmerkung belegt bzw. kenntlich gemacht. Dies gilt in gleicher Weise für Bilder, Tabellen, Zeichnungen und Skizzen, die nicht von mir selbst erstellt wurden.

Münster, 30. September 2016

Thore Thießen

Inhaltsverzeichnis

1 Einleitung

Hybride Petri-Netze sind eine Variante von Petri-Netzen (PN), mit denen sich hybride Systeme modellieren lassen. Hybride Systeme sind dadurch gekennzeichnet, dass ihr Zustand durch diskrete und kontinuierliche Zustandsvariablen charakterisiert ist [4]. Man kann zwischen autonomen (*autonomous*) und zeitgesteuerten (*timed*) hybriden PNs unterscheiden. Mit autonomen hybriden PNs kann eine qualitative Analyse des möglichen Verhaltens eines Systems durchgeführt werden, bei zeitgesteuerten hybriden PNs sind Stellen oder Transitionen Schaltzeiten zugeordnet, wodurch eine quantitative Analyse eines Systems möglich ist. [5]

Ein Beispiel für hybride Systeme, die mit hybriden PNs modelliert werden können, sind kritische Infrastrukturen. Als kritische Infrastrukturen werden diejenigen Infrastrukturen bezeichnet, die „für die Funktionsfähigkeit moderner Gesellschaften von wichtiger Bedeutung sind“ und deren „Ausfall oder (...) Beeinträchtigung nachhaltige Störungen im Gesamtsystem zur Folge hat“. Zu den sog. kritischen technischen Basisinfrastrukturen zählen beispielsweise Energieversorgung, Verkehr und Trinkwasserversorgung. [6]

Viele hybride Systeme, wie auch die kritischen Infrastrukturen, bestehen aus einem kontinuierlichem System, das durch ein System mit diskreten Zuständen gesteuert wird. Dabei sind das diskrete und kontinuierliche System in sich geschlossen, es findet also kein Austausch zwischen den beiden Systemen statt. Hybride PNs, die solche Systeme modellieren, werden als elementare hybride PNs bezeichnet [5].

HPnGs (*Hybrid Petri nets with a single General one-shot transition*) sind elementare hybride PNs mit zeitgesteuerten Transitionen, bei denen die Schaltzeit genau einer Transition durch eine Zufallsverteilung beschrieben ist. Durch diese Zufallskomponente sind HPnGs geeignet, Fehler und Störungen in hybriden Systemen – wie bspw. kritischer Infrastruktur – zu modellieren, bei denen zufällige Warte- bzw. Reparaturzeiten auftreten können. [9]

Um Aussagen über ein System aus der Modellierung mit einem HPnG ableiten zu können, kann die Wahrscheinlichkeit für das Auftreten bestimmter Zustände im Modell berechnet werden. STL (*Stochastic Time Logic*) ist ein Formalismus, mit dem die Bedingungen für diese Zustände formuliert werden können. In dieser Arbeit werden zwei Verfahren zur Berechnung dieser Wahrscheinlichkeiten betrachtet: Die *Parametric Reachability Analysis* [9] und die *Region-based Analysis* auf Basis des *Stochastic Time Diagram* [7].

1.1 Fragestellung

Der Gegenstand der vorliegenden Arbeit soll ein Vergleich der beiden oben genannten Verfahren zur Analyse von HPnGs – *Parametric Reachability Analysis* und *Region-based Analysis* – sein. Dazu werden beide Verfahren vorgestellt und in einem ersten Schritt die Eigenschaften beider Verfahren verglichen, die die Anwendbarkeit auf bestimmte Modelle beschränken können.

Weiterhin wird im zweiten Schritt ein in der Anzahl der Stellen und Transitionen skalierbares HPnG erstellen, welches ein realistisches Szenario in einer kritischen Infrastruktur modelliert. Für dieses Modell soll dann mit beiden Verfahren die Wahrscheinlichkeit für das Auftreten kritischer Zustände bestimmt werden. Ziel ist ein Vergleich der Laufzeit und der Genauigkeit beider Verfahren in der Anwendung auf ein realistisches Fallbeispiel.

Analysen kritischer Infrastrukturen ermöglichen es, „alle vorhandenen und zu erwartenden Risiken im Vorfeld“ zu erkennen. So können schwerwiegende Störungen vermieden werden. [6] Für diese Analysen sind genaue und schnelle Verfahren notwendig. Der Vergleich der Verfahren erfolgt im Hinblick auf diese Anwendung der Verfahren.

In dieser Arbeit sollen folgende Teilfragen beantwortet werden:

(a) Auf welche Modelle können die beiden Verfahren angewandt werden?

(b) Welche Eigenschaften der Modelle können von den Verfahren untersucht werden?

(c) Für welche Untersuchungen bei welchen Modellen ist welches Verfahren besser geeignet?

HPnGs, STL sowie die beiden Verfahren wurden bereits an anderer Stelle vorgestellt und entwickelt [7] [8] [9], auf die Beschreibung wird deswegen in dieser Arbeit zurückgegriffen. Auch ein Vergleich der Laufzeit beider Verfahren wurde bereits von Ghasemieh et al. [7] vorgenommen. Dieser Vergleich soll hier ausführlicher und anhand eines neuen Fallbeispiels wiederholt werden, wobei die Ergebnisse der vorherigen Arbeit überprüft werden. Erweitert wird der Vergleich der Verfahren dabei zudem um eine Gegenüberstellung der Beschränkungen beider Verfahren.

1.2 Aufbau der Arbeit

In Kapitel 2 dieser Arbeit werden HPnGs vorgestellt und definiert. Dazu werden deren Bestandteile, Zustände und Zustandsevolution erläutert. Außerdem wird die *Stochastic Time Logic* vorgestellt. Im folgenden Kapitel 3 werden dann die beiden Verfahren *Parametric Reachability Analysis* und *Region-based Analysis* besprochen. Dazu wird jeweils der zugrunde liegende Algorithmus und das Verfahren zur Bestimmung der Wahrscheinlichkeit einer STL-Formel nachvollzogen. Danach werden die möglichen Einschränkungen der Verfahren diskutiert.

In Kapitel 4 wird das Fallbeispiel dargelegt und in Kapitel 5 werden die beiden Verfahren dann sowohl bezüglich ihrer Einschränkungen als auch am Beispiel bezüglich ihrer Laufzeit verglichen. Den Abschluss bildet das Fazit in Kapitel 6, in dem die Ergebnisse der Arbeit zusammengefasst werden.

2 HPnGs

In diesem Kapitel werden HPnGs als eine Form hybrider PNs vorgestellt. Eine genauere Darstellung findet sich in der Arbeit von Gribaudo und Remke [9], welche auch hier als Grundlage dient. Zunächst werden in Kapitel 2.1 die Bestandteile des Modells mit ihrer semantischen Bedeutung besprochen und das Modell definiert. Im folgenden Kapitel 2.2 werden dann Zustände im Modell definiert und die Zustandsevolution erläutert, d.h. die Entwicklung des Modells von einem Zustand zu einem anderen.

Den Abschluss bildet eine Vorstellung der STL in Kapitel 2.3. Diese gehört nicht zur Definition der HPnGs, bezieht sich aber auf dessen Zustände und wird deswegen direkt im Anschluss erläutert.

2.1 Bestandteile des Modells

HPnGs sind als PNs gerichtete Graphen, in denen sich grundsätzlich zwei Arten von Knoten unterscheiden lassen: Stellen – welchen als Markierung ein bestimmter Wert zugeordnet wird – und Transitionen. Da es sich um ein hybrides PN handelt, gibt es diskrete und kontinuierliche Stellen, deren Markierung jeweils ein diskreter (in $\mathbb{N}$) oder kontinuierlicher Wert (in $\mathbb{R}_{\geq 0}$) entspricht.

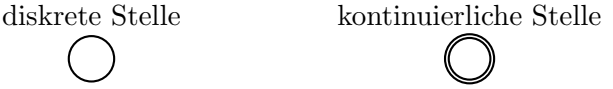

Abbildung 2.1: ***Grafische HPnG-Notation von Stellen***

Definition 1 (Tokens)**.** Tokens sind eine alternative Beschreibung für die Markierung einer diskreten Stelle. Die Anzahl von Tokens zu einer diskreten Stelle entspricht dabei der Markierung dieser Stelle. Das Hinzufügen bzw. Entfernen von n Tokens entspricht der Erhöhung bzw. Verringerung der Markierung um den Wert n. [9]

Transitionen sind Knoten, die Stellen miteinander verbinden und deren Struktur und Parameter maßgeblich das Verhalten des Modells beeinflussen. Auch hier lassen sich diskrete und kontinuierliche Transitionen unterschieden. Es gibt drei Arten von diskreten Transitionen: deterministische, sofortige und generelle Transitionen. Deterministische Transitionen können

eine Veränderung der diskreten Markierungen nach einer festen Schaltzeit bewirken. Sofortige Transitionen sind deterministische Transitionen mit einer Schaltzeit von 0. Bei generellen Transitionen ist die Schaltzeit zufällig verteilt. Die generelle Transition ist namensgebend für HPnGs, in jedem Modell gibt es gemäß der Definition genau eine generelle Transition, die nur ein einziges Mal aktiviert werden kann.

Zusätzlich gibt es nur eine einzige Art kontinuierlicher Transitionen, die eine Veränderung der kontinuierlichen Markierungen bewirken können. Im Unterschied zu den diskreten Transitionen finden diese Veränderungen nicht nach einer Schaltzeit zu einem bestimmten Zeitpunkt statt, sondern ständig mit einer bestimmten Rate. Da sowohl diskrete Transitionen die kontinuierlichen Markierungen als auch kontinuierliche Transitionen die diskreten Markierungen nicht verändern können, sind HPnGs dementsprechend elementare hybride PNs. Allerdings ist eine gegenseitige Beeinflussung der diskreten und kontinuierlichen Markierungen (durch Test- und Inhibitor-Bögen, s.u.) möglich.

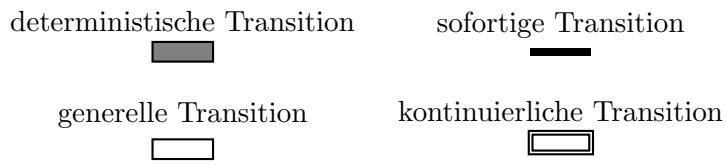

Abbildung 2.2: ***Grafische HPnG-Notation von Transitionen***

Die Verbindung zwischen Stellen und Transitionen erfolgt über Bögen, welche den Kanten des Graphen entsprechen. Grundsätzlich verbindet ein Bogen immer genau eine Stelle mit einer Transition, außerdem kann zwischen einem solchen Paar (von Stelle und Transition) nur ein Bogen existieren. Es gibt vier Arten von Bögen: diskrete Bögen, kontinuierliche Bögen, Test-Bögen und Inhibitor-Bögen, wobei jedem Bogen ein Gewicht zugeordnet wird. Diskrete Bögen verbinden eine diskrete Stelle mit einer diskreten Transition, entweder als Eingabe-Bogen in Stelle-Transition-Richtung oder als Ausgabe-Bogen in Transition-Stelle-Richtung. Das Gewicht des Bogens bestimmt die Anzahl von Tokens, die einer Stelle beim Schalten der Transition entnommen oder hinzugefügt werden.

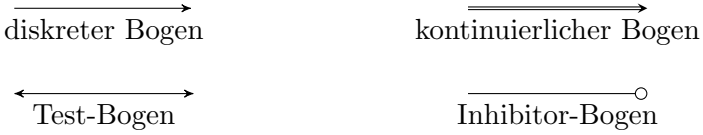

Abbildung 2.3: ***Grafische HPnG-Notation von Bögen***

Kontinuierliche Bögen verbinden kontinuierliche Stellen mit kontinuierlichen Transitionen, analog zu diskreten Bögen gibt es auch hier Ein- und Ausgabe-Bögen. Das Gewicht ist bei kontinuierlichen Bögen ein Faktor zu der Flussrate der Transition (vgl. 2.2.2), das Produkt von Gewicht und Flussrate bestimmt den Fluss durch die Transition von bzw. zu den verbundenen Stellen.

Test- und Inhibitor-Bögen können diskrete oder kontinuierliche Stellen mit diskreten oder kontinuierlichen Transitionen verbinden, allerdings nur in dieser Richtung. Sie verändern nicht die Markierung der Stellen, sondern deaktivieren die Transitionen, wenn die Stelle eine geringere (bei Test-Bögen) oder nicht-geringere (bei Inhibitor-Bögen) Markierung als das Gewicht des Bogens besitzt.

