FAKULTÄT FÜR
INFORMATIK

Erweiterung von hybriden HnMRM durch Einfluss des kontinuierlichen Teils auf den Diskreten

Jens Schiborowski
Matrikelnummer 180788
Otto-von-Guericke-Universität Magdeburg
jens.schiborowski@st.ovgu.de

Erstgutachter:
Dr.-Ing Claudia Krull
Zweitgutachter:
Prof. Dr. rer. nat. habil. Stefan Schirra

Magdeburg, 24.06.2013

Zusammenfassung

Gegenstand dieser Arbeit ist die Weiterentwicklung hybrider Hidden non-Markovian Reward Models, so dass auch die Verhaltensrekonstruktion von Modellen möglich ist, in welchen der diskrete Systemzustand durch den kontinuierlichen Teil des Modells beeinflusst wird. Mithilfe dieser Erweiterung sollen den Hidden non-Markovian Reward Models weitere Anwendungsfelder eröffnet werden. Um die Umsetzung dieser Erweiterung und deren Leistungsfähigkeit beurteilen zu können, wurden zwei Modelle aus dem Bereich der erneuerbaren Energien erstellt, anhand denen die Korrektheit der Erweiterung gezeigt wurde. Die Ergebnisse der dafür durchgeführten Experimente zeigen, dass die Erweiterung die an sie gesetzten Ziele erreichen konnte und somit der Weg für eine Vielzahl an potentiellen Anwendungsfeldern für Hidden non-Markovian Reward Models geebnet wurde.

Inhaltsverzeichnis

1. Einleitung

1.1. Motivation und Hintergrund

Im Fachbereich der Simulation existiert eine Vielzahl an Aufgabenstellungen, die mit den Mitteln, die von einfachen Simulationsarten wie zum Beispiel diskreter Simulation geliefert werden, nicht zufriedenstellend gelöst werden können. Ein möglicher Lösungsansatz dafür ist die Verwendung hybrider Systeme, die die Möglichkeiten verschiedener Ansätze in sich vereinen. Im Rahmen dieser Arbeit werden ausschließlich hybride Systeme betrachtet, welche aus kontinuierlichen Systemen sowie aus diskreten, ereignisorierentierten Systemen bestehen, insbesondere denen, welche nur partiell beobachtbar sind. Diese Systeme beinhalten Elemente, bei welchen sich das Verhalten nicht vollständig beobachten lässt, was dazu führt, dass von diesen Elementen nur die Ausgabewerte bekannt sind. Für die partiell beobachtbaren Systeme werden die Hidden Markov Models als stochastisches Modell genutzt.

Diese besitzen ein breites Anwendungsspektrum, welches unter anderem Gesten- sowie Spracherkennung umfasst. In dieser Arbeit sollen jedoch Simulationsmodelle mit dem Hintergrund der erneuerbare Energien betrachtet werden. Die Bedeutung dieses Forschungsfeldes ist nicht zu vernachlässigen und nimmt zudem auch stetig zu. Dies ist unter anderem geschuldet durch die stetig steigenden Kosten fossiler Brennstoffe und dem in letzter Zeit immer mehr in die den Fokus der Öffentlichkeit gerückten Gefahrenpotential der konservativen Arten der Energieproduktion.

Die Modelle, welche in dieser Arbeit umgesetzt werden, befassen sich mit der Nachbildung des dynamischen Verhaltens zwischen Energieverbraucher und Energieversorger sowohl aus Versorger- als aus Verbrauchersicht. Diese Modelle beinhalten sowohl kontinuierliche als auch diskrete Systembestandteile, wobei jeweils Teile des Modells nur partiell beobachtbar sind. Bei dem Modell aus Versorgersicht fällt das Kundenverhalten in diese Kategorie und bei dem Modell aus Verbrauchersicht kann der Verlauf des Energieverbrauchs nur zu bestimmten Zeitpunkten abgefragt werden. Zur Umsetzung dieser Modelle wird eine Erweiterung der HMM, die Hidden non-Markovian Reward Models genutzt. Entwickelt von Krull und Horton[1], bieten diese die Möglichkeit, bei partiell beobachtbaren Hybridsystemen, in denen das diskrete Verhalten stochastisch und nicht

beobachtbar ist und für die für die kontinuierlichen Variablen Samples verfügbar sind, Verhaltensrekonstruktionen durchzuführen. Da diese Technik erst vor kurzer Zeit vorgestellt wurde, ist die Frage nach den Anwendungsfeldern noch größtenteils offen. Daher soll unter anderem in dieser Arbeit ermittelt werden, ob die ausgewählten Modelle zufriedenstellend mit HnMRM abgebildet und ihr Verhalten rekonstruiert werden kann.

Weiterhin bieten die HnMRM keine Möglichkeit, das Verhalten von Systemen zu rekonstruieren, in denen der diskrete Systemzustand durch das kontinuierliche Verhalten beeinflusst werden kann. Da Problemstellungen, die ein solches Verhalten aufweisen, durchaus in dem gewählten Anwendungsfeld zu finden sind, ist die Entwicklung einer Erweiterung, welche die Verhaltensrekonstruktion solcher Modelle ermöglicht, das Ziel dieser Arbeit.

1.2. Ziele

In diesem Kapitel werden die Ziele der Arbeit vorgestellt. Diese dienen dazu den Rahmen dieser Arbeit abzustecken und später der Bewertung der Ergebnisse der Arbeit eine fundierte Grundlage zu geben.

Das Hauptziel dieser Arbeit ist, wie der Titel bereits zeigt, folgendes:

- Mit Hilfe der entwickelten Erweiterung können Hidden non-Markovian Reward Models den Einfluss des kontinuierlichen Verhaltens auf den diskreten Zustand korrekt abbilden

Da diese Formulierung doch recht wage ist und Raum für Interpretationen lässt, wurden folgende Teilziele formuliert:

1. Das Verhalten von einfachen Modellen mit weniger als sieben Zuständen kann rekonstruiert werden
2. Die Rechenzeit ist im Vergleich zu Hidden non-Markovian Reward Models nicht mehr als 20% höher

Teilziel 1 dient zur Eingrenzung der zu überprüfenden Anwendungsfälle, so dass die Schlussfolgerung gezogen werden kann, dass wenn die neu entwickelte Erweiterung für kleine Beispielmodelle korrekt arbeitet, sie auch für größere Modelle korrekt funktioniert. Teilziel 2 dient dazu, die Anwendbarkeit der Erweiterung zu gewährleisten, da das Grundgerüst, welches als Basis für die Erweiterung genutzt wird, performant arbeitet und dies nicht signifikant geändert werden soll.

1.3. Struktur der Arbeit

Nachdem im vorherigen Abschnitt die Ziele vorgestellt wurden, wird in diesem Abschnitt nun auf die Arbeitsschritte eingegangen, welche nötig sind, um die Ziele zu erreichen. Dazu werden im nächsten Kapitel zunächst die theoretischen Grundlagen der genutzten Techniken, namentlich Hidden non-Markovian Models, Reward Modeling sowie abschließend der Hidden non-Markovian Reward Models gegeben.

Nachdem dies geschehen ist, folgt die Vorstellung und Erläuterung der entwickelten Erweiterung. Diese umfasst neben der Formalisierung, welche anhand eines kleinen Beispiels näher verdeutlicht wird, auch die Implementierung der theoretischen Grundlagen.

Wenn dies gezeigt wurde, wird der Fokus auf die Modelle gelegt, deren Verhalten rekonstruiert werden soll. Dazu werden in dem folgenden Kapitel zunächst die Experimente beschrieben, welche zur Erfüllung der Ziele von Nöten sind. Anschließend wird für jedes Modell gesondert zunächst eine kurze inhaltliche Beschreibung gegeben, welche von der Erläuterung, wie die nötigen Testdaten generiert werden, gefolgt wird. Abschließend für dieses Kapitel werden die durchgeführten Experimente ausgewertet und die Ergebnisse hinsichtlich der gesetzten Ziele analysiert.

Als Abschluss werden noch einmal alle Erkenntnisse dieser Arbeit zusammengefasst und ein finales Fazit gezogen. Zudem erfolgt ein Ausblick auf mögliche Erweiterungen der Hidden non-Markovian Reward Models.

2. Grundlagen

In diesem Kapitel wird auf die Themen eingegangen, welche das Verständnis der folgenden Kapitel ermöglichen soll. Zudem wird in diesem Zusammenhang auch der aktuelle Stand der Technik dargestellt und dem Thema dieser Arbeit somit ein Kontext gegeben.