Definition 2 (HPnG)**.** Ein HPnG ist ein Tupel $\langle P, T, A, m_0, x_0, \Phi \rangle$ [9]:

- $P = P^d \cup P^c$ ist die Menge der Stellen, die aus den diskreten (P^d) und kontinuierlichen Stellen (P^c) besteht.
- $T = T^D \cup T^I \cup T^G \cup T^F$ ist die Menge aller Transitionen mit den deterministischen (T^D), sofortigen (T^I), generellen (T^G) und kontinuierlichen Transitionen (T^F).
- $A = A^d \cup A^f \cup A^t \cup A^h$ ist die Menge der Bögen bestehend aus diskreten (A^d), kontinuierlichen (A^f), Test- (A^t) und Inhibitor-Bögen (A^h).
- $m_0 \in \mathbb{N}^{|P^d|}$ sind die Markierungen der diskreten und $x_0 \in \mathbb{R}_{\geq 0}{}^{|P^c|}$ die Markierungen der kontinuierlichen Stellen im Ausgangszustands des Modells.

Φ ist wiederum ein Tupel aus neun Funktionen, das den Stellen, Transitionen und Bögen verschiedene Parameter zuweist:

- Jeder kontinuierlichen Stelle P^c kann eine positive reelle obere Schranke für die Markierung zugewiesen werden.
- Allen Bögen A wird ein positives Gewicht zugeordnet. Für Bögen, die mit einer diskreten Stelle verbunden sind, muss dieses Gewicht ganzzahlig sein.
- Deterministischen und sofortigen Transitionen $T^D \cup T^I$ wird eine nicht-negative ganzzahlige Priorität und ein positives Gewicht[a] zugewiesen, welche für die Konfliktauflösung verwendet werden (vgl. 2.2.1).
- Analog wird kontinuierlichen Bögen A^f eine nicht-negative Priorität und ein positiver Anteil zugewiesen. Diese Parameter sind relevant für die Flussratenanpassung (vgl. 2.2.2).
- Deterministischen Transitionen T^D wird eine nicht-negative Schaltzeit, kontinuierlichen Transitionen $x \in T^F$ eine positive nominale Flussrate $\phi_f^T(x)$ und der einzigen generellen Transition $T_1^G \in T^G$ eine Zufallsverteilung für die Schaltzeit zugewiesen.

[a] Dieses Gewicht ist zu unterscheiden von dem oben beschriebenen Gewicht der Bögen.

2.2 Zustand und Zustandsevolution

Der Zustand eines HPnG ist zeitabhängig und Zustandsveränderungen können dadurch auftreten, dass diskrete Transitionen feuern (vgl. 2.2.1), wodurch diskrete Markierungen sofort verändert werden, oder durch eine positive Flussrate in einer kontinuierlichen Transition (vgl. 2.2.2), wodurch sich kontinuierliche Markierungen mit der Zeit ändern.

Definition 3 (Zustand)**.** Ein Zustand in einem HPnG ist als ein Tupel $\langle m, x, c, g\rangle$ definiert [9]. Dabei sind $m \in \mathbb{N}^{|P^d|}$ und $x \in \mathbb{R}_{\geq 0}{}^{|P^c|}$ die Markierungen der diskreten und kontinuierlichen Stellen.

Für jede deterministische Transition enthält $c \in \mathbb{R}_{\geq 0}{}^{|P^c|}$ die Länge des Zeitraums, in dem die Transition seit dem letzten Feuern aktiviert war (Aktivierung vgl. 2.2.1). Analog enthält $g \in \mathbb{R}_{\geq 0} \cup \{-1\}$ die Länge des Aktivierungszeitraums für die generelle Transition, wird aber auf einen ungültigen Wert (-1) gesetzt, wenn die generelle Transition gefeuert hat.

2.2.1 Diskrete Transitionen

Diskrete Transitionen können nur feuern, wenn sie aktiviert sind, und eine notwendige Bedingung für die Aktivierung ist eine Konzession.

Definition 4 (Konzession für diskrete Transitionen)**.** Eine diskrete Transition hat Konzession, wenn die drei folgenden Bedingungen erfüllt sind [9]:

(a) In allen Stellen, in denen beim Feuern der Transition durch Eingabe-Bögen Tokens entfernt werden, sind genug Tokens vorhanden.

(b) Die Markierung in allen Stellen, die mit der Transition durch einen Test-Bogen verbunden sind, ist mindestens so groß ist wie das Gewicht des Bogens.

(c) Die Markierung in allen Stellen, die mit der Transition durch einen Inhibitor-Bogen verbunden sind, ist kleiner als das Gewicht des Bogens.

Definition 5 (Aktivierung diskreter Transitionen)**.** Alle sofortigen Transitionen, die Konzession haben, sind damit auch aktiviert. Deterministische und generelle Transitionen sind aktiviert, wenn sie Konzession haben und keine sofortige Transition Konzession hat. [9]

Solange deterministische oder generelle Transitionen aktiviert sind, erhöht sich deren Zähler (in c des Zustands-Tupels) für die Länge ihres Aktivierungszeitraums und die Transitionen feuern genau dann, wenn der Zähler genau ihrer Schaltzeit entspricht. Die generelle Transition

feuert damit entsprechend der Zufallsverteilung für ihre Schaltzeit mit einer bestimmten Wahrscheinlichkeit in einem Zeitraum, in dem sie aktiviert ist. Der Zähler wird für deterministische bzw. generelle Transitionen nur nach dem Feuern zurück auf 0 bzw. -1 gesetzt, er bleibt somit auch bei einer Deaktivierung ohne Feuern erhalten (*preemptive resume memory policy*).

Aufgrund der Definition der Aktivierung haben sofortige Transitionen Vorrang vor deterministischen Transitionen und feuern damit zuerst. Außerdem haben deterministische Transitionen Vorrang vor der generellen Transition. Dennoch können Konflikte auftreten, wenn zwei sofortige Transitionen aktiviert sind oder der Zähler von zwei deterministischen Transitionen gleichzeitig die jeweilige Schaltzeit erreicht. In diesen Fällen wird eine Konfliktauflösung auf Basis der Prioritäten und Gewichte der Transitionen durchgeführt: Dabei haben jeweils die Transitionen mit der höchsten Priorität Vorrang. Von mehreren Transitionen mit der höchsten Priorität wird eine zufällig ausgewählt, wobei die Vorrangswahrscheinlichkeit für eine Transition dem Gewicht dieser Transition geteilt durch die Summe der Gewichte (aller Transitionen mit der höchsten Priorität) entspricht. Durch dieses zweistufige Konzept sind sowohl Vorrangsordnungen als auch indeterministische Modelle möglich.

2.2.2 Kontinuierliche Transitionen

Eine kontinuierliche Transition hat genau dann Konzession, wenn sie die Bedingungen (b) und (c) (vgl. 2.2.1) erfüllt; kontinuierliche Transitionen mit Konzession sind gleichzeitig auch aktiviert.

Definition 6 (Flussrate)**.** Einer aktivierten kontinuierlichen Transition x wird eine Flussrate im Bereich $[0, \phi_f^T(x)]$ zugewiesen. Wenn eine kontinuierliche Transition nicht aktiviert ist, hat sie eine Flussrate von 0. Wenn alle Stellen, die durch einen Eingabe-Bogen mit der Transition verbunden sind, eine Markierung größer als 0 besitzen und alle Stellen, die durch einen Ausgabe-Bogen mit der Transition verbunden sind, eine Markierung kleiner als die obere Schranke besitzen, so ist die Flussrate gleich der nominalen Flussrate $\phi_f^T(x)$. [9]

Definition 7 (Drift)**.** Drift ist die Veränderungsrate der Markierung einer kontinuierlichen Stelle, wenn alle Ein- und Ausflüsse miteinander verrechnet werden [9]. Ein Drift von 0 bedeutet damit, dass sich die Markierung der Stelle nicht verändert, nicht jedoch dass die Stelle keine Ein- oder Ausflüsse hat.

Wenn eine Ausgabe-Stelle eine Markierung an der oberen Schranke besitzt, so muss eine Flussratenanpassung für alle Transitionen vorgenommen werden, die durch Ausgabe-Bögen mit dieser Stelle verbunden sind. Analoges gilt bei einer Eingabe-Stelle mit einer Markierung von 0 für alle Transitionen, die mit dieser durch Eingabe-Bögen verbunden sind. Die Flussratenanpassung erfüllt die folgenden Anforderungen:

- Keine Stelle mit einer Markierung von 0 hat einen negativen Drift und keine Stelle mit einer Markierung an der oberen Schranke hat einen positiven Drift. Dadurch ist garantiert, dass die Markierung einer kontinuierlichen Stelle auf den gültigen Bereich beschränkt bleibt.
- Verfügbare Ein- bzw. Ausfluss-Kapazitäten der Stelle werden iterativ in Reihenfolge der Priorität auf die Bögen aufgeteilt.
- Bei Bögen mit gleicher Priorität werden die Kapazitäten proportional zum Produkt aus dem Gewicht des Bogens und der nominalen Flussraten der Transition aufgeteilt. Die Flussrate einer Transition kann dabei jedoch die nominale Flussrate nicht überschreiten.

Der genaue Algorithmus für die Flussratenanpassung ist in der Arbeit von Gribaudo und Remke [9] beschrieben.

2.2.3 Zusammenfassung

Die Zustandsevolution eines HPnG beginnt zum Zeitpunkt $t = 0$ mit dem Zustand

$$Z_0 = \langle m_0, x_0, \langle 0 \dots 0 \rangle, 0 \rangle \tag{2.1}$$

Durch die kontinuierlichen Transitionen gibt es bei HPnGs im Gegensatz zu klassischen PNs im Allgemeinen eine überabzählbare Anzahl möglicher Zustände eines Modells, auch wenn die Schaltzeit der generellen Transition als fest angenommen wird. Es lassen sich jedoch Ereignisse wie das Feuern einer diskreten Transition oder das Erreichen des Gewichts von Test- bzw. Inhibitor-Bögen in Stellen identifizieren, durch die die Menge von Zuständen des Modells diskretisiert werden kann.

Wichtig ist auch anzumerken, dass die Zustandsevolution eines Modells nicht zwangsläufig eindeutig ist. Einerseits ist die Schaltzeit der generellen Transition prinzipiell zufällig verteilt, aber auch das Feuern anderer diskreter Transitionen ist bei gleicher Priorität indeterministisch. Dies hat zur Folge, dass die Angabe eines Zeitpunkts und der Schaltzeit der generellen Transition im Bezug auf ein HPnG nicht notwendigerweise ausreicht, einen eindeutigen Zustand zu identifizieren.

2.3 Stochastic Time Logic

Stochastic Time Logic (STL) erlaubt die Erstellung von Formeln, mit denen Bedingungen für die Zustände eines HPnG ausgedrückt werden können.

Definition 8 (STL-Formel)**.** Die Struktur einer STL-Formel Ψ ist definiert als [8]

$$\Psi := \text{tt} \mid x_P \geq c \mid m_P = a \mid \neg\Psi \mid \Psi \wedge \Psi \mid \Psi \mathcal{U}^{[T_1,T_2]} \Psi$$

Einer solchen Formel kann ein Wahrheitswert (wahr oder falsch) zugeordnet werden, wenn sie auf einen Zustand bezogen wird. Für einen Zustand z zum Zeitpunkt t eines HPnG mit fester aber beliebiger Schaltzeit der generellen Transition wird eine Formel dabei wie folgt – ggf. rekursiv – ausgewertet:

(a) tt ist immer wahr.

(b) $x_P \geq c$ ist genau dann wahr, wenn die kontinuierliche Stelle P im Zustand z eine Markierung von mindestens c besitzt, d.h. $z.x_P \geq c$.

(c) $m_P = a$ ist genau dann wahr, wenn die diskrete Stelle P im Zustand z eine Markierung von genau a besitzt, d.h. $z.m_P = a$.

(d) $\neg\Psi$ ist genau dann wahr, wenn die Formel Ψ für z falsch ist.

(e) $\Psi_1 \wedge \Psi_2$ ist genau dann wahr, wenn sowohl Ψ_1 als auch Ψ_2 für z wahr sind.

(f) $\Psi_1 \mathcal{U}^{[T_1,T_2]} \Psi_2$ ist für Zustand z genau dann wahr, wenn es einen Zeitpunkt $u \in [t + T_1, t + T_2]$ gibt, so dass (1) Ψ_2 für den Zustand des Modells zum Zeitpunkt u wahr ist und (2) Ψ_1 für jeden Zustand des Modells im Zeitbereich $[t, u]$ wahr ist [8].

Die Formelausdrücke (a) und (b) beziehen sich direkt auf die Markierung einer Stelle. Zusammen mit der Negation (d) und der Konjunktion (e) lassen sich so Formeln bilden, die komplexen Bedingungen im Bezug auf die Markierungen des Modells entsprechen. Beispielsweise entspricht die Formel $\neg(\neg\Psi_1 \wedge \neg\Psi_2)$ der Disjunktion der Formeln Ψ_1, Ψ_2 und die Formel $x_P \geq c_1 \wedge \neg(x_P \geq c_2)$ entspricht der Überprüfung der Markierung der kontinuierlichen Stelle P auf den Bereich $[c_1, c_2)$.

Im Gegensatz dazu bezieht sich der Formelausdruck (f), der *until*-Operator genannt wird, nicht nur auf die Markierungen. Vielmehr wird die Erreichbarkeit bestimmter Markierungen überprüft. Damit ermöglicht der Operator die Formulierung von Aussagen, die nicht nur den aktuellen, sondern auch mögliche zukünftige Zustände mit einbeziehen. [8] Zum *until*-Operator ist jedoch anzumerken, dass er (in der oben genannten Variante) implizit das deterministische Feuern der sofortigen und deterministischen Transitionen voraussetzt Der Grund dafür ist, dass in der Definition davon ausgegangen wird, dass der Zustand eines HPnG durch die Angabe von Zeitpunkt und Schaltzeit der generellen Transition eindeutig bestimmt ist. Der STL-*until*-Operator ist angelehnt an den *until*-Operator der *Continuous Stochastic Logic* (CSL) [2].

Eine STL-Formel kann auch allgemein auf ein HPnG zu einem Zeitpunkt t bezogen werden, auch wenn dann kein eindeutiger Zustand bestimmt ist. Der Formel wird dann aber nicht

mehr ein Wahrheitswert, sondern ein Wahrscheinlichkeitswert zugeordnet. Dieser entspricht der Wahrscheinlichkeit, dass die Formel zum Zeitpunkt t der Ausführung des Modells wahr ist, wenn die Schaltzeit der generellen Transition bei dessen erster Aktivierung gemäß Definition aus der Zufallsverteilung gesampelt wird. Durch diesen Bezug von Formeln auf Zeitpunkte des Modells im Gegensatz zu einzelnen Zuständen ist es überhaupt erst möglich, allgemeine Aussagen über das Modell zu treffen. So kann beispielsweise die Wahrscheinlichkeit für das Auftreten bestimmter Szenarien in hybriden Systemen errechnet werden, wenn sich das Szenario als STL-Formel im Bezug auf ein HPnG formulieren lässt.

3 Analyseverfahren

Parametric Reachability Analysis und *Region-based Analysis* sind zwei Verfahren, mit denen die Wahrscheinlichkeit für die Wahrheit einer STL-Formel zu einem Zeitpunkt in einem HPnG berechnet werden kann. Beide basieren auf unterschiedlichen Datenstrukturen und haben unterschiedliche Einschränkungen.

In diesem Kapitel sollen beide Verfahren nacheinander präsentiert werden. Dafür wird jeweils die generelle Idee des Verfahrens vorgestellt und dann der Algorithmus zur Erstellung der zentralen Datenstruktur nachvollzogen. Anschließend wird die Berechnung der Wahrscheinlichkeit einer STL-Formel anhand der Datenstruktur – das sogenannt *Model Checking* – erläutert. Die *Parametric Reachability Analysis* wird in der Arbeit von Gribaudo und Remke [9] und die *Region-based Analysis* in der Arbeit von Ghasemieh et al. [7] in allen Details vorgestellt, die Darstellung der Verfahren in dieser Arbeit basiert auf den beiden Arbeiten.

Abschließend werden für beide Verfahren die jeweiligen Einschränkungen und damit auch die Anwendbarkeit für die Analyse von hybriden Systemen diskutiert.

3.1 Parametric Reachability Analysis

Der *Parametric Reachability Analysis* liegt zentral eine Baumstruktur zugrunde. In diesem sogenannten *Parametric Location Tree* sind alle möglichen Zustände des untersuchten HPnG (bis zu einer Maximalzeit $t_{\max}$) abgebildet. Dabei entspricht ein Knoten des Baums – *Parametric Location* genannt – einer Menge von Zuständen. Ein Knoten ist dann Nachfolger eines anderen, wenn die Zustände des nachfolgenden Knoten mögliche Nachfolgezustände der Zustände des vorhergehenden Knoten sind. Durch die Baumstruktur können im *Parametric Location Tree* unterschiedliche Entwicklungen des Modells abgebildet werden, sowohl für unterschiedliche Schaltzeiten der generellen Transition als auch für einen möglichen Indeterminismus beim Feuern der diskreten Transitionen.

Definition 9 (Parametric Location)**.** Eine *Parametric Location* ist definiert als ein Tupel $\langle t, m, x, c, g, l, r, p \rangle$ [9] und ist eine Erweiterung des Zustands eines HPnG:

- $t : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$ ist eine lineare Funktion in Abhängigkeit von der Schaltzeit s der generellen Transition und gibt für jeden Wert von s den Zeitpunkt an, zu dem im

Modell der erste Zustand zu der *Parametric Location* auftritt.

- $m \in \mathbb{N}^{|P^d|}$ enthält die Markierungen der diskreten Stellen und $g \in \mathbb{R}_{\geq 0} \cup \{-1\}$ entspricht der Länge des Aktivierungszeitraums der generellen Transition zum Zeitpunkt $t(s)$. Damit ist die Bedeutung dieser Elemente analog zu der der Elementen m und g des Zustandstupels.
- $t : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}{}^{|P^c|}$ ist eine lineare Funktion, die für eine Schaltzeit s der generellen Transition den kontinuierlichen Stellen zum Zeitpunkt $t(s)$ die Markierung $x(s)$ zuweist.
- $c : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}{}^{|T^D|}$ ist ebenfalls eine lineare Funktion, $c(s)$ enthält die Zähler für die Länge der Aktivierungszeiträume der deterministischen Transitionen zum Zeitpunkt $t(s)$.
- Der Bereich $[l, r) \subseteq \mathbb{R}_{\geq 0}$ gibt mögliche Werte für die Schaltzeit s der generellen Transition an, für die diese *Parametric Location* gültig ist.
- $p \in (0, 1]$ gibt die Wahrscheinlichkeit für diese *Parametric Location* an, wenn eine indeterministische Konfliktauflösung zwischen diskreten Transitionen durchgeführt wird.

Der *Parametric Location Tree* kann ausgehend von der Wurzel (analog zum Anfangszustand Z_0 in Formel (2.1))

$$L_0 = \langle 0, m_0, x_0, \langle 0 \dots 0 \rangle, 0, 0, \infty, 1 \rangle$$

aufgebaut werden, wobei eine obere Schranke $t_{\max}$ für die Zeit festgelegt wird, damit der Aufbau immer terminiert. Für jede *Parametric Location* werden die möglichen nächsten Ereignisse (Feuern einer sofortigen oder deterministischen Transition, Markierung einer Stelle erreicht eine obere oder untere Schranke, usw.) bestimmt. Wenn die generelle Transition noch nicht gefeuert hat, sind t, x und c konstant, daher treten alle Ereignisse, die als nächstes Ereignis in Frage kommen, – abgesehen vom Feuern der generellen Transition – für alle Werte von s mit der gleichen minimalen Verzögerung Δt auf. Die folgenden *Parametric Locations* werden aus den möglichen Ereignissen berechnet und als Nachfolger zum aktuellen Knoten in den Baum eingefügt. Zudem wird eine *Parametric Location* für das mögliche Feuern der generellen Transition eingefügt, wenn diese aktiviert ist und die anderen möglichen Ereignisse erst nach einer Verzögerung von $\Delta t > 0$ eintreten. Der Bereich für s wird für den Knoten zum Feuern der generellen Transition auf $[l, g + \Delta t)$ gesetzt, für die anderen eingefügten Knoten ist er $[g + \Delta t, r)$ (wobei l und r den Grenzen von s für die aktuelle *Parametric Location* entsprechen).

Nach dem Feuern der generellen Transition kann diese entsprechend der Definition des HPnG nicht noch einmal feuern, weswegen sie nicht weiter berücksichtigt werden muss. Dafür stehen t, x und c nun möglicherweise in linearer Abhängigkeit von s. Deswegen kann es hier mehrere Sätze möglicher nächster Ereignisse mit jeweils minimalem Verzögerungen

für verschiedene Teilintervalle von $[l, r)$ geben. Es werden für alle möglichen Ereignisse die entsprechenden Folgeknoten berechnet und eingefügt, wobei der Bereich für s jeweils entsprechend auf den Bereich gesetzt wird, in dem die Verzögerung für das Ereignis minimal war. Für einen Knoten werden keine Folgeknoten eingefügt, wenn keine Ereignisse eintreten können oder dadurch die obere Schranke für die Zeit überschritten werden würde. Ein Zustand, in dem keine weiteren Ereignisse eintreten können, wird als absorbierend bezeichnet.

Dadurch, dass das Auftreten eines Ereignisses die Erstellung einer neuen *Parametric Location* bedingt, sind die diskreten Markierungen innerhalb einer Parametric Location konstant, denn jedes Feuern einer diskreten Transition, wodurch die diskreten Markierungen verändert werden können, entspricht einem Ereignis. Zudem ist auch der Drift der kontinuierlichen Stellen in einer *Parametric Location* konstant, denn eine Anpassung der Flussraten der fließenden Transitionen ist nur nach dem Eintreten eines Ereignisses notwendig. In der Arbeit von Gribaudo und Remke [9] ist in der Definition der *Parametric Location* auch ein Element d beschrieben, welches den konstanten Drift für alle kontinuierlichen Stellen enthält. Das Element wurde hier weggelassen, da die Werte sich aus den Werten der anderen Elemente ergeben.

3.1.1 Model Checking

Die Berechnung der Wahrscheinlichkeit für die Wahrheit einer STL-Formel zu einem Zeitpunkt in einem HPnG ist in der Arbeit von Gribaudo und Remke [9] durch eine Diskretisierung der Zufallsverteilung für s umgesetzt. Dafür wird eine Funktion $\xi_L^\Psi(t \mid s)$ definiert, die ausgehend vom Knoten L die Wahrscheinlichkeit dafür berechnet, dass die Formel Ψ zum Zeitpunkt t und zur Schaltzeit s der generellen Transition gültig ist. Dafür werden der Knoten L sowie rekursiv alle Folgeknoten berücksichtigt und gemäß ihrer Wahrscheinlichkeit p gewichtet. So kann die Wahrscheinlichkeit angenähert werden mit

$$\sum_{i=0}^{\infty} \left[\xi_{L_0}^{\Psi}(t \mid i * \Delta s) * pdf(i * \Delta s) * \Delta s\right] \tag{3.1}$$

Dabei sind Δs die Größe der Diskretisierungsschritte für s, $pdf : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$ die Wahrscheinlichkeitsdichte für s und L_0 die Wurzel des *Parametric Location Tree*. Eine mögliche Implementierung dieser Annäherung ist in 5.1 beschrieben.

Durch die Diskretisierung ist dieses Verfahren für die meisten Zufallsverteilungen nicht exakt. Es ist jedoch prinzipiell eine beliebig genaue Annäherung an das tatsächliche Ergebnis durch eine Verkleinerung der Diskretisierungsschritte möglich. Dabei steigt jedoch gleichzeitig auch der Rechenaufwand.

Eine Alternative zu der Diskretisierung wäre eine geometrische Lösung: Bei dieser könnte zu einer Zeit t für jede *Parametric Location*, die zu diesem Zeitpunkt gültig ist, das Intervall

für s bestimmt werden, in dem die Formel gültig ist. Diese Bestimmung ist möglich, da die Markierungen in einer *Parametric Location* entweder konstant oder nur linear von s und t abhängig sind. Die Intervalle würden dann mit der jeweiligen Wahrscheinlichkeit p und über die kumulative Verteilungsfunktion gewichtet und zu der Gesamtwahrscheinlichkeit für die Formel addiert werden. Dieses Verfahren ist derzeit für den *Parametric Location Tree* nicht implementiert.