2.1. Hidden non-Markovian Models

Hidden non-Markovian Models sind ein Modellierungsparadigma, welches die Modellierung und Analyse verborgener Systeme ermöglicht [2]. Sie basieren auf den Hidden Markov Models [3], welche als verborgene Modelle ausschließlich discrete-time Markov chains [4] verwenden können. Im Gegensatz dazu ist es mit Hidden non-Markovian Models möglich, als verborgene Modelle beliebige diskrete stochastische Modelle zu verwenden. Dies hat den Vorteil, dass sich die Wahl der Zustandsübergangsraten nicht mehr ausschließlich auf zeitunabhängige Exponentialverteilungen beschränkt, sondern dass beliebige Verteilungen wie unter anderem Normal- oder Weibullverteilungen verwendet werden können. Formal erfolgt diese Erweiterung durch das Hinzufügen von Zeitabhängigkeit, stochastischen Feuerzeiten von Transitionen sowie dem Hinzufügen von Zeitstempeln zu Symbolemissionen [1].

Um den Zustandsraum eines Hidden non-Markovian Models zu repräsentieren, werden hier stochastische Petrinetze genutzt, da die Zustände eines stochastischen Petrinetzes in direkter Beziehung zu den diskreten Zuständen des Modells stehen [2]. Da die Anzahl der möglichen Definitionen eines stochastischen Petrinetzes recht vielfältig ist, wird hier auf „augmented stochastic Petri nets“ (ASPN) [5] zurückgegriffen. Nach deren Definition setzt sich ein solches stochastisches Petrinetz aus drei grundlegenden Elementen zusammen: Zustand, Transitionen sowie Kanten. Zustände sowie Transitionen sind die zwei Arten von Knoten aus denen ein Petrinetz aufgebaut ist, welche durch Kanten so miteinander verbunden sind, dass jeweils ein Zustand mit einer Transition oder umgekehrt verbunden ist. Die Transitionen, welche schalten können, sobald sich in den vorherigen Zuständen Marken befinden, können zusätzlich mit einer Verzögerung belegt sein, welche das Schalten erst nach einer bestimmten Zeit ermöglicht. Außerdem können Transitionen noch die race-age-Eigenschaft aufweisen, welche besagt, dass das Alter einer Transition

auch dann erhöht wird, wenn diese im aktuellen Zustand nicht aktiv ist.

Die Probleme, welche sich mit Hidden non-Markovian Models lösen lassen, können in drei allgemeine Kategorien aufgeteilt werden [2]. Die erste Kategorie ist die Evaluation, bei der anhand eines vollständigen Modells sowie einer gegebenen Spur die Wahrscheinlichkeit, mit der das Modell die gegebene Spur erzeugen kann, berechnet werden kann. Um dies zu ermöglichen, existieren zwei verschiedene Lösungsalgorithmen. Zum einen der Forward-Algorithmus, in welchem iterativ die Wahrscheinlichkeit der gegebenen Spur berechnet wird, sowie zum anderen der Backward-Algorithmus, in welchem die Gesamtwahrscheinlichkeit der Spur direkt berechnet wird. Die zweite Kategorie ist die Decodierung, welche ebenfalls ein vollständiges Modells sowie eine Spur benötigt. Hier ist jedoch die Aufgabe, den Pfad zu finden, welcher mit der höchsten Wahrscheinlichkeit die Spur erzeugt hat. Der für diese Fragestellung am häufigsten genutzte Algorithmus ist der Viterbi-Algorithmus. Die dritte Kategorie ist das Training, bei dem anhand einer Ausgabesequenz und einem bereits parametrisierten Modell die optimale Parametrisierung des Modells hinsichtlich der Erzeugung der Ausgabespur gefunden werden soll. Der für die Beantwortung dieser Fragestellung bevorzugte Algorithmus ist der Baum-Welch Algorithmus, welcher jedoch nicht für Hidden non-Markovian Models anwendbar ist. Deshalb findet die dritte Kategorie im folgenden Verlauf dieser Arbeit keine weitere Beachtung.

2.2. Reward Models

Wenn nun unter anderem in stochastischen Petrinetzen kontinuierliche Werte dargestellt werden sollen, können dafür sogenannte Reward Models genutzt werden [6, 7]. Reward Models erlauben es, einem stochastischen Prozess, wie in diesem Fall einem stochastischen Petrinetz, eine Struktur für diese kontinuierlichen Werte, auch Reward-Werte genannt, hinzuzufügen. Aufteilen lassen sich die kontinuierlichen Werte, genannt Rewards, in zwei Hauptkategorien, die Rate Rewards sowie die Impulse Rewards. Rate Rewards verändern reellwertige Variablen anhand einer gewöhnlichen Differentialgleichung, welche abhängig vom aktuellen Systemzustand ist. Ein Beispiel für ein Rate Reward wäre unter anderem der Stromverbrauch eines Fernsehers im Standby-Betrieb. Impulse Rewards hingegen verändern reellwertige Variablen, sobald durch das Schalten einer Transition ein Zustandswechsel ausgelöst wird. Beispiele für Impulse Rewards sind die Kosten einer Wartung oder der Profit beim Verkauf von Gütern.

2.3. Hidden non-Markovian Reward Models

Um nun die Verhaltensrekonstruktion partiell beobachtbarer Systeme, ermöglicht durch Hidden non-Markovian Models, mit der Verwendung von Reward-Werten zu kombinieren, wurden die Hidden non-Markovian Reward Models entwickelt. Formal lässt sich das mathematische Modell, welches den HnMRM zu Grunde liegt, als das folgende 6-Tupel $HnMRM = (S, TR, A, \Pi, \vec{\rho}, rr)$ beschreiben [1]. Das Tupel beinhaltet die folgenden Elemente:

- S - Menge von N diskreten Systemzuständen
- TR - Menge von Transitionen mit $TR_i = (dist, ir, aging)$, wobei $dist$ einer kontinuierlichen Wahrscheinlichkeitsdichtefunktion, ir der Änderung von $\vec{\rho}$ durch Impulse Rewards und $aging$ den Zustand der race-age-Eigenschaft beschreibt
- $A = \{a_{ij}\}$ - komplettes Zustandstransitionsverhalten und a_{ij} = Menge von Transitionen, die einen Zustandswechsel von S_i nach S_j auslösen können
- $\Pi = (\pi_1, \ldots, \pi_N)$ - initialer Wahrscheinlichkeitsvektor der diskreten Zustände und π_i = Wahrscheinlichkeit des Systems zum Zeitpunkt 0 in Zustand S_i zu sein
- $\vec{\rho} = (\rho_1, \ldots, \rho_m)$ Vektor aller kontinuierlichen Reward Values
- Rate Rewards definiert durch $rr(s, \vec{\rho})$; beschreibt Veränderungen in $\vec{\rho}$ durch Rate Rewards mit Hilfe von Differentialgleichung $\frac{d\vec{\rho}}{dt}$ abhängig vom Systemzustand und aktuellem Reward Value
- Spur O - Sequenz von Messungen von einer oder mehreren Reward Values $\vec{\rho}$ mit Zeitstempeln e_t
- Pfad Q - Sequenz von Systemzuständen s_t mit Zeitstempeln e_t

Mit Hilfe dieser Elemente ist es nun möglich ein partiell beobachtbares Modell mit Reward-Werten zu beschreiben und zu analysieren. Wie die bereits in Kapitel 2.1 genannten Fragestellungen für die Verwendung von Hidden non-Markovian Reward Models beantwortet werden können, wird im nächsten Abschnitt erläutert.

2.4. Proxels

Um die in Kapitel 2.1 vorgestellten Fragenstellungen beantworten zu können, wird das Konzept der Proxel [8] genutzt. Ein Proxel ist ein Wahrscheinlichkeitselement, welches

alle relevanten Informationen zur Berechnung der zukünftigen Entwicklung eines Systems beinhaltet. Angewandt auf Hidden non-Markovian Reward Models bedeutet dies, dass ein Proxel den aktuellen Systemzustand, das Alter aller aktiven oder race-age-Transitionen, eine Repräsentation des Pfades, der zu diesem Proxel geführt hat, die aktuellen Reward-Werte sowie die Wahrscheinlichkeit dieser Kombination aus Eigenschaften beinhaltet.