3.1.2 Einschränkungen

Die *Parametric Reachability Analysis* ist auf jedes HPnG und jede Zufallsverteilung anwendbar. Für die Berechnung der Wahrscheinlichkeit einer STL-Formel muss jedoch zwischen Genauigkeit und Rechenaufwand abgewogen werden. Dieser Umstand könnte durch die Implementierung des oben beschriebenen geometrischen Verfahrens behoben werden, dann muss allerdings eine kumulative Verteilungsfunktion zu der Zufallsverteilung verfügbar sein.

Eine große Einschränkung, der das Verfahren unterliegt, ist, dass der *until*-Operator der STL für indeterministische HPnGs nicht definiert ist. Entsprechend ist der Operator für dieses Verfahren nicht implementiert. Dadurch sind die Analysewerkzeuge, die mit diesem Verfahren für hybride Systeme zur Verfügung stehen, eingeschränkt, weil nur die Wahrscheinlichkeit für bestimmte Markierungen zu bestimmten Zeitpunkten berechnet werden kann, nicht die Erreichbarkeit bestimmter Markierungen.

3.2 Region-based Analysis

Das Funktionsprinzip der *Region-based Analysis* ist es, die s-t-Ebene aller möglichen Zustände des untersuchten HPnG in verschiedene Regionen zu zerlegen. Die Regionen sind dabei analog zu den *Parametric Locations* der *Parametric Reachability Analysis*, d.h. auch sie haben eine konstante Markierung der diskreten Stellen, eine linear von s abhängige Markierung der kontinuierlichen Stellen zu Beginn, einen konstanten Drift der kontinuierlichen Stellen und werden jeweils durch das Auftreten von Ereignissen von den folgenden Regionen getrennt.

Da die Zustände des Modells auf die s-t-Ebene abgebildet werden, muss ein Paar $\langle t, s \rangle$ eindeutig einen Zustand identifizieren. Deswegen können mit diesem Verfahren nur die HPnGs untersucht werden, bei denen keine indeterministische Konfliktauflösung für das Feuern diskreter Transitionen angewandt werden muss, denn durch diese gibt es neben der generellen Transition eine weitere Quelle von Indeterminismus im Modell.

Die Variable s bezeichnet bei der *Region-based Analysis* den Zeitpunkt, zu dem die generelle Transition feuert, im Gegensatz zur Schaltzeit der generellen Transition bei der *Parametric Reachability Analysis*. Die beiden Bedeutungen von s unterscheiden sich folglich genau in den Modellen, in denen die generelle Transition nicht ständig aktiviert ist. Denn dann

korrespondiert der Zeitpunkt, zu dem die generelle Transition schaltet, nicht mit der Schaltzeit der generellen Transition. Dadurch verkompliziert sich die Berechnung der Wahrscheinlichkeit zu einer STL-Formel, welche in 3.2.1 besprochen wird.

Die grafische Darstellung aller Regionen in der Ebene – mit s auf der horizontalen und t auf der vertikalen Achse – wird *Stochastic Time Diagram* (STD) genannt. Dieses ist durch die Gerade $g(s) = s$, welche das Schalten der generellen Transition repräsentiert, in zwei Bereiche aufgeteilt. In dem Bereich unter der Geraden – deterministischer Bereich genannt – kann die generelle Transition noch nicht gefeuert haben, da die Zeit t im Modell noch nicht den Schaltzeitpunkt s der generellen Transition erreicht hat. Daher befindet sich das Modell hier für alle Werte von s im gleichen Zustand. Der Bereich ist lediglich in der t-Dimension durch das Eintreten von Ereignissen in verschiedene deterministische Regionen aufgeteilt.

Im Bereich über der Geraden – stochastischer Bereich genannt – hat die generelle Transition bereits gefeuert. Deswegen kann dieser Bereich sowohl in t- als auch in s-Dimension in verschiedene stochastische Regionen zerlegt sein. Der stochastische Bereich ist nicht zwangsläufig vollständig von Regionen abgedeckt, denn die generelle Transition kann nicht zu Zeitpunkten feuern, an denen sie deaktiviert ist. Ein Punkt $\langle t, s\rangle$ im stochastischen Bereich ist deswegen genau dann abgedeckt, wenn die generelle Transition zum (deterministischen) Zeitpunkt $t = s$ im Modell aktiviert war.

Die Berechnung des STD erfolgt in zwei Schritten: Zunächst werden die deterministischen Regionen im deterministischen Bereich bestimmt. Dafür wird das Modell bis zu einer oberen Schranke $t_{\max}$ für t ausgewertet, ohne die generelle Transition zu berücksichtigen. Für auftretende Ereignisse werden dabei horizontalen Linien – jeweils zur entsprechenden Zeit der Auftretens – im deterministischen Bereich des STD eingefügt. Diese grenzen die deterministischen Regionen voneinander ab. Deterministische Regionen sind links durch die Gerade g begrenzt, rechts sind sie offen.

Dann werden die stochastischen Regionen im stochastischen Bereich bestimmt. Dafür wird von den Liniensegmenten ausgegangen, die beim Aufteilen der Geraden g durch die horizontalen Linien des deterministischen Bereichs entstehen. Es werden dabei nur die Liniensegmente berücksichtigt, bei denen die generelle Transition aktiviert ist. Jedes solche Segment im s-Intervall $[l, r)$ bildet den Ausgangspunkt für eine neue Region. Links und rechts ist die Region durch die Geraden $m(t) = l$ und $n(t) = r$ begrenzt. Zu dieser Region werden dann die Teilintervalle von $[l, r)$ mit dem jeweils als nächstes auftretenden Ereignis bestimmt. Für jedes dieser Teilintervalle wird die Region oben durch die Gerade begrenzt, die den Zeitpunkt für das Auftreten des entsprechenden Ereignisses darstellt. Damit ergibt sich ein Polygon im STD. Die oberen Grenzen für das Polygon werden weiterhin als Ausgangssegmente für die Berechnung der folgenden stochastischen Regionen verwendet. Die Berechnung des STD endet, wenn die obere Grenze $t_{\max}$ erreicht ist.

Die Berechnung der Regionen im stochastischen Bereich ist prinzipiell analog zu der

Berechnung der *Parametric Locations*. Wie bei den *Parametric Locations* hat auch jede Region im stochastischen Bereich genau einen Vorgänger. Deswegen wird der Raum aller möglichen Zustände bei der Anwendung beider Verfahren auf das gleiche Modelle in gleicher Weise zerlegt.

3.2.1 Model Checking

Dadurch, dass das indeterministische Feuern sofortiger oder deterministischer Transitionen für dieses Verfahren nicht zulässig ist, lassen sich folgende Annahmen treffen:

(i) Da ein von einer Region abgedecktes Paar $\langle t, s\rangle$ eindeutig einen Zustand $\Gamma(s,t)$ identifiziert, lässt sich einer STL-Formel Ψ im Bezug ein ein solches Paar $\langle t, s\rangle$ ein Wahrheitswert zuordnen: $\Gamma(s,t) \models^{s,t} \Psi$, wobei $\models^{s,t}$ der Wahrheitsrelation entspricht.

(ii) Durch diesen eindeutigen Wahrheitswert lässt sich für elementare Formeln (vgl. 2.3 Ausdrücke (a), (b), (c) und (f)) zu einem festen Zeitpunkt t eine Menge von Teilintervallen von $[0,\infty)$ für s finden, für die diese Formel im Modell gültig ist.

(iii) Zusammengesetzte Formeln (vgl. 2.3 Ausdrücke (d) und (e)) können dann durch Operationen auf den Mengen von Teilintervallen umgesetzt werden. Für die Negation muss ein Intervall M über $[0,\infty)$ invertiert werden ($[0,\infty) \setminus M$) und für die Konjunktion müssen zwei Intervalle M_1, M_2 geschnitten werden ($M_1 \cap M_2$).

Da innerhalb einer Region die diskreten Markierungen konstant und die kontinuierlichen Markierungen nur linear von s und t abhängig sind, lassen sich für eine Region leicht die Intervalle bestimmen, in denen eine STL-Formel über diese Markierungen zum Zeitpunkt t gültig ist. Eine Formel über eine diskrete Stelle ist dabei entweder vollständig oder gar nicht in der Region gültig, für eine Formel über eine kontinuierliche Stelle lässt sich das Intervall geometrisch bestimmen.

Die Bestimmung der Intervalle, in denen eine Formel der Form $\Psi_1 \mathcal{U}^{[T_1,T_2]} \Psi_2$ zum Zeitpunkt t gültig ist, ist – weil auch zukünftige Zustände berücksichtigt werden müssen – komplexer. Das prinzipielle Verfahren zur Bestimmung dieser Intervalle ist es, zunächst die Intervalle zu bestimmen, in denen Ψ_1 zum Zeitpunkt t wahr ist. Dann wird schrittweise der Betrachtungszeitpunkt im Modell erhöht und die Intervalle dabei jeweils auf diejenigen Intervalle beschränkt, in denen Ψ_1 auch weiterhin gültig ist. Das wird für jeden Punkt s in den Intervallen solange fortgesetzt, bis entweder ein Zeitpunkt $t_2 \in [t+T_1, t+T_2]$ mit $\Gamma(s,t_2) \models^{s,t} \Psi_2$ gefunden oder die obere Zeitschranke $t+T_2$ erreicht wurde. Im ersten Fall ist die Formel zum Punkt s gültig, im zweiten Fall wird der Punkt verworfen. Eine detailliertere Beschreibung des Verfahrens findet sich in der Arbeit von Ghasemieh et al. [8].

Für die Berechnung der Wahrscheinlichkeit der gesamten Formel werden für jede Region, welche zum Zeitpunkt t gültig ist, für alle elementaren Teilausdrücke der Formel diejenigen

Intervalle bestimmt, in denen der Teilausdruck gültig ist. Die Ergebnisse aller Regionen werden zusammengefasst und ggf. gemäß (iii) rekursiv kombiniert. Das Resultat ist eine Menge $M = \{[l_1, r_1) \dots [l_n, r_n)\}$ von disjunkten Teilintervallen.

Aufgrund der von der *Parametric Reachability Analysis* abweichenden Belegung von s beziehen sich die Intervalle dieser Menge jedoch auf Zeitpunkte, zu denen die generelle Transition schaltet, nicht auf die Schaltzeit der generellen Transition. Die Schaltzeit wird jedoch benötigt, wenn die Intervalle in Bezug zu der Zufallsverteilung für die Schaltzeit gesetzt werden soll. Deswegen wird eine Funktion $h : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$ benötigt, die zu einem Schaltzeitpunkt die entsprechende Schaltzeit liefert. Eine derartige eindeutige Umrechnung ist möglich, da das Modell ohne das indeterministische Feuern diskreter Transitionen und vor dem Feuern der generellen Transition komplett deterministisch ist. Somit gibt es eine eindeutige Menge von Zeitintervallen, in denen die generelle Transition aktiviert ist. Dekonditioniert man den Schaltzeitpunkt über die Aktivierungsintervalle, so ergibt sich die Schaltzeit. Für den trivialen Fall, dass die generelle Transition nach einer Verzögerung c dauerhaft aktiviert ist, ergibt sich so beispielsweise

$$h(s) = s - c$$

Damit kann die Wahrscheinlichkeit zur gesamten Formel berechnet werden mit

$$\sum_{i \in \{1 \dots n\}} [cdf(h(r_i)) - cdf(h(l_i))] \tag{3.2}$$

Dabei gilt, dass $cdf : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$ die kumulative Wahrscheinlichkeitsfunktion für s ist. Für jede kumulative Wahrscheinlichkeitsfunktion gilt $cdf(\infty) = 1$, daher lässt sich so auch die Wahrscheinlichkeit zu unbegrenzten Intervallen bestimmen. Da die Berechnung der Intervalle exakt erfolgt, ist auch diese berechnete Wahrscheinlichkeit exakt und keine Annäherung. Die Genauigkeit ist dabei jedoch durch die verwendeten Datentypen beschränkt.

3.2.2 Einschränkungen

Eine offensichtliche Einschränkung des Verfahrens ist die Beschränkung auf Modelle ohne indeterministisches Feuern sofortiger und deterministischer Transitionen. In einem Modell tritt beispielsweise dann kein indeterministisches Feuern auf, wenn paarweise alle sofortigen Transitionen, die durch Eingabe-Bögen mit der gleichen diskreten Stelle verbunden sind, eine unterschiedliche Priorität besitzen. Analog muss gleiches auch für alle zwei deterministischen Transitionen mit der gleichen Eingabe-Stelle gelten. Eine Folge der Einschränkung ist aber, dass sich bestimmte (indeterministische) hybride Systeme mit diesem Verfahren nicht untersuchen lassen.

Eine weitere mögliche Einschränkung stellt die Verwendung der kumulativen Verteilungsfunktion zur Berechnung der Wahrscheinlichkeit dar. Nicht für jede Zufallsverteilung lässt

sich die kumulative Verteilungsfunktion in einer geschlossenen Form angeben. Für solche Zufallsverteilungen kann dann aber eine Diskretisierung der Zufallsverteilung über die berechneten Intervalle durchgeführt werden. Dadurch wird jedoch die exakte Lösung zugunsten einer Annäherung aufgegeben und es muss, wie beim *Model Checking* der *Parametric Reachability Analysis*, eine Abwägung zwischen Genauigkeit und Rechenaufwand stattfinden.

Die Berechnung der Intervalle für den *until*-Operator – wie sie in [8] beschrieben ist – ist in der Implementierung bereits umgesetzt. Die Implementierung hat jedoch derzeit noch die Einschränkung, dass keine verschachtelten *until*-Operatoren verwendet werden können.

4 Anwendungsbeispiel

Als Anwendungsbeispiel für den Vergleich der beiden Verfahren soll ein Modell für ein hybrides System einer kritischen Infrastruktur erstellt werden. Dazu werden in Kapitel 4.1 der reale Bezugspunkt des Anwendungsbeispiels und die Hintergründe für die Modellierung erläutert. In Kapitel 4.2 wird dann die konkrete Modellierung des Fallbeispiels als HPnG besprochen.

4.1 Hintergrund

Das Anwendungsbeispiel modelliert die Wasserversorgung einer Kleinstadt in einer Gebirgsregion. Bestandteile der Wasserversorgung sind ein großes Hauptreservoir in der Nähe der Stadt und mehrere Gebirgsquellen, denen jeweils noch kleine Zwischenreservoire angeschlossen sind. Dabei liegt eine der Quellen (Hauptquelle) niedriger als die Stadt, das Wasser muss daher mit einer Pumpe vom Zwischen- zum Hauptreservoir gepumpt werden. Die restlichen Quellen und ihre Zwischenspeicher liegen höher als der Hauptspeicher. Das Anwendungsbeispiel wurde in Anlehnung an die Wasserversorgung von Axenfels entworfen [1].

Für dieses Wasserversorgungssystem soll ein *Given the Occurrence Of Disaster*-Szenario (GOOD-Szenario) modelliert werden. Dabei wird ein konkreter Ausfall des Systems untersucht; es geht nicht um die Fragen, ob und wie es zu einem solchen Ausfall kommen kann [3]. So können beispielsweise die Wahrscheinlichkeit für das Unterschreiten bestimmter Leistungsparameter oder die Dauer bis zur Wiederherstellung der vollständigen Funktionsfähigkeit des Systems bestimmt werden. Modellierungen von GOOD-Szenarien sind deswegen insbesondere zur Untersuchung kritischer Infrastruktur hilfreich, denn diese müssen auch im Falle eines Ausfalls resistent gegen einen Zusammenbruch des Systems sein. Mit Hilfe der in dieser Arbeit beschriebenen Formalismen und Verfahren lassen sich die Risiken im Falle eines Ausfalls abschätzen und mögliche Konsequenzen schon vor dem Ernstfall ziehen.

Für das konkrete Anwendungsbeispiel soll ein Ausfall der Pumpanlage an der Hauptquelle nach einer festen Verzögerung modelliert werden. Durch den Ausfall verringert sich der Zufluss in das Hauptreservoir, so dass sich das Hauptreservoir für die Zeit des Ausfalls entleert. Die Reparaturzeit für die Pumpanlage ist zufällig verteilt, nach der Reparatur ist die Pumpanlage sofort wieder einsatzfähig.

Untersucht werden soll für dieses Modell die Wahrscheinlichkeit dafür, dass der Wasserspiegel im Hauptreservoir unter einen kritischen Wert fällt, denn dann kann die Wasserversorgung

in der Stadt nicht mehr garantiert werden. Dafür wird das Modell bis zu einer Maximalzeit ausgewertet und stündlich die Wahrscheinlichkeit für das Unterschreiten des kritischen Werts berechnet.

Das Anwendungsbeispiel ist skalierbar, da die Anzahl der zusätzlichen Quellen neben der Hauptquelle verändert werden kann. Dabei sollen für alle zusätzlichen Quellen unterschiedliche Raten für die Wassergewinnung gewählt werden. Im Kontext dieser Arbeit werden Skalierungen des Fallbeispiels mit 1-35 zusätzlichen Quellen analysiert.

4.2 Modellierung als HPnG

Abbildung 4.1 zeigt, wie das Modell als HPnG umgesetzt werden kann. Dabei ist die Skalierung $n \in \{1 \dots 35\}$ die Anzahl der zusätzlichen Quellen neben der Hauptquelle. Als Einheiten werden Stunden (für die Zeit), Kubikmeter (für die Kapazitäten und Markierungen der kontinuierlichen Stellen) und Kubikmeter pro Stunde (für die Flussraten) angenommen.

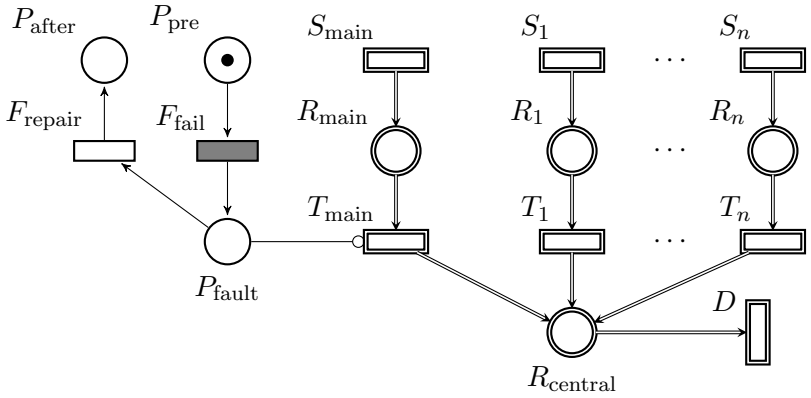

Abbildung 4.1: ***Schematische Darstellung des Fallbeispiels in grafischer HPnG-Notation***

Für jede Quelle $x \in \{\text{main}, 1 \dots n\}$ ist S_x die eigentliche Quelle des Wassers, R_x ist das Zwischenreservoir und T_x ist die Pump- bzw. Transportleitung zum Hauptreservoir R_{central}. Das Hauptreservoir hat eine Kapazität von 200 und einen anfänglichen Wasserstand von 150. Die Hauptquelle hat eine nominale Flussrate von 40, eine Zwischenreservoir-Kapazität von 40 (mit einem anfänglichen Wasserstand von 20) und kann Wasser mit einer Rate von 48 zum Hauptreservoir transportieren.

Alle anderen Quellen haben jeweils unterschiedliche nominale Flussraten im Bereich $(0, 70]$, eine Zwischenreservoir-Kapazität von $\frac{70}{n}$ (mit einem anfänglichen Wasserstand von $\frac{70}{2n}$) und die Transportrate zum Hauptreservoir beträgt das 1.2-fache der nominalen Flussrate der Quelle. Die Quellraten wurden gleich und die Kapazitäten unterschiedlich gewählt, damit die Zwischenreservoire zu jeweils unterschiedlichen Zeitpunkten die obere bzw. untere Grenze erreichen.

Die Stelle P_{fault} enthält genau dann ein Token, wenn die Pumpe durch Ausfall deaktiviert ist. Sie erhält ihr Token nach einer konstanten Ausfallzeit über die deterministische Transition F_{fail} und gibt es nach zufälliger Zeit über die generelle Transition F_{repair} wieder ab. Wenn die Stelle P_{fault} ein Token besitzt, inhibiert sie die Transition T_{main}, wodurch der Fluss vom Zwischenreservoir der Hauptquelle zum Hauptreservoir unterbrochen wird. Die Ausfallzeit beträgt 1 und die Reparaturzeit ist gemäß der gefalteten Normalverteilung mit $\mu = 6$, $\sigma = 2$ verteilt.

Die Transition D modelliert den Wasserverbrauch durch die Stadt. Sie ist direkt an das Hauptreservoir angeschlossen und hat eine nominale Flussrate von 100. Dadurch ist die gesamte Rate aller Quellen mit 110 höher als der Verbrauch, ohne die Hauptquelle ist die Rate (70 als Summe aller zusätzlichen Quellen) jedoch niedriger als der Verbrauch.

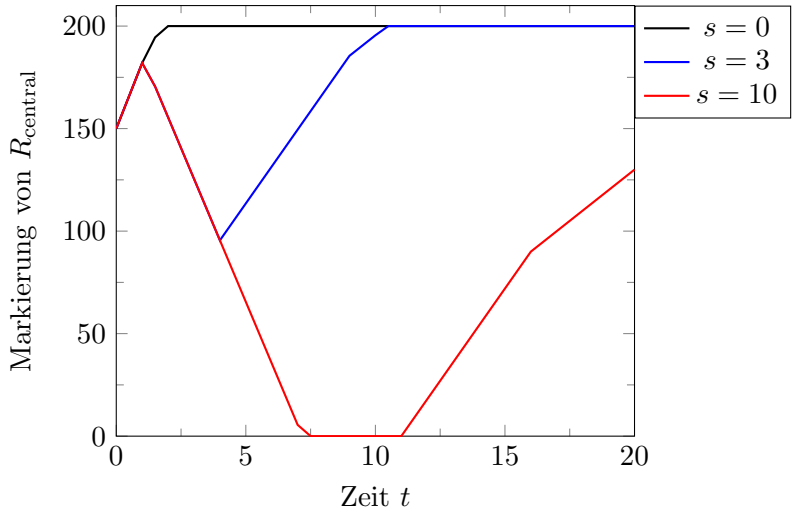

Abbildung 4.2: ***Markierung der Stelle $R_{central}$ für das Fallbeispiel mit der Skalierung $n = 1$ in Abhängigkeit von der Zeit t für verschiedene Schaltzeiten s der generellen Transition***

Das Modell soll für einen Zeitraum von $t_{\max} = 20$ Stunden untersucht werden. Abbildung 4.2 zeigt den Verlauf des Wasserstandes im Hauptreservoir für drei verschiedene Reparaturzeiten s in einem Modell mit einer zusätzlichen Quelle ($n = 1$). Dabei ist deutlich zu erkennen, dass der Wasserstand umso tiefer fällt und länger niedrig bleibt, je länger die Reparatur dauert.

Als kritischer Wasserstand soll gelten, wenn der Wasserspiegel im Hauptreservoir unter einen Level von 30[1] fällt. In diesem Fall könnte die Wasserversorgung der Stadt nicht garantiert werden. So könnte beispielsweise ausreichend Wasser für Löscharbeiten fehlen. Ein Unterschreiten des kritischen Wasserstandes lässt sich mit der STL-Formel

$$\neg(x_{R_{\text{central}}} >= 30) \tag{4.1}$$

[1] Dieser Wert wurde frei gewählt als eine plausible untere Schwelle, bis zu der eine Wasserversorgung der Stadt garantiert werden kann.

überprüfen. Die Wahrscheinlichkeit für das Auftreten von kritischen Wasserständen soll von den beiden Verfahren bestimmt werden.

Das Modell besitzt insgesamt 3 diskrete Stellen, jeweils eine deterministische und generelle Transition sowie $2 + n$ kontinuierliche Stellen und $3 + 2n$ kontinuierliche Transitionen. Sowohl die Anzahl der kontinuierlichen Stellen als die die Anzahl der kontinuierlichen Transitionen skaliert damit linear in Abhängigkeit von n. Die Gewichte aller Bögen und deterministischer Transitionen sind 1, die Prioritäten aller kontinuierlichen Bögen und deterministischer Transitionen sind 0, die Anteile aller kontinuierlichen Bögen und Gewichte aller deterministischer Transitionen sind 1.