Der verwendete Algorithmus arbeitet wie folgt: Als Eingabewerte erhält der Algorithmus eine Spur, anhand derer er die diese Spur erzeugenden Pfade berechnen soll. Unter Verwendung diskreter Zeitschritte werden für jeden Zeitschritt alle möglichen Pfade berechnet. Somit ist das Grundprinzip dieses Algorithmus eine Breitensuche. Da aber nicht alle berechneten Pfade die gewünschte Spur erzeugen können, werden nur die validen Pfade übernommen und weiter bearbeitet. Ein Pfad ist dann valid, wenn die absolute Differenz zwischen den berechneten Reward-Werten und den dazugehörigen Samplewerten aus der Spur kleiner als ein festgelegter Parameter ϵ ist, welcher die akzeptable Differenz zwischen Samplewert und Reward-Wert festlegt. Sollte die Differenz größer sein, so wird der Proxel entfernt. Wenn nun alle Samplewerte der Spur bearbeitet wurden, liefert dieser Algorithmus alle validen Pfade, welche diesen Werteverlauf entsprechen.

Nachdem nun die für das nötige Verständnis benötigten Informationen gegeben wurden, ist die Grundlage für das weitere Vorgehen gelegt. Wie bereits erwähnt, ist es mit den bisherigen Mitteln möglich, das Verhalten von partiell beobachtbaren Systemen zu rekonstruieren. Dies funktioniert jedoch nicht mehr, sobald in den Systemen Verhalten vorkommt, in welchem der diskrete Systemzustand durch die kontinuierlichen Werte beeinflusst wird. Wie dieses Problem gelöst wurde, wird im nächsten Kapitel dargestellt.

3. Erweiterung HnMRM

In diesem Kapitel wird nun die Weiterentwicklung erläutert, welche es ermöglicht, mit Hilfe von Hidden non-Markovian Reward Models das Verhalten von System zu rekonstruieren, in welchen der diskrete Systemzustand von den kontinuierlichen Variablen beeinflusst wird. Dazu wird zunächst die theoretische Grundlage unter Verwendung der formalen Beschreibung erläutert. Zudem wird ein Beispiel gegeben, um die Vorgehensweise deutlich darzulegen. Im zweiten Abschnitt dieses Kapitels wird dann die Implementierung der Erweiterung vorgestellt.

3.1. Formalisierung

Um aufzuzeigen, wie die geplante Erweiterung funktionieren kann, wird zunächst verdeutlicht, welche Anforderungen an diese gestellt werden und wie diese anschließend umgesetzt wurden. Um das gesetzte Ziel umzusetzen, muss eine Möglichkeit gefunden werden, wie anhand der jeweils aktuellen Reward-Werte, welche den kontinuierlichen Teil des Systems darstellen, der aktuelle Systemzustand beeinflusst werden kann. Da für die Darstellung der internen Modelle der Hidden non-Markovian Reward Models stochastische Petrinetze genutzt werden, lässt sich die Beeinflussung des diskreten Systemzustandes als das Schalten beziehungsweise Nicht-Schalten von Transitionen darstellen. In den Vorgang des Schaltens einer Transition müssen nun also noch die Reward-Werte integriert werden, um diesen den gewünschten Einfluss zuzuschreiben. Die gewählte Möglichkeit dies zu verwirklichen ist es, den Transitionen die Funktionalität hinzuzufügen, dass vor dem Schalten einer Transition die aktuellen Reward-Werte des Systems gegen einen festgelegten Wert verglichen werden. Dann kann abhängig von dem Ergebnis des Vergleichs das Schalten der Transition erlaubt oder ausgesetzt werden. Ein alternativer Lösungsansatz war es, dass sobald ein bestimmter Grenzwert erreicht, beziehungsweise über- oder unterschritten wird, sofort eine Transition schaltet. Dieser Ansatz wurde jedoch verworfen, da dafür zeitlose Transitionen, also Transitionen, welche sofort und ohne zugehörige Verteilung schalten können, nötig sind, welche in der Definition der Hidden non-Markovian Reward Models nicht enthalten sind. Um nun den gewählten Ansatz umzusetzen, ist eine Modifikation der Transitionsdefinition notwendig. Hierzu wird die

ursprüngliche Definition der Transition (Kapitel 2.3) um einen M-elementigen Vektor erweitert, wobei M die Anzahl der verschiedenen Reward Values ist. Jedes Element dieses Vektors beinhaltet wiederum ein Paar aus Vergleichsoperation und Vergleichswert. Diese Kombination wurde gewählt, um bei dem Vergleich gegen mehrere Reward Values flexibler zu sein, da man so nicht auf einen Vergleichsoperator beschränkt wird. Aus dieser Beschreibung ergibt sich folgende formale Definition der erweiterten Transition.

- TR - Menge von Transitionen mit $TR_i = (dist, ir, aging, hasGuard, guard)$, wobei hasGuard beschreibt, ob die Transition eine Guard-Funktion besitzt und guard ein M-elementiger Vektor mit $guard_i = \{compOperator_i, guardValue_i\}$ ist, mit $compOperator_i$ der Vergleichsoperator zugehörig zu dem i-ten Reward-Wert und $guardValue_i$ der Vergleichswert zugehörig zu dem i-ten Reward-Wert

Für die Repräsentation von HnMRM als stochastisches Petrinetz wird die Erweiterung der Transition um den beschriebenen Vektor nach dem Prinzip einer Guard-Funktion behandelt. Im dem Kontext eines Petrinetzes wird eine Guard-Funktion durch eine Bedingung, welche an eine Transition gebunden ist und das Schalten dieser Transition bei Erfüllen der Bedingung ermöglicht, implementiert [9]. Die folgende Verfahrensweise weicht jedoch von der ursprünglichen Definition einer Guard-Funktion ab. Dies wird darin begründet, dass die gewählte Verfahrensweise in dem Simulationstool Anylogic Anwendung findet, welches für die Erzeugung von Testdaten genutzt wird und damit als Referenz genutzt wird. Da bei den hier verwendeten Petrinetzen die Transitionen mit einer Verzögerung belegt sind, wird, sobald die Verzögerung abgelaufen ist und die Transition somit schalten würde, überprüft, ob die gesetzten Bedingungen erfüllt sind. Falls dies der Fall ist, schaltet die Transition wie geplant. Sollten die Bedingungen jedoch nicht erfüllt werden, so darf die Transition nicht schalten und die Verzögerung wird anhand ihrer Verteilungsfunktion zurückgesetzt.

Die Funktionsweise der Erweiterung wird nun anhand eines kurzen Beispiels näher erläutert. Das Beispiel, welches hierfür gewählt wurde, ist eines der Modelle, die für die Durchführung der Experimente ausgewählt wurden. In diesem wird das dynamische Verhalten zwischen Energieverbraucher und Energielieferant aus Sicht des Verbrauchers vereinfacht nachgebildet.

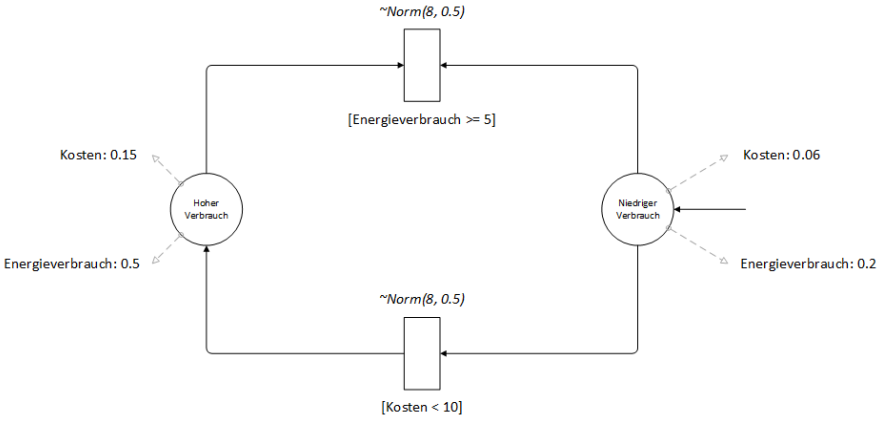

Abbildung 3.1.: Petrinetz Modell 1

Wie in Grafik 3.1 zu erkennen, befindest sich der Verbraucher entweder im Zustand *Hoher Verbrauch* oder *Niedriger Verbrauch* und als Reward-Werte werden der *Energieverbrauch* und die damit zusammenhängenden *Kosten* genutzt. Der Wechsel zwischen den beiden Zuständen erfolgt grundsätzlich in Intervallen von $\sim Norm(8,\ 0.5)h$, ist aber zusätzlich noch an Bedingungen geknüpft: Der Zustandswechsel von *Hoher Verbrauch* zu *Niedriger Verbrauch* kann nur erfolgen, wenn die aktuell verbrauchte Energie über einem Mindestenergieverbrauch, welcher in diesem Fall auf 5 kWh festgelegt wurde, liegt. Der Wechsel von *Niedriger Verbrauch* zu *Hoher Verbrauch* kann nur erfolgen, wenn die aktuellen Kosten eine Kostenobergrenze von in diesem Beispiel 10 € nicht überschreiten. Der Energieverbrauch wird abhängig von dem Zustand bestimmt und ist im Zustand *Hoher Verbrauch* 0.5 kW pro Stunde sowie bei *Niedriger Verbrauch* 0.2 kW pro Stunde. Die Kosten sind ebenfalls abhängig von dem aktuellen Verbrauchsverhalten und werden als 0.3 € * aktueller Energieverbrauch pro Stunde berechnet, woraus sich folgende Werte ergeben: Im Zustand *Hoher Verbrauch* betragen die Kosten 0.15 € pro Stunde und im zweiten Zustand 0.06 € pro Stunde. Es erfolgt zudem alle 24 Stunden ein Reset, bei dem die Reward-Werte wieder auf 0 gesetzt werden. Zu Beginn sind sowohl der Energieverbrauch als auch die Kosten 0 und der Startzustand ist der hohe Verbrauch.