5 Vergleich der Verfahren

In diesem Kapitel sollen die beiden Verfahren verglichen werden. Dafür werden nacheinander die drei in der Einleitung angeführten Teilfragen beantwortet. Die ersten beiden Teilfragen (a) und (b), welche Modelle und Eigenschaften der Modelle die Verfahren untersuchen können, beziehen sich auf qualitative Aspekte der Verfahren, die z.T. schon in Kapitel 3 behandelt wurden. Daher werden die relevanten Aspekte der Verfahren in den nächsten Absätzen zur Beantwortung der Fragen zusammengefasst. Die Teilfrage (c) soll in Kapitel 5.2 quantitativ durch einen Vergleich der Laufzeit beider Verfahren bei der Analyse des Fallbeispiels beantwortet werden. Dafür wird vorher in Kapitel 5.1 kurz auf die Implementierungen der Verfahren eingegangen.

Auf welche Modelle können die beiden Verfahren angewandt werden? – Beide Verfahren dienen der Analyse von HPnGs. Damit sind sie auf die Untersuchung hybrider PNs festgelegt, die nur eine generelle Transition besitzen, die einmal feuern kann. Prinzipiell ließen sich jedoch beide Verfahren so modifizieren, dass sie auch Modelle mit mehreren generellen Transitionen untersuchen können. Dafür müssten bei der *Parametric Reachability Analysis* zusätzliche Variablen für die Bereiche der Schaltzeiten der weiteren generellen Transition eingeführt werden bzw. bei der *Region-based Analysis* das STD um weitere Dimensionen für weitere Schaltzeitpunkte erweitert werden. Für beide Verfahren würde der Rechenaufwand jedoch immens steigen [9] und die Beschreibungen sowie Implementierungen der Verfahren setzen derzeit jeweils nur eine generelle Transition voraus.

Ein wesentlicher Unterschied zwischen den beiden Verfahren besteht jedoch für Modelle mit indeterministischem Feuern der sofortigen oder deterministischen Transitionen: Solche Modelle lassen sich ausschließlich mit der *Parametric Reachability Analysis* untersuchen. Es ist jedoch fraglich, ob es sich dabei um eine schwerwiegende Einschränkung für die Anwendbarkeit der *Region-based Analysis* handelt, denn nicht alle hybriden Systeme zeigen indeterministisches Verhalten dieser Art. Zudem lassen sich für die *Region-based Analysis* mehrere Modelle mit unterschiedlichen Prioritäten der Transitionen erstellen (bei wenigen zufälligen Ereignissen), oder zufälliges Schalten sofortiger Transitionen lässt sich durch kontinuierliche Transitionen approximieren (bei sehr vielen zufälligen Ereignissen) [5].

Welche Eigenschaften der Modelle können von den Verfahren untersucht werden? – Prinzipiell dienen beide Verfahren der quantitativen Analyse von Modellen, insbesondere der Untersuchung der Wahrscheinlichkeiten für das Auftreten bestimmter Zustände im Modell. Dabei ermöglichen beide Verfahren das Berechnen der Wahrscheinlichkeit dafür,

dass eine bestimmte Markierungskonstellation zu einem bestimmten Zeitpunkt t im Modell auftritt. Bei beiden Verfahren können durch das Verknüpfen elementaren STL-Ausdrücke komplexe Formeln zur Beschreibung von Zuständen gebildet werden.

Hingegen kann die Erreichbarkeit bestimmter Markierungen nur durch Verwendung des *until*-Operators und mit der *Region-based Analysis* untersucht werden. Dieses ermöglicht eine neue Dimension der Analyse eines Modells. Beispielsweise kann festgestellt werden, mit welcher Wahrscheinlichkeit sich ein Modell in einem Zeitraum von einer kritischen zu einer nicht-kritischen Markierung entwickeln kann.

Damit ergibt sich für die Anwendbarkeit der Verfahren ein differenziertes Bild: *Parametric Reachability Analysis* kann zwar auf mehr Modelle angewandt werden, *Region-based Analysis* ermöglicht jedoch eine eingehendere Analyse der Modelle. Diese Beschränkungen sind dabei keine technischen Eigenheiten der beiden Verfahren, denn erst durch die Beschränkung auf deterministische Modelle ist für die *Region-based Analysis* die Verwendung des *until*-Operators in dieser Form möglich.

5.1 Anpassungen der Implementierungen

Für den quantitativen Vergleich der Verfahren wurden die C/C++-Implementierungen beider Verfahren von den Autoren Gribaudo und Ghasemieh zur Verfügung gestellt. Es waren jedoch einige Anpassungen nötig, die hier kurz skizziert werden sollen.

Für die *Parametric Reachability Analysis* waren Funktionen für das Einlesen von Modellen sowie für die Erzeugung des *Parametric Location Tree* bereits implementiert. Ebenfalls vorhanden war eine Funktion $\xi_L^{\Psi}(t \mid s)$ (vgl. 3.1.1). Diese Funktion wurde so modifiziert, dass alle STL-Formel-Ausdrücke (a)-(e) (vgl. 2.3) unterstützt werden. Außerdem wurde eine Funktion hinzugefügt, die die in 3.1.1 erwähnte Diskretisierung der Zufallsverteilung durchführt. Dabei bestehen zwei relevante Unterschiede zu einer naiven Implementierung der Formel (3.1):

(1) Das Sampeln der Werte für s aus der Zufallsverteilung findet in der Mitte der Diskretisierungsintervalle (also mit einem Offset von $\frac{\Delta s}{2}$) statt.

(2) Eine Diskretisierung wird nur für das Intervall $s \in [0, t)$ vorgenommen, wobei gleichzeitig die kumulative Wahrscheinlichkeit c für das Intervall bestimmt wird. Da die generelle Transition noch nicht gefeuert haben kann, gilt für das übrige Intervall $[t, \infty)$ wie in den deterministischen Regionen des STD

$$\forall s_1, s_2 \in [t, \infty) : \xi_L^{\Psi}(t \mid s_1) = \xi_L^{\Psi}(t \mid s_2)$$

Die Summe für die Diskretisierung über das verbleibende Intervall kann deswegen

zusammengefasst werden als

$$(1 - c) * \xi_{L_0}^{\Psi}(t \mid t)$$

Dadurch kann die unendliche Summe in endlich vielen Schritten berechnet werden. In der Implementierung wird das restliche Intervall $[t, \infty)$ ignoriert, wenn die kumulative Wahrscheinlichkeit c bereits größer als 1 ist.

In der Implementierung der *Region-based Analysis* waren ebenfalls bereits die Funktionen zum Einlesen von Modellen, zur Erzeugung des *Stochastic Time Diagram* und zum Berechnen von STL-Formeln vorhanden. Diese Implementierung wurde überarbeitet und es wurden einige Fehler in der Implementierung korrigiert. Die Anpassungen haben keine Auswirkung auf die Laufzeitkomplexität der Implementierung.

5.2 Vergleich am Anwendungsbeispiel

Die Untersuchung des Fallbeispiels durch die beiden Verfahren läuft jeweils in zwei Schritten ab: Zunächst wird die zentrale Datenstruktur – der *Parametric Location Tree* bzw. das *Stochastic Time Diagram* – erzeugt, dann wird die Wahrscheinlichkeit für die STL-Formel (4.1) anhand der Datenstruktur bestimmt (*Model Checking*), jeweils für mehrere Zeitpunkte im Modell. Die beiden Verfahren nehmen bei diesen Schritten jeweils prinzipiell die gleichen Berechnungen vor, deswegen lassen sich die Laufzeiten für den ersten und den zweiten Schritt getrennt vergleichen.

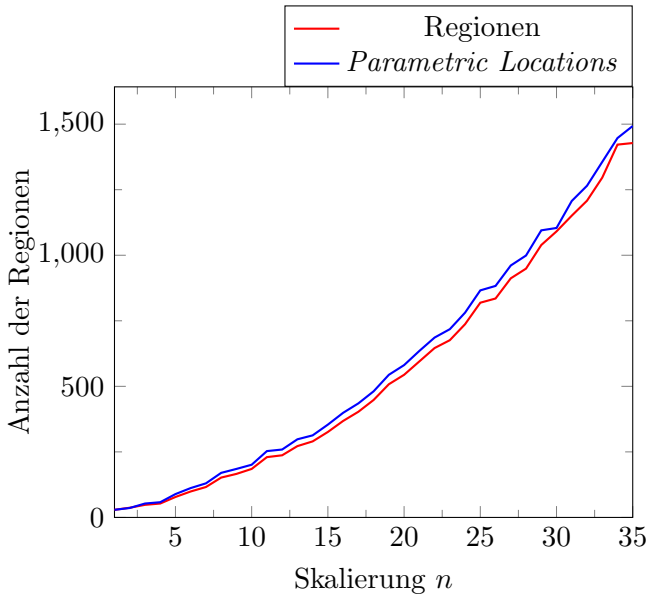

Abbildung 5.1: ***Anzahl der Regionen bei der Analyse des Fallbeispiels in Abhängigkeit von der Skalierung*** n

Zunächst wird die Laufzeit für die Erzeugung der Datenstrukturen betrachtet. Bei der Er-

zeugung wird der Raum der möglichen Zustände des Modells von der *Parametric Reachability Analysis* in verschiedene *Parametric Locations* und von der *Region-based Analysis* in verschiedene (deterministische und stochastische) Regionen zerlegt. Abbildung 5.1 zeigt, wie viele *Parametric Locations* bzw. Regionen[1] dabei für verschiedene Skalierungen n des Fallbeispiels auftreten. In der Abbildung ist ersichtlich, dass die Anzahlen in etwa übereinstimmen, Abweichungen lassen sich durch eine unterschiedliche Behandlung des Auftretens der maximalen Ausführungszeit des Modell erklären. Diese Beobachtung unterstreicht, dass beide Verfahren prinzipiell den gleichen Ansatz zur Strukturierung des Modells verfolgen.

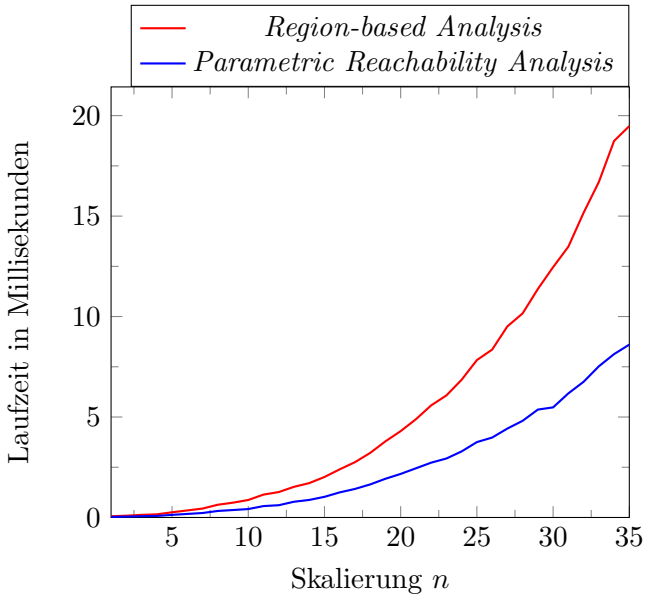

Abbildung 5.2: ***Laufzeit für die Erzeugung der Datenstruktur bei der Analyse des Fallbeispiels in Abhängigkeit von der Skalierung*** n

In Abbildung 5.2 ist dargestellt, wie lange die Erzeugung der Datenstrukturen für beide Verfahren in Abhängigkeit von der Skalierung n des Fallbeispiels dauert. Dabei ist deutlich zu erkennen, dass die Erzeugung des *Parametric Location Tree* weniger Zeit in Anspruch nimmt. Dies deckt sich mit den Beobachtungen von Ghasemieh et al. [7], die eine um den Faktor 3-4 längere Laufzeit für die *Region-based Analysis* beobachtet haben, auch wenn die Unterschiede hier etwas geringer ausfallen. Als eine Ursache für die längere Laufzeit haben die Autoren aufwändigere Berechnungen im Zusammenhang mit der Erzeugung der Polygone der stochastischen Region angegeben.