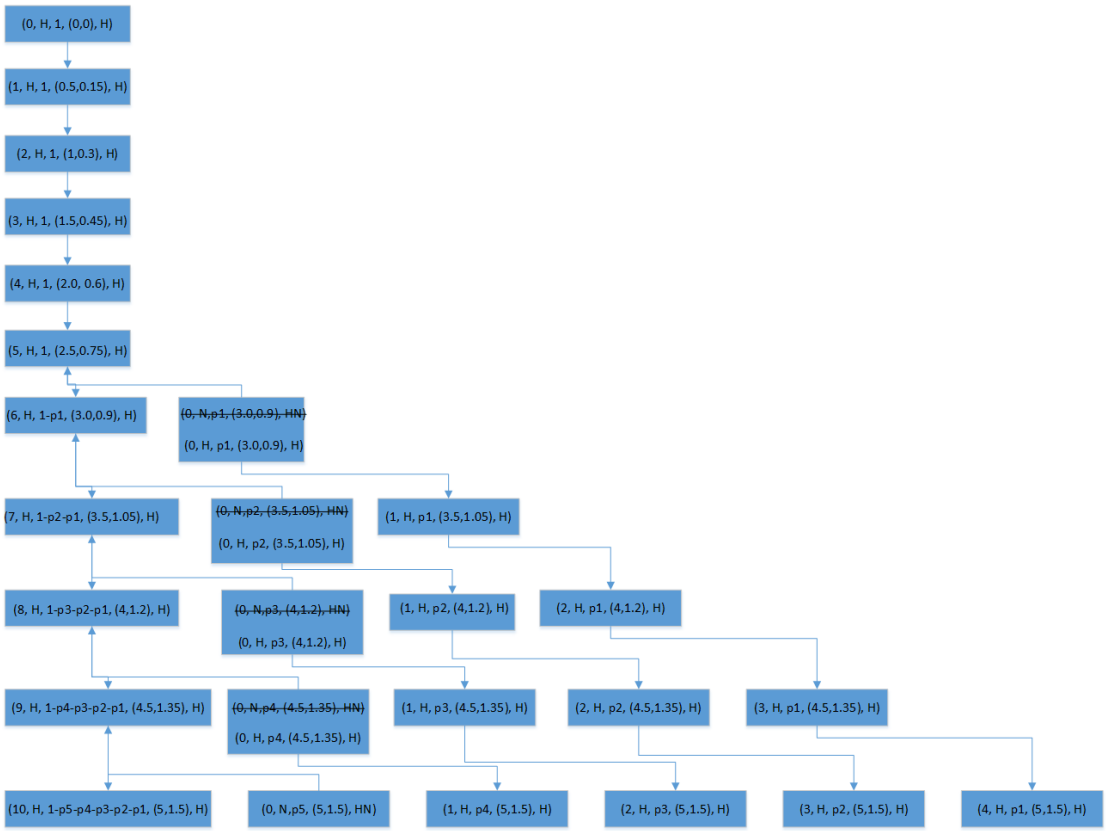

Abbildung 3.2.: Zustandsraum des Beispielmodells

Die Grafik 3.2 zeigt die Entwicklung des Zustandsraums während der ersten zehn Zeitschritte. In jedem Zeitschritt werden die im aktuellen Zeitpunkt behandelten Proxel dargestellt. Der Inhalt der Proxel, dargestellt durch das Tupel, wurde zu Beispielzwecken auf ein Minimum reduziert. Enthalten sind nur noch in folgender Reihenfolge: Das Alter der aktiven Transition, der aktuelle diskrete Systemzustand, wobei H für *Hoher Verbrauch* und N für *Niedriger Verbrauch* steht, die aktuelle Pfadwahrscheinlichkeit, der aktuelle Stand der kontinuierlichen Variablen sowie der bisherige Pfad. Wie zu erkennen ist, wird in den ersten fünf Zeitschritten kein Zustandswechsel ausgelöst. Erst ab dem nächsten Zeitschritt versucht die Transition zu schalten. Dies misslingt jedoch, da der aktuelle Wert des Energieverbrauchs noch unter dem Vergleichswert liegt. Deshalb entsteht ein neuer Proxel, bei welchem das Alter der Transition auf 0 zurückgesetzt wurde. Dieses Verfahren wird die nächsten Zeitschritte wiederholt, bis schlussendlich im zehnten Zeitschritt die Transition schalten darf und ein Zustandswechsel ausgelöst wird. Die Repräsentation des Zustandsraumes für die gewählten Zeitschritte zeigt damit nun das zuvor beschriebene Verhalten. Sollte bei einem Zustandswechsel die Bedingung der

Transition nicht erfüllt werden, ändert sich der aktuelle Zustand nicht, aber das Alter der Transition wird zurückgesetzt.

3.2. Implementierung

In diesem Abschnitt wird nun beschrieben, wie die im vorherigen Abschnitt beschriebene Erweiterung umgesetzt und implementiert wurde. Als Basis für die Implementierung wurde der Quellcode verwendet, der in [1] Verwendung fand. Ebenso wie die Basis wurde die Erweiterung in C++ geschrieben und unter Verwendung der GNU Compiler Collection, kurz GCC, kompiliert. Um den gewählten M-elementigen Vektor, welcher die Vergleichswerte und die Vergleichsoperatoren beinhaltet, zu implementieren, wurde die Definition der Transition um drei zusätzliche Variablen erweitert. Zunächst wird eine Boolean-Variable benötigt, welche besagt, ob die Transition eine zu überprüfende Bedingung enthält oder nicht. Zudem werden zwei Vector-Variablen genutzt, welche die Schwellwerte als Double-Werte und die dazu gehörigen Vergleichsoperatoren als Integer-Werte beinhalten. Um die entwickelte Erweiterung vollständig umzusetzen, ist zudem eine Anpassung des Lösungsalgorithmus von Nöten. Das folgende Vorgehen findet an den Stellen Anwendung, an denen Proxel hinzugefügt werden würden, welche durch einen Zustandswechsel entstehen. Der intuitive erste Schritt ist die Überprüfung, ob die Boolean-Variable hasGuard gesetzt ist. Sollte dies nicht der Fall sein, so wird ohne weitere Bearbeitungsschritte der Proxel für den Zustandswechsel hinzugefügt. Sollte die Transition jedoch eine Guard-Funktion aufweisen, erfolgt der Vergleich der aktuellen Reward-Werte mit den gesetzten Schwellwerten unter Berücksichtigung der gewählten Vergleichsoperationen. Hierbei erfolgt eine UND-Verknüpfung der jeweiligen Vergleichsergebnisse, um die Aussage treffen zu können, ob die Guard-Funktion erfüllt wurde. Sollte dies eintreten, so wird ebenfalls wie bei dem Fall, dass keine Guard-Funktion vorhanden ist fortgefahren. Sollte die Guard-Funktion nicht erfüllt sein, erfolgt neben dem Zurücksetzen des Alters dieser Transition das Hinzufügen eines Proxels, welcher abgesehen von dem aktualisierten Alter und den aktuellen Pfad bis zu diesem Zustand die gleichen Elemente beinhaltet wie die bei den anderen Möglichkeiten hinzugefügten Proxel. Der folgende Pseudocode zeigt nun das beschriebene Vorgehen, welches immer dann zum Tragen kommt, sollte ein Proxel hinzugefügt werden, welcher durch einen Zustandswechsel erzeugt wird.

Algorithm 1 Überprüfung der Guard-Funktion

```
if Transition ohne Guard-Funktion then
    Proxel mit Zustandswechsel hinzufügen
else
    Aktuelle Reward-Werte mit Vergleichswerten anhand der Vergleichsoperatoren vergleichen
    if Vergleich erfolgreich then
        Proxel mit Zustandswechsel hinzufügen
    else
        Alter der aktiven Transition = 0
        Proxel ohne Zustandswechsel hinzufügen
    end if
end if
```

Zusätzlich zu den bereits beschriebenen Änderungen wurde der Basisquellcode dahingehend erweitert, so dass die Anzahl der Reward-Werte nicht mehr nur auf einen Reward-Wert beschränkt ist. Dies geschah durch das Ersetzen der Double-Variablen durch Vector-Variablen, wie sie auch schon bei der Erweiterung der Transition genutzt wurden. Zusätzlich mussten nur die Ausgabeoperationen sowie die Vergleichsoperationen, welche die Reward-Werte nutzten, minimal angepasst werden. Die Erweiterung auf eine beliebige Anzahl Reward-Werte geschah, zum einen um ein breiteres Anwendungsspektrum zu schaffen und zum andern weil die gewählten Modelle von Beginn an mit mehreren Reward-Werten geplant waren. Mit allen durchgeführten Änderungen ist es nun möglich, das Verhalten von Systemen zu rekonstruieren, in welchen der diskrete Systemzustand durch Berücksichtigung der kontinuierlichen Werte beeinflusst werden kann. Die Korrektheit dieser Erweiterung wird nun in den nächsten Kapiteln anhand von zwei Beispielen gezeigt werden.

4. Experimente

In diesem Kapitel wird unter Einbeziehung der gesetzten Ziele die Planung der nötigen Experimente beschrieben. Dazu werden zunächst noch einmal die Ziele genannt:

1. Das Verhalten von einfachen Modellen mit weniger als sieben Zuständen kann rekonstruiert werden
2. Die Rechenzeit ist im Vergleich zu Hidden non-Markovian Reward Models nicht mehr als 20% höher

Um das erste der beiden Ziele zu erreichen, muss gezeigt werden, dass mindestens eine der Fragestellungen aus Kapitel 2.1 zufriedenstellend beantwortet werden kann. Hierbei wird sich aber wie bereits erwähnt auf die ersten beiden Fragestellungen beschränkt. Dazu wird eine Spur als Eingabe für den Lösungsalgorithmus der Hidden non-Markovian Reward Models benötigt. Wie diese Spur erzeugt wird, wird im nächsten Abschnitt näher erläutert. Der Lösungsalgorithmus liefert nach der Eingabe der Spur alle Pfade, welche diese Spur mit der maximalen Abweichung eines festgelegten Parameters ϵ erzeugen können. Diese Pfade beinhalten neben der Pfadwahrscheinlichkeit, welche zur Beantwortung der Fragestellung Evaluation benötigt wird, auch die Sequenz der Systemzustände mit den Zeitstempeln, welche angeben, zu welchem Zeitpunkt in den jeweiligen Zustand gewechselt wurde. Dies erlaubt die Beantwortung der zweiten Fragestellung, der Decodierung. Damit lassen sich nun mit Hilfe der Ausgabe des Lösungsalgorithmus Rückschlüsse auf die Erfüllung des ersten Zieles schließen. Um das verbleibende Ziel zu erreichen, wird die Laufzeit des Lösungsalgorithmus mit den Laufzeiten verglichen, welche in [1] angegeben sind. Das Mittel für die Beeinflussung der Laufzeit des Lösungsalgorithmus ist die Variation des ϵ Parameters. Je größer der Parameterwert, desto mehr Proxel werden erzeugt und somit steigt auch die Laufzeit an.

4.1. Modell 1

In diesem Abschnitt wird auf das erste der zwei ausgewählten Modelle eingegangen. Dazu erfolgt zunächst die Beschreibung des Sachverhalts, welcher mit diesem Modell abgebildet werden soll. Anschließend wird beschrieben, welche Testdaten für die Durchführung

der Experimente benötigt werden und in welchem Umfang diese bereitgestellt werden müssen. Abschließend erfolgt die Vorstellung der aus den Experimenten gewonnen Ergebnisse, die Präsentation der Schlussfolgerungen, welche sich aus diesen Ergebnisse ziehen lassen sowie Auswirkung der Schlussfolgerungen auf die gesetzten Ziele.

4.1.1. Modellbeschreibung

In dem ersten Modell wird das dynamische Verhalten zwischen Energieproduzent und Energieverbraucher aus der Sicht eines Verbrauchers nachgebildet. Als Zeiteinheit wurden Stunden gewählt Der Verbraucher kann hierbei entweder einen hohen Verbrauch oder einen niedrigen Verbrauch aufweisen. Als die kontinuierlichen Variablen werden in diesem Beispiel der Energieverbrauch sowie die daraus resultierenden Kosten betrachtet. Der Wechsel von niedrigem Verbrauch auf hohen Verbrauch findet mit einer Verteilung von $\sim Norm(8,\ 0.5)h$ statt, ist aber mit der Bedingung belegt, dass die Kosten unter einer festgelegten Kostengrenze von in diesem Fall 10 € pro Tag liegen. Der Wechsel von hohem Verbrauch zu niedrigem Verbrauch unterliegt derselben Verteilung, ist aber mit einer anderen Bedingung belegt. Für diesen Zustandswechsel muss der Energieverbrauch über einem Mindestenergieverbrauch von 5kWh pro Tag liegen. Da für die Bedingungen die Werte pro Tag genutzt werden, werden die kontinuierlichen Variablen regelmäßig alle 24 Stunden zurückgesetzt. Dieser gesamte Sachverhalt wird noch einmal in Grafik 4.1 veranschaulicht.

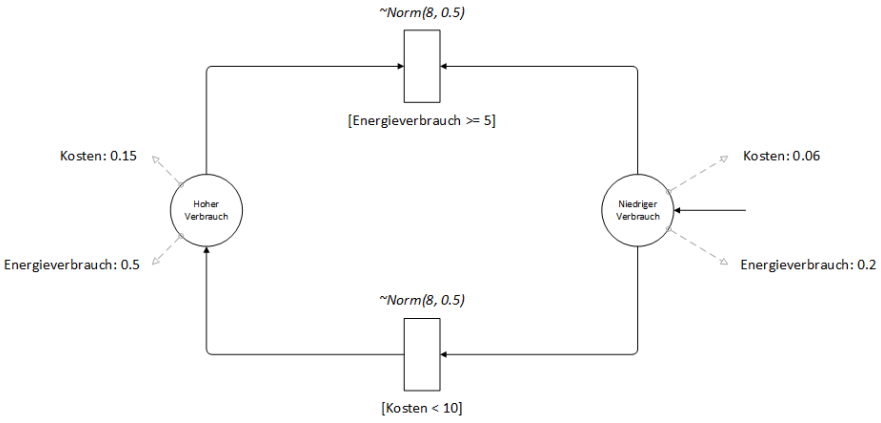

Abbildung 4.1.: Petrinetz Modell 1

Mögliche Fragestellungen, welche sich mit einem solchem Modell beantworten lassen könnten, ist die Zuordnung von gegebenen Kundendaten anhand des verbrauchten Stroms

in ein vorgegebenes Verbraucherprofil wie zum Beispiel Privat- oder Geschäftskunde. Diese Zuordnung kann genutzt werden, um zum Beispiel dem Kunden spezielle Konditionen anzubieten oder die Angaben des Kunden hinsichtlich seines Verbrauchsverhaltens zu überprüfen.

4.1.2. Testdatengenierung

Um die im bereits erwähnte Spur zu erzeugen, wird auf das Simulationstool Anylogic [10] zurückgegriffen. Hierzu wurde ein Simulationsmodell des beschriebenen Sachverhalts erstellt. Dieses wird in der folgenden Grafik 4.2 gezeigt.