Die Anzahl der Regionen und damit die Laufzeit für die Erzeugung der Datenstrukturen ist vom konkreten Aufbau des Modells und nicht allein von der Anzahl der Stellen und Transitionen abhängig. Die Laufzeit für die Erzeugung steigt dabei für verschiedene Skalierungen eines Modells mindestens so schnell wie die Anzahl der Regionen, denn die Regionen müssen jeweils einzeln erzeugt werden. Hinzu kommen pro Region Laufzeitkosten für die Flussratenanpassung

[1] *Parametric Locations* und Regionen werden im Folgenden unter dem Begriff Regionen zusammengefasst.

und das Erkennen der nächsten Ereignisse. Diese hängen von der Anzahl der Stellen und Transitionen im Modell ab. Für das konkrete Fallbeispiel steigt die Anzahl der Regionen mit der Skalierung nur langsam an, die Anzahl der Regionen ist für $n = 35$ nur etwa 50x so groß wie für $n = 1$. Deswegen wächst auch die benötigte Laufzeit zur Erzeugung der Datenstrukturen nur moderat und bleibt unter 20 Millisekunden.

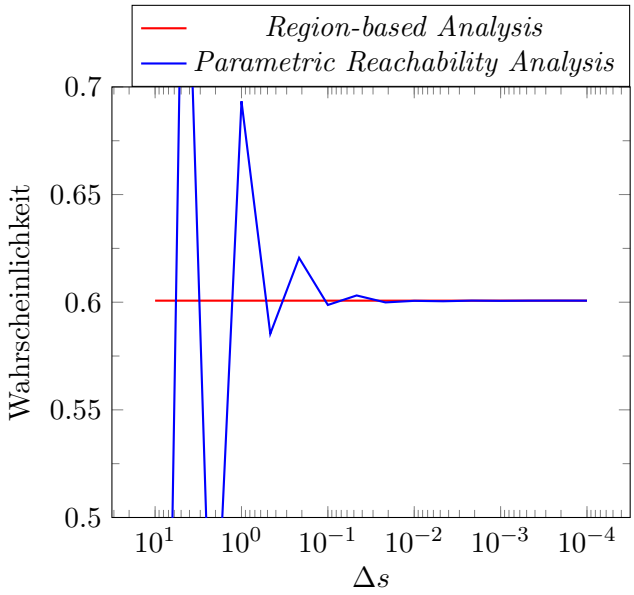

Abbildung 5.3: ***Von Parametric Reachability Analysis bestimmte Wahrscheinlichkeit für die STL-Formel*** (4.1) ***zum Zeitpunkt*** $t = 7$ ***in Abhängigkeit von der Diskretisierungsschrittgröße*** Δs

Um die Laufzeit für das *Model Checking* vergleichen zu können, muss eine passende Diskretisierungsschrittgröße Δs für die *Parametric Reachability Analysis* gewählt werden, denn diese beeinflusst maßgeblich die Laufzeit. Wie bereits in Kapitel 3.1.1 bemerkt, ist dabei eine Abwägung zwischen Genauigkeit und Laufzeit erforderlich. Abbildung 5.3 zeigt die Genauigkeit der *Parametric Reachability Analysis* für verschiedene Diskretisierungsschrittgrößen. Zum Vergleich ist dort auch die von der *Region-based Analysis* berechnete Wahrscheinlichkeit abgebildet, auch wenn ihre Berechnung unabhängig von der Diskretisierungsschrittgröße ist. Die gemessene Wahrscheinlichkeit bezieht die STL-Formel (4.1) auf das Fallbeispiel mit der Skalierung $n = 1$ zur Zeit $t = 7$.

Für die weiteren Vergleiche der Laufzeit wird die Diskretisierungsschrittgröße mit $\Delta s = 10^{-2}$ angenommen. In der Abbildung ist erkennbar, dass durch diese Schrittgröße bereits eine hohe Genauigkeit erreicht wird. Dazu ist jedoch anzumerken, dass eine hohe Genauigkeit des Diskretisierungs-Ansatzes der *Parametric Reachability Analysis* hier durch zwei Faktoren begünstigt wird:

(a) Die verwendete gefaltete Normalverteilung mit $\mu = 6$, $\sigma = 2$ besitzt eine hohe Varianz und verläuft deswegen mit betragsmäßig geringer Steigung. Dadurch sinkt der Fehler

durch die Diskretisierung in der Formel (3.1). Für Wahrscheinlichkeitsverteilungen mit einer geringeren Varianz müssten ggf. kleinere Diskretisierungsschrittgrößen gewählt werden.

(b) Für jede Zeit t gibt es im Modell – wenn überhaupt – lediglich ein geschlossenes Intervall $[s, \infty)$ für die Schaltzeit der generellen Transition, bei der der kritische Wert im Hauptreservoir unterschritten wird. Durch diese geringe Fragmentierung der Intervalle sinkt der Fehler durch die Abtastung in der Formel (3.1).

Diese Faktoren bewirken eine gleichmäßige Annäherung an das korrekte Ergebnis durch eine Verringerung der Schrittgröße und ermöglichen eine hohe Genauigkeit durch die gewählte Schrittgröße $\Delta s = 10^{-2}$. Die Wahl ist damit für dieses Fallbeispiel angemessen, jedoch keineswegs universell gültig. Ghasemieh et al. [7] verwendeten für ihren Vergleich der beiden Verfahren eine Schrittgröße $\Delta s = 0.5 * 10^{-2}$. Im Vergleich zu ihrer Arbeit ist deswegen für dieses Fallbeispiel eine halbierte Laufzeit für das *Model Checking* mit der *Parametric Reachability Analysis* zu erwarten.

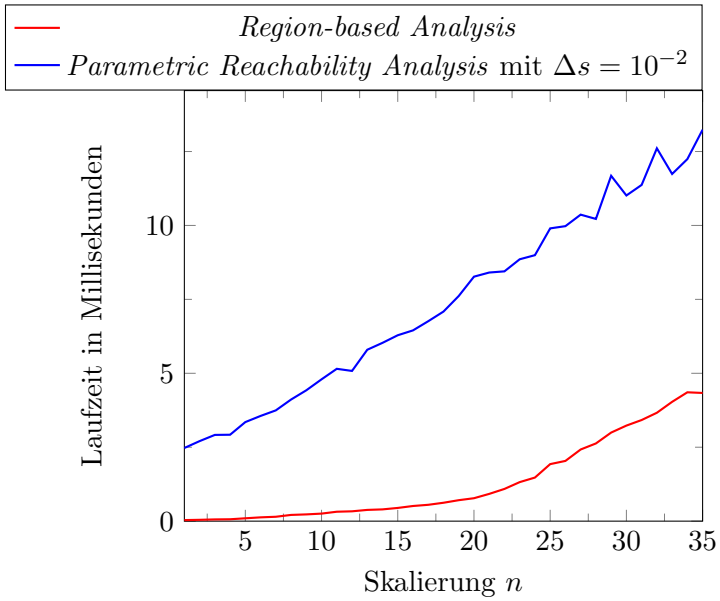

Abbildung 5.4: ***Laufzeit für das Model Checking des Fallbeispiels mit der STL-Formel*** (4.1) ***in Abhängigkeit von der Skalierung*** n

Abbildung 5.4 zeigt, wie lange das *Model Checking* für die beiden Verfahren in Abhängigkeit von der Skalierung des Fallbeispiels dauert. Dazu wurde die STL-Formel (4.1) zu den Zeitpunkten $t \in \{0 \dots 19\}$ auf das Fallbeispiel bezogen. Die Laufzeit ist für jede Skalierung jeweils die summierte Laufzeit für das *Model Checking* zu allen Zeitpunkten t.

An der Abbildung lassen sich zwei Beobachtungen machen: Einerseits ist deutlich geringere Laufzeit beim *Model Checking* durch die *Region-based Analysis* zu beobachten. Dies fällt insbesondere für kleinere Skalierungen des Fallbeispiels auf, für die das Verfahren weniger als eine Millisekunde benötigt. Diese Laufzeitunterschiede passen zu den Beobachtungen von

Ghasemieh et al. [7], in deren Untersuchung die *Parametric Reachability Analysis* um den Faktor 20-100 langsamer war.

Andererseits sind für das vorliegende Fallbeispiels derart große Unterschiede bezüglich der Laufzeit nur bei kleinen Skalierungen ($n \leq 14$) zu beobachten, für höhere Skalierungen ist die *Parametric Reachability Analysis* nur etwa um den Faktor 3-15 langsamer. Diese Anomalie bleibt auch bestehen, wenn die zu erwartende halbierte Laufzeit gegenüber der Arbeit von Ghasemieh et al. [7] berücksichtigt wird. Insgesamt ist zu erkennen, dass die prozentualen Laufzeitunterschiede für größere Skalierungen dieses Fallbeispiels tendenziell geringer werden.

Der Grund für diese unterschiedlichen Entwicklungen der Laufzeit in Abhängigkeit von der Skalierung des Fallbeispiels liegt in den unterschiedlichen Verfahren zur Bestimmung der Wahrscheinlichkeiten: Für die *Parametric Reachability Analysis* wird, wie in Formel (3.1) erkennbar, die Schaltzeit s der generellen Transition diskretisiert. Die Diskretisierungsschrittgröße Δs ist damit ein wichtiger Faktor für die Laufzeit. Ein weiterer Faktor ist die Laufzeit der Funktion $\xi_L^{\Psi}(t \mid s)$. Diese Funktion wird für jeden Iterationsschritt neu ausgewertet, wobei jeweils im *Parametric Location Tree* die gültige *Parametric Location* für das aktuelle Paar $\langle t, s \rangle$ gefunden werden muss. Da das *Model Checking* über die gesamte Zeitspanne vorgenommen wird, für die das Modell ausgewertet wurde, hängt die Laufzeit für die Auswertung dieser Funktion damit insgesamt auch von der Tiefe des *Parametric Location Tree* ab.

Für die *Region-based Analysis* wird jede Region des *Stochastic Time Diagram* auf eine Berührung mit der Geraden zur aktuellen Zeit t überprüft. Damit ist für dieses Verfahren die Anzahl der Regionen ein linearer und der maßgebliche Faktor für die Laufzeit. Für die gültigen Regionen zur Zeit t müssen zusätzlich die Intervalle für s bestimmt werden, in denen die Formel wahr ist. Die Anzahl der gültigen Regionen entspricht, auf den *Parametric Location Tree* übertragen, der Breite des Baums zur Zeit t.

Diese unterschiedlichen Verfahren erklären die unterschiedlichen Entwicklungen der Laufzeit in Abhängigkeit von der Skalierung, die in Abbildung 5.4 erkennbar sind. Die aktuelle Implementierung des *Model Checking* für die *Region-based Analysis* hat eine geringere Laufzeit für Modelle mit wenigen Regionen, wobei die Laufzeit linear mit der Anzahl der Regionen ansteigt. Die Implementierung des *Model Checking* für die *Parametric Reachability Analysis* ist durch die Diskretisierung relativ langsam für Modelle mit wenigen Regionen. Ihre Laufzeit steigt aber dadurch, dass sie nur von der Tiefe des *Parametric Location Tree* abhängt, weniger schnell an als die der *Region-based Analysis*. So könnte die *Parametric Reachability Analysis* für sehr große Modelle ($n > 100$) ein schnelleres *Model Checking* als die *Region-based Analysis* ermöglichen.

Die Abbildung zeigt für die *Parametric Reachability Analysis* einige Ausschläge bezüglich der Laufzeit ($t = 29$, $t = 32$). Diese können anhand der Algorithmen nicht erklärt werden. Möglicherweise handelt es sich dabei um Messfehler.

Abschließend ist in Abbildung 5.5 die Laufzeit für die Auswertung der STL-Formel (4.1)

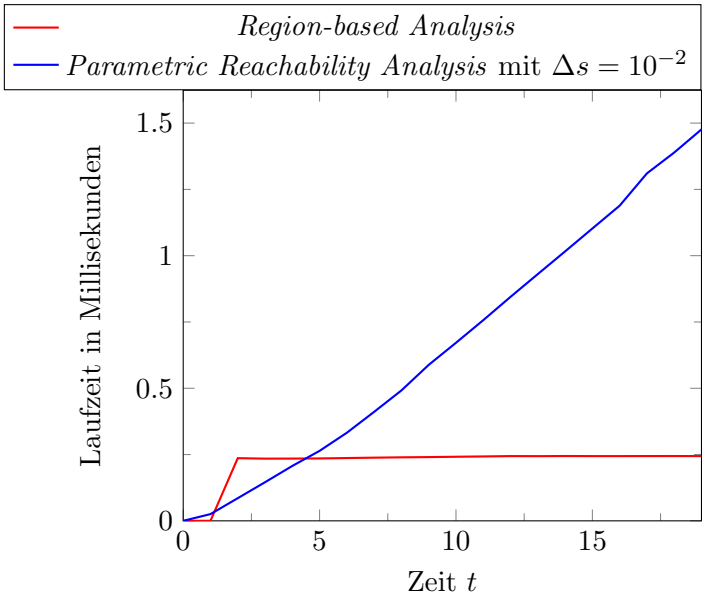

Abbildung 5.5: ***Laufzeit für das Überprüfen der STL-Formel*** (4.1) ***im Bezug auf das Fallbeispiels mit der Skalierung*** $n = 35$ ***in Abhängigkeit von der Zeit*** t

im Bezug auf verschiedene Zeitpunkte t eines Modells dargestellt. Als Modell dient hierzu das Fallbeispiel mit einer Skalierung $n = 35$. Für die *Region-based Analysis* ist hierbei eine etwa konstante Laufzeit zu beobachten, da unabhängig von der Zeit t immer alle Regionen betrachtet werden. Dabei hat das Berechnen der Intervalle für die jeweils gültigen Regionen offensichtlich kaum einen Einfluss auf die Laufzeit. Anders verhält es sich bei den Zeiten $t \leq 1$, da für diese Zeiten aufgrund der verzögerten Aktivierung der generellen Transition überhaupt keine stochastischen Regionen betrachtet werden müssen. Dieser Fall ist in der Implementierung separat berücksichtigt.