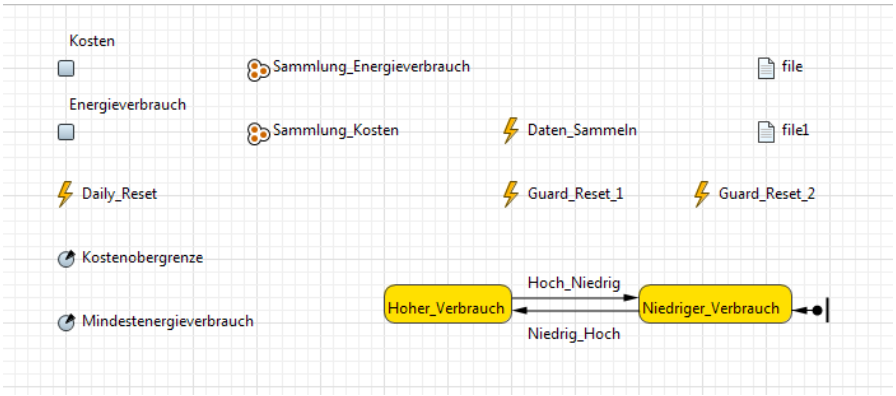

Abbildung 4.2.: Anylogic-Implementierung Modell 1

Die Zustände *Hoher Verbrauch* und *Niedriger Verbrauch* sowie die Transitionen *Hoch_Niedrig* und *Niedrig_Hoch* mit ihren Schaltzeiten entsprechen den Spezifikationen aus Grafik 4.1 und die kontinuierlichen Werte werden als Stock-Variablen mit ihren zugehörigen einfachen Differentialgleichungen abgebildet. Die Schaltbedingungen der Transitionen wurden unter Verwendung der Anylogic eigenen Guard-Funktion für Transitionen implementiert. Da diese bei Nichterfüllung der Bedingung die Transition deaktivieren, werden die Transitionen ereignisgesteuert jedes Mal reaktiviert, falls dieser Fall eintritt. Die Erfassung der Samplewerte erfolgte nach einer Verteilung von $\sim Norm(2, 0.5)h$, wobei die Laufzeit der Simulationen auf eine Laufzeit von 200 Stunden in Modellzeit beschränkt wurde. Damit beläuft sich die durchschnittliche Größe einer unter Verwendung dieser Simulation erzeugten Spur auf 90 bis 100 Samplewerte. Die folgende Tabelle 4.1.2 zeigt einen kleinen Ausschnitt aus einer auf diese Art erzeugten Spur.

Zeit	Energieverbrauch	Kosten
1,99	0,40	0,12
4,42	0,88	0,27
6,43	1,29	0,39
8,41	1,76	0,53
10,71	2,91	0,87
12,75	3,93	1,18
14,65	4,88	1,46
17,77	5,87	1,76
19,15	6,14	1,84

Tabelle 4.1.: Ausschnitt aus der verwendeten Spur

Zudem erfolgt die Speicherung der Zeitpunkte der Zustandsübergänge mit den dazugehörigen Zielzuständen. Dies geschieht, um der Auswertung der Experimente noch eine zusätzliche Auswertungsmöglichkeit hinzuzufügen, falls die bisher ermittelten Ergebnisse nicht zufriedenstellend ausfallen sollten.

4.1.3. Auswertung

Um bei der Ausführung des Lösungsalgorithmus valide Pfade zu erzeugen, erfolgte zunächst ein Parametertuning des ϵ-Parameters. Hierbei wurden nach mehreren Tests das Wertepaar (0.4, 0.2) gewählt, was bedeutet, dass bei den kontinuierlichen Variablen folgende maximale Abweichung der aktuellen Werte zu den Samples erlaubt ist: Der Wert für den Energieverbrauch darf höchstens um den Faktor 0.4 kWh abweichen und der Wert für die Kosten höchstens um den Faktor 0.2 €. Dieses Wertepaar wurde gewählt, da diese Kombination die geringstmögliche an Werten ist, sodass das Ergebnis des Algorithmus valide Pfade enthält. Wurden kleinere Werte gewählt, so enthielt das Ergebnis keine validen Pfade mehr. Unter Verwendung des ausgewählten Wertepaares erzeugt der Lösungsalgorithmus 8 mögliche Pfade. Aus Zeitgründen erfolgte jedoch nur die Analyse des wahrscheinlichsten dieser Pfade.

Zur Pfadanalyse existieren mehrere Ansätze. Ein Ansatz ist es, anhand des Pfades den Verlauf der kontinuierlichen Variablen zu rekonstruieren und diesen anschließend mit den Samplewerten aus der Spur zu vergleichen. Der zweite Ansatz ist der direkte Vergleich des ausgewählten Pfades mit den bei der Testdatengenerierung ebenfalls erzeugten Daten der Zustandswechsel. Für dieses Modell werden beide Ansätze durchgeführt. Zunächst

wurde die Rekonstruktion der kontinuierlichen Variablen durchgeführt. Die Basis hierfür bildet der Pfad (Tabelle A.1 im Anhang), welcher der wahrscheinlichste erzeugte Pfad mit einer Wahrscheinlichkeit $1.78 * 10^{-7}$ ist.

	EV	K
NV	0,2	0,1
HV	0,5	0,2

Tabelle 4.2.: Rate Rewards Modell 1

Die im Pfad verwendeten Kürzel *HV* und *NV* stehen für die jeweiligen Zustände *Hoher Verbrauch* sowie *Niedriger Verbrauch*. Die Matrix (Tabelle 4.1.3) stellt noch einmal den Rate Reward für die kontinuierlichen Variablen *Energieverbrauch* und *Kosten* anhand des aktuellen Systemzustands dar. Anhand dieser Daten können nun die folgenden Graphen 4.3 und 4.4 rekonstruiert werden.

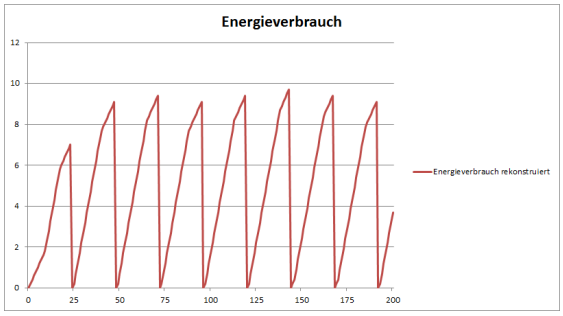

Abbildung 4.3.: Verlauf Energieverbrauch

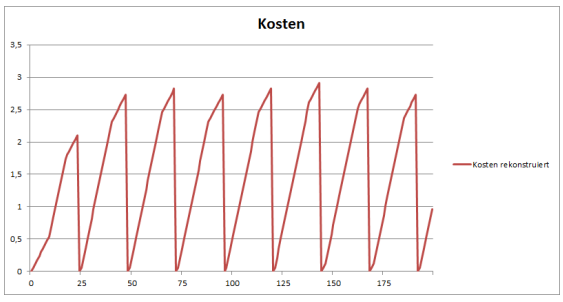

Abbildung 4.4.: Verlauf Kosten

Diese Graphen 4.3 und 4.4 zeigen den Verlauf der kontinuierlichen Werte auf Basis des rekonstruierten Pfades. Wenn nun diese beiden Graphen mit den jeweiligen Samples aus

der Eingabespur kombiniert werden, entstehen die beiden folgenden Graphen 4.5 und 4.6.

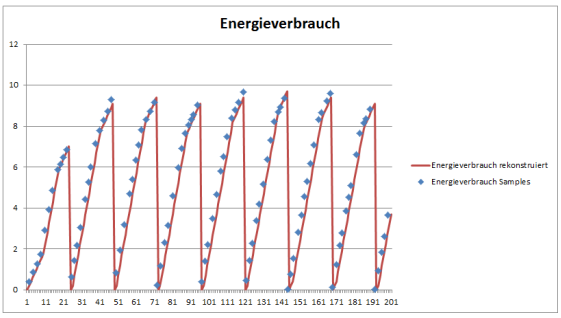

Abbildung 4.5.: Verlauf Energieverbrauch + Samples

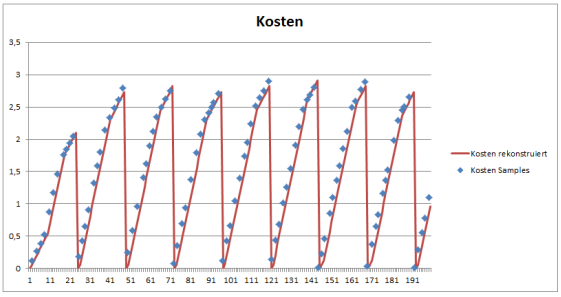

Abbildung 4.6.: Verlauf Kosten + Samples

Die Graphen 4.5 und 4.6 zeigen, dass zwischen den Samples und den rekonstruierten Daten nur minimale Abweichungen auftreten und somit ist zumindest teilweise gezeigt, dass die Verhaltensrekonstruktion für dieses Modell erfolgreich war. Wenn nun noch gezeigt wird, dass der zweite Ansatz ebenfalls korrekte Ergebnisse liefert, ist gezeigt, dass die Verhaltensrekonstruktion mit Hilfe der neu entwickelten Erweiterung für dieses Modell korrekt arbeitet.

Für den zweiten Analyseansatz wird nun der Pfad mit den gesammelten Daten der Zustandsübergänge verglichen. Die folgenden Tabellen zeigen die Gegenüberstellung dieser Daten.

Der Vergleich der Daten zeigt, dass zum einen der Zustandsverlauf sowohl des Ursprungsmodells als auch des rekonstruierten Modells miteinander übereinstimmen. Da dies allein noch kein Zeichen für die Korrektheit ist, muss die Übereinstimmung zusätzlich zu den Zuständen auch die Zeitpunkte der Zustandswechsel betreffen. Dies ist hier ebenfalls der

Zeit	Zustand
8,14	HV
15,85	NV
24,04	HV
38,90	NV
46,54	HV
63,44	NV
70,86	HV
86,93	NV
95,67	HV
112,44	NV
120,43	HV
136,76	NV
144,66	HV
160,92	NV
169,08	HV
184,25	NV
192,24	HV

Tabelle 4.3.: tatsächliche Zustandswechsel Modell 1

Zeit	Zustand
9	HV
17	NV
25	HV
40	NV
49	HV
65	NV
73	HV
88	NV
97	HV
113	NV
121	HV
138	NV
146	HV
162	NV
170	HV
185	NV
193	HV

Tabelle 4.4.: Zustandsübergänge des verwendeten Pfades

Fall, da die vorhandenen Abweichungen zwischen den Zeitpunkten minimal sind und sich bei den Abweichungen von einem Wert kleiner als 1 durch die Verwendung diskreter Zeitschritte erklären lassen. Die übrigen Abweichungen lassen sich durch die erlaubte Abweichung der kontinuierlichen Werte erklären, was bewirkt, dass die Transitionen schon wenige Zeiteinheiten früher beziehungsweise später feuern können. Aus diesem Ergebnis lässt sich schlussfolgern, dass die Verhaltensrekonstruktion für dieses Modell erfolgreich war. Da beide Analyseansätze positive Ergebnisse liefern, wurde somit gezeigt, dass das erste der beiden Ziele in Bezug auf dieses Experiment erreicht wurde.

Nun gilt es zu zeigen, dass das zweite Ziel ebenfalls erreicht wurde. Hierfür wird die Laufzeit des Lösungsalgorithmus mit der Erweiterung mit der Laufzeit des ursprünglichen Algorithmus verglichen. Hierbei erfolgt zunächst eine Normalisierung anhand der Anzahl erzeugter Proxel um eine Vergleichbarkeit verschiedener Modelle zu ermöglichen. Für die Verarbeitung dieses Modells benötigte der Lösungsalgorithmus 0.04 Sekunden in denen 2703 Proxel bearbeitet wurden. Daraus ergibt sich eine durchschnittliche Be-

arbeitungszeit von $\sim 1.5 * 10^{-5}$ Sekunden pro Proxel. Das Vergleichsmodell aus [1] ist von ähnlicher Komplexität und besteht aus 3 Zuständen mit 4 Transitionen. Für die Verarbeitung dieses Modells mit Hilfe des ursprünglichen Algorithmus, für welches 200 Samples verteilt auf 1000 Zeiteinheiten genutzt wurden, wurden für die Bearbeitung von 902635 Proxeln 29,65 Sekunden benötigt. Damit ergibt sich eine durchschnittliche Bearbeitungszeit pro Proxel von $\sim 3.3 * 10^{-5}$ Sekunden pro Proxel. Damit zeigt sich, dass die entwickelte Erweiterung dazu fähig ist, die Rekonstruktion von partiell beobachtbaren Modellen, in denen der diskrete Systemzustand durch die kontinuierlichen Variablen beeinflusst wird, im selben Zeitrahmen durchzuführen, den der ursprüngliche Algorithmus für dessen Problemstellungen benötigt.

Somit wurde nun für das erste der gewählten Modelle gezeigt, dass die an den Algorithmus gesetzten Ziele erreicht wurden. Um nun Aussagen über eine größere Menge an Modellen treffen zu können, wird im nächsten Kapitel ein komplexeres Modell untersucht.

4.2. Modell 2

Dieser Abschnitt beschäftigt sich mit dem zweiten gewählten Modell und ist inhaltlich ähnlich dem vorherigen Abschnitt aufgebaut. Zunächst erfolgt eine Beschreibung des Sachverhalts, dessen Verhalten rekonstruiert werden soll. Anschließend erfolgt die Beschreibung, wie und in welchem Umfang die benötigten Testdaten generiert wurden. Abschließend erfolgt die Auswertung der durchgeführten Experimente, welche die Darstellung der gewonnenen Ergebnisse sowie deren Auswirkungen auf die gesetzten Ziele umfasst.

4.2.1. Modellbeschreibung

In dem zweiten Modell wird ebenso wie im ersten Modell das dynamische Verhalten zwischen Energieproduzent und Energieverbraucher nachgebildet, in diesem Fall jedoch aus der Sicht des Energieversorgers. Als Zeiteinheit wurden wiederum Stunden gewählt. Als die kontinuierlichen Variablen werden die *Energiereserven* (in kWh) des Versorgers sowie ein möglicher *Ökobonus* (in €) genutzt. Die Energiereserven entsprechen der für einen Kunden reservierten Energiemenge und werden durch eine festgesetzte Speichergröße von 10 kWh limitiert. Der Ökobonus wird durch die Art der gelieferten Energie beeinflusst und soll die Verwendung erneuerbarer Energien belohnen. Der Versorger kann

sich entscheiden, ob er Energie produziert, was Auswirkungen auf die Energiereserven hat. Für den Wechsel zwischen den Zuständen *Verbraucher produziert* und *Verbraucher produziert nicht* wird im nach $\sim Norm(2,\ 0.5)h$ verteilten Abstand der aktuelle Stand der Energiereserven überprüft. Die Transition *Produktionsstopp* kann nur dann schalten, wenn die aktuellen Energiereserven den maximalen Wert von 10 kWh erreicht haben. Die Transition *Produktionsstart* kann nur dann schalten, wenn die aktuellen Energiereserven weniger als 2 kWh betragen. Der Kunde kann die aktuellen Energiereserven ebenfalls beeinflussen, indem er entweder Energie verbraucht oder aber nicht. Der Wechsel zwischen *Kunde verbraucht* und *Kunde verbraucht nicht* ist mit einer Verteilung von $\sim Norm(14,1)h$ versehen und Wechsel zurück ist mit einer Verteilung von $\sim Norm(10,1)h$ versehen. Die Energieart, welche der Energieproduzent liefert, kann entweder *erneuerbare Energie* oder *Kernenergie* sein. Der Wechsel zwischen den Energiearten ist ebenfalls mit einer Verteilung von $\sim Norm(2,\ 0.5)h$ belegt und an Bedingungen geknüpft. Damit der Wechsel von *Kernenergie* zu *erneuerbare Energie* ausgelöst werden kann, müssen die aktuellen Energiereserven mindestens 5 kWh betragen und sobald die aktuellen Energiereserven kleiner 2 kWh sind, ist der Wechsel von *erneuerbare Energie* zu *Kernenergie* möglich. In der folgenden Grafik 4.7 wird der komplette Sachverhalt noch einmal verdeutlicht.

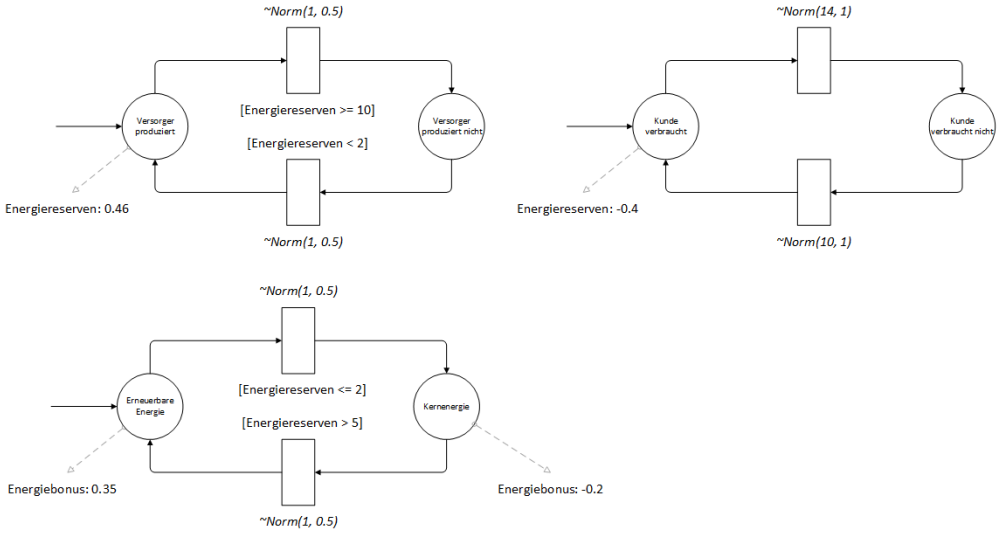

Abbildung 4.7.: Petrinetz Modell 2

Mögliche Fragestellungen, welche sich mit einem solchem Modell beantworten lassen könnten, umfassen die Kategorisierung des Kundenverhaltens und des Kundentyps. Dies kann nun genutzt werden, um die bereitzustellenden Energiereserven zu planen, um

notfalls Alternativen bereitstellen zu können.