Für die *Parametric Reachability Analysis* steigt die Laufzeit etwa linear mit der betrachteten Zeit t im Modell. Dafür gibt es zwei Gründe: Einerseits muss die Diskretisierung, wie in Kapitel 5.1 beschrieben, über ein wachsendes Intervall $[0, t)$ durchgeführt werden. Andererseits muss die Funktion $\xi_L^{\Psi}(t \mid s)$ den *Parametric Location Tree* für ein wachsendes t bis zu einer größeren Tiefe durchsuchen.

Das *Model Checking* beider Verfahren bietet noch Raum für weitere Optimierungen. Dabei kann insbesondere ausgenutzt werden, dass zum Überprüfen einer Formel die gültigen Regionen zu einer konstanten Zeit t benötigt werden. Zum Beispiel könnte zu jeder stochastischen Region des *Stochastic Time Diagram* diejenige bezüglich s nächste Region hinterlegt sein, die als erste einen Teil des Zeitbereichs der aktuellen Region abdeckt. Damit wäre ein schnelleres Finden der nächsten gültigen ohne ein Überprüfen aller Regionen möglich.

Prinzipiell weisen beide Verfahren zwei unterschiedliche Laufzeitverhalten für das Überprüfen einer Formel auf: Die Laufzeit für die *Parametric Reachability Analysis* hängt immer zumindest

von der Anzahl der Diskretisierungsschritte (und damit von t und Δs) ab. Hingegen hängt die Laufzeit für die *Region-based Analysis* immer zumindest von der Breite des Baums der Regionen zur Zeit t ab.

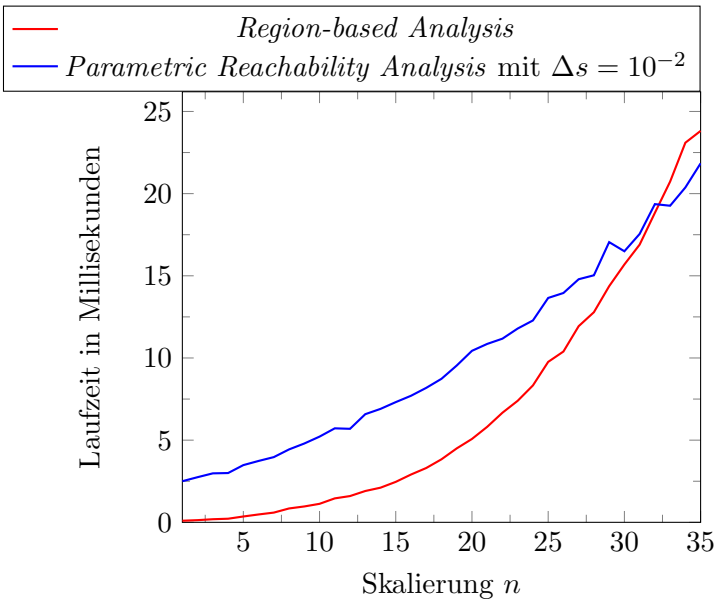

Abbildung 5.6: ***Laufzeit für die Analyse des Fallbeispiels in Abhängigkeit von der Skalierung*** n

Abbildung 5.6 zeigt die summierte Laufzeit der Teilschritte bei beiden Verfahren (vgl. Abb. 5.2 und Abb. 5.4). Diese Abbildung kann für einen Versuch der Beantwortung von Teilfrage (c) herangezogen werden. Für das aktuelle Fallbeispiel sind die Unterschiede bezüglich der Laufzeit zwischen beiden Verfahren gering. Dennoch ist bis zu einer Skalierung von 32 die *Region-based Analysis*, für größere Skalierungen jedoch die *Parametric Reachability Analysis* schneller. Die *Region-based Analysis* ist für kleinere Skalierungen des Modells aufgrund der Geschwindigkeitsvorteile beim *Model Checking* schneller. Für größere Skalierungen wird dieser Vorteil jedoch durch die schnellere Erstellung der Datenstruktur bei der *Parametric Reachability Analysis* ausgeglichen.

Die in der Arbeit von Ghasemieh et al. [7] festgestellten deutlichen Vorteile der *Region-based Analysis* bezüglich der Laufzeit lassen sich demnach für dieses Fallbeispiel nicht bestätigen. Es lassen sich aber auf Basis der Ergebnisse des Vergleichs einige Kriterien identifizieren, anhand derer sich für einzelne Anwendungsfälle das jeweils besser geeignete Verfahren auswählen lässt:

- Die *Region-based Analysis* liefert exakte Ergebnisse. Sie ist deswegen vorzuziehen, wenn eine hohe Genauigkeit gefordert ist.
- Die *Region-based Analysis* ermöglicht außerdem ein schnelles *Model Checking*. Sie ist in den meisten Fällen gut geeignet, wenn für wenige Modelle viele Formeln zu vielen Zeitpunkten überprüft werden sollen.

- Die *Parametric Reachability Analysis* kann schnell grobe Annäherungen liefern. Sie kann beispielsweise dazu verwendet werden, viele Modelle schnell zu untersuchen, wenn keine exakten Ergebnisse erforderlich sind.
- Die *Parametric Reachability Analysis* benötigt weniger Zeit, um die Datenstruktur zu erzeugen. Damit ist sie gut geeignet, wenn für viele Modelle wenige Formeln überprüft werden sollen.

Insgesamt sollte die Verwendung der *Region-based Analysis* bevorzugt werden, wenn keine der oben genannten Kriterien für eine Verwendung der *Parametric Reachability Analysis* eindeutig zutreffen. Gründe dafür sind die Genauigkeit der *Region-based Analysis* und die Möglichkeit zur Verwendung des *until*-Operators.

Alle Berechnungen wurden auf demselben Computer mit folgender Ausstattung vorgenommen:

- Prozessor: AMD Phenom II X4 965
- Arbeitsspeicher: 8GB mit einem Speichertakt von 667MHz
- Mainboard: MSI 970A-G46
- Betriebssystem: Linux-4.7.4 (64bit; Archlinux)

Die für diesen Vergleich verwendeten angepassten Implementierungen der Verfahren sind auf der der Arbeit beiliegenden CD enthalten. Zur Bestimmung der Laufzeiten wurden die Berechnungen mit Hilfe eines Python-Skriptes in 50facher Wiederholung durchgeführt und der Durchschnitt über die ermittelten Laufzeiten gebildet. Dieses Skript sowie ein weiteres zur Erzeugung der HPnG-Modelle findet sich ebenfalls auf der CD.

6 Fazit

In der vorliegenden Arbeit konnte beim Vergleich der beiden Verfahren nicht gezeigt werden, dass ein Verfahren für alle Modelle und Untersuchungen dem anderen überlegen ist. Beide Verfahren unterliegen Einschränkungen bezüglich der Anwendbarkeit auf bestimmte Modelle oder bezüglich der Verfügbarkeit bestimmter Untersuchungsmethoden. Auch beim Vergleich der Laufzeit konnten für beide Verfahren Anwendungsfälle identifiziert werden, in denen sie jeweils besser geeignet sind. Bezüglich der Laufzeit wurden in Kapitel 5.2 Kriterien identifiziert, anhand derer ein geeignetes Verfahren ausgewählt werden kann.

Prinzipiell sind die beiden Verfahren einander sehr ähnlich. Sie unterscheiden sich hauptsächlich in zwei Aspekten: Die *Parametric Reachability Analysis* kann (bezüglich des Feuerns deterministischer und sofortiger Transitionen) indeterministische Modelle untersuchen und realisiert das *Model Checking* mittels Abtastung/Diskretisierung, während die *Region-based Analysis* auf deterministische Modelle beschränkt ist und das *Model Checking* geometrisch löst.

Diese Aspekte mit den jeweiligen Vor- und Nachteilen können getrennt voneinander betrachtet werden. Denn wie bereits in Kapitel 3.1.1 angemerkt, wäre eine geometrisches *Model Checking* für die *Parametric Reachability Analysis* umsetzbar, ebenso wie eine Abtastung/Diskretisierung für die *Region-based Analysis*.

Die Beschränkung auf deterministische Modelle schließt zwar die Untersuchung einiger Modelle aus, vereinfacht jedoch die geometrische Lösung des *Model Checking* und erlaubt die Verwendung des *until*-Operators. Die geometrische Lösung des *Model Checking* ist im Vergleich zur Abtastung/Diskretisierung exakt und in vielen Fällen schneller. Dafür kann bei der Abtastung/Diskretisierung die Genauigkeit frei gewählt werden, wodurch schnell grobe Annäherungen möglich sind. Außerdem kann sie in einigen Spezialfällen schneller sein, weil ihre Laufzeit anders von der Anzahl der Regionen abhängt.

Zukünftige Arbeiten könnten die Trennung der beiden oben genannten Aspekte durch eine Implementierung der geometrischen Lösung für die *Parametric Reachability Analysis* und der Abtastung/Diskretisierung für die *Region-based Analysis* realisieren. Eine weitere Möglichkeit ist die Optimierung des *Model Checking* für die *Region-based Analysis*. Eine Möglichkeit dafür wäre eine Implementierung des in Kapitel 5.2 skizzierten Verfahrens, bei dem für jede Region die jeweils bezüglich s nächste Region hinterlegt wird.

Abbildungsverzeichnis

Literaturverzeichnis

[1] Wasserversorgung Axenfels AG. Technischer Aufbau der Anlage. `http://www.wasserversorgung-axenfels.ch/cgi-bin/dokumente/aufbau.pdf`. Zugriff am 22. September 2016.

[2] Christel Baier, Boudewijn Haverkort, Holger Hermanns, and J-P Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Transactions on software engineering*, 29(6):524–541, 2003.

[3] Lucia Cloth and Boudewijn R Haverkort. Model checking for survivability! In *Second International Conference on the Quantitative Evaluation of Systems (QEST'05)*, pages 145–154. IEEE, 2005.

[4] René David. Modeling of hybrid systems using continuous and hybrid Petri nets. In *Petri Nets and Performance Models, 1997., Proceedings of the Seventh International Workshop on*, pages 47–58. IEEE, 1997.

[5] René David and Hassane Alla. On hybrid Petri nets. *Discrete Event Dynamic Systems*, 11(1-2):9–40, 2001.

[6] Bundesministerium des Innern. Nationale Strategie zum Schutz Kritischer Infrastrukturen (KRITIS-Strategie). `https://www.bmi.bund.de/SharedDocs/Downloads/DE/Broschueren/2009/kritis.pdf`, Juni 2009. Zugriff am 22. September 2016.

[7] Hamed Ghasemieh, Anne Remke, Boudewijn Haverkort, and Marco Gribaudo. Region-Based analysis of hybrid petri nets with a single general one-shot transition. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 139–154. Springer, 2012.

[8] Hamed Ghasemieh, Anne Remke, and Boudewijn R Haverkort. Survivability evaluation of fluid critical infrastructures using hybrid Petri nets. In *Dependable Computing (PRDC), 2013 IEEE 19th Pacific Rim International Symposium on*, pages 152–161. IEEE, 2013.

[9] Marco Gribaudo and Anne Remke. Hybrid Petri nets with a general one-shot transition. Reprint submitted to Performance Evaluation, 2015.