4.2.2. Testdatengenierung

Ebenso wie bei dem ersten Modell wird hier auf das Simulationstool Anylogic zurückgegriffen. Ebenfalls wurde der Sachverhalt dieses Modells in ein Simulationsmodell geformt. Das so entstandene Simulationsmodell wird in der folgenden Grafik 4.8 veranschaulicht.

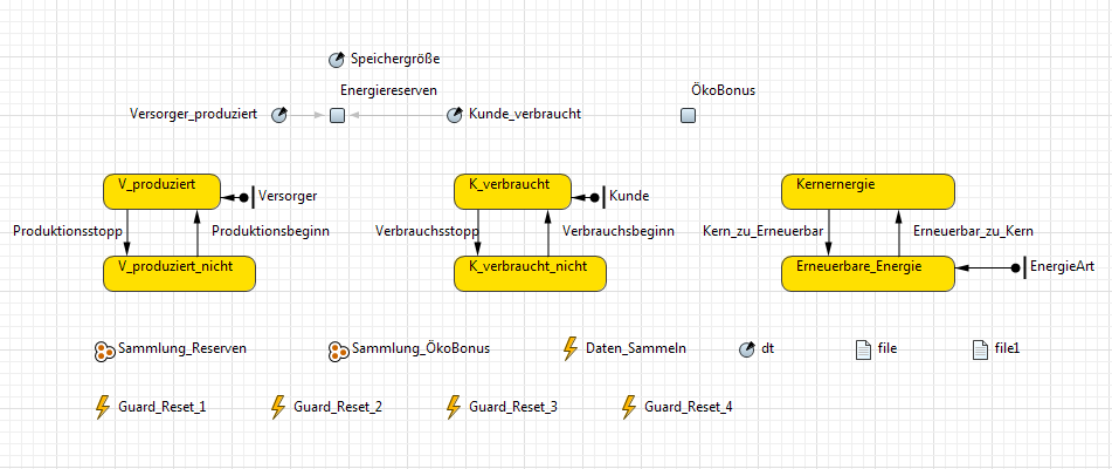

Abbildung 4.8.: Anylogic-Implementierung Modell 2

Die in der Grafik 4.8 dargestellten Zustände *V_produziert*, *V_produziert_nicht*, *K_verbraucht*, *K_verbraucht_nicht*, *Kernenergie* und *Erneuerbare_Energie*, wobei V für Versorger und K für Kunde steht, sowie die Transitionen *Produktionsstopp*, *Produktionsbeginn*, *Verbrauchsstopp*, *Verbrauchsbeginn*, *Kern_zu_Erneuerbar* und *Erneuerbar_zu_Kern* entsprechen den Spezifikationen des Modells aus Grafik 4.7. Die Erfassung der Samplewerte erfolgt nach denselben Vorgehen wie bei dem ersten Modell und erfasst nach einer Verteilung von $\sim Norm(2,\ 0.5)h$ die Werte für die kontinuierlichen Variablen, wobei der Zeitrahmen der Simulation ebenfalls 200 Zeiteinheiten beträgt. Somit werden ebenfalls rund 100 Samplewerte für die Spur erzeugt. Um nun die Struktur dieser Spur zu verdeutlichen, wird im Folgenden ein kurzer Ausschnitt der Spur präsentiert.

Zeit	Energiereserven	Ökobonus
2,30	0,14	0,58
4,39	0,26	0,16
6,75	0,40	-0,31
7,87	0,47	-0,54
9,39	0,56	-0,84
12,00	0,72	-1,36
14,32	1,47	-1,83
15,94	2,22	-2,15

Tabelle 4.5.: Ausschnitt aus der verwendeten Spur

Wie schon bei dem vorherigen Modell werden auch hier die Zeitpunkte der Zustandsübergänge mit ihren Zielzuständen protokolliert, um nachher die Rekonstruktion auf Korrektheit überprüfen zu können.

4.2.3. Auswertung

Um die Erzeugung valider Pfade zu ermöglichen, muss auch für dieses Experiment zunächst ein Parametertuning des ϵ-Parameters durchgeführt werden. Hier wurde das Wertepaar (1.0, 0.8) gewählt, wodurch bei den kontinuierlichen Variablen folgende maximale Abweichungen erlaubt sind: Der Wert für die Energiereserven darf höchstens um den Faktor 1.0 kWh abweichen und der Wert für den Ökobonus höchstens um den Faktor 0.8 €. Unter Verwendung dieses Wertepaares werden 152 Pfade erstellt, welche das durch die gegebene Spur beschriebene Verhalten erzeugen können. Auch hier wird nur der wahrscheinlichste der Pfade zur Analyse herangezogen.

Die im vorherigen Abschnitt genutzten Analysemethoden werden auch für dieses Modell genutzt. Zuerst wird anhand des ausgewählten Pfades der Verlauf der kontinuierlichen Variablen rekonstruiert und anschließend mit den Samplewerten aus der Spur verglichen. Anschließend erfolgt der direkte Vergleich des genutzten Pfades mit den während der Testdatengenerierung erzeugten Daten der Zustandsübergänge. Die Basis für diese Ansätze ist der Pfad (Tabelle A.2 im Anhang), welcher eine Wahrscheinlichkeit von $4.16 * 10^{-18}$ aufweist.

	Energiereserven	Ökobonus
VP_KV_KE	0,06	-0,2
VP_KV_EE	0,06	0,35
VP_KVN_KE	0,46	-0,2
VP_KVN_EE	0,46	0,35
VPN_KV_KE	-0,4	-0,2
VPN_KV_EE	-0,4	0,35
VPN_KVN_KE	0	-0,2
VPN_KVN_EE	0	0,35

Tabelle 4.6.: Rate Rewards Modell 2

Die im Pfad verwendeten Kürzel VP, VPN, KV, KVN, KE und EE stehen für die jeweiligen Zustände *V_produziert*, *V_produziert_nicht*, *K_verbraucht*, *K_verbraucht_nicht*, *Kernenergie* und *Erneuerbare_Energie*. Die Matrix stellt noch einmal den Rate Reward für die kontinuierlichen Variablen *Energiereserve* und *Ökobonus* anhand des aktuellen Systemzustands dar. Anhand dieser Daten können nun die folgenden Graphen 4.9 und 4.10 rekonstruiert werden.

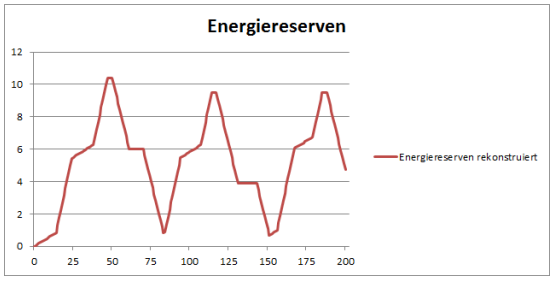

Abbildung 4.9.: Verlauf Energiereserven

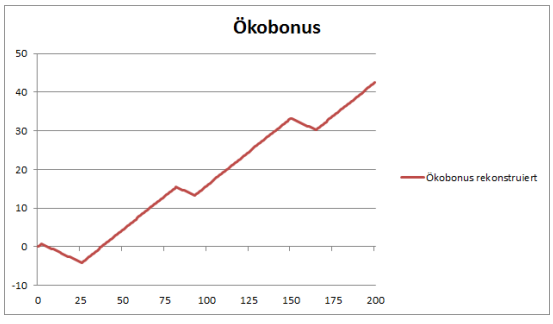

Abbildung 4.10.: Verlauf Ökobonus

Diese Graphen 4.9 und 4.10 zeigen den Verlauf der kontinuierlichen Werte auf Basis des rekonstruierten Pfades. Wenn nun diese beiden Graphen mit den jeweiligen Samples aus der Eingabespur kombiniert werden, entstehen die beiden folgenden Graphen 4.11 und 4.12.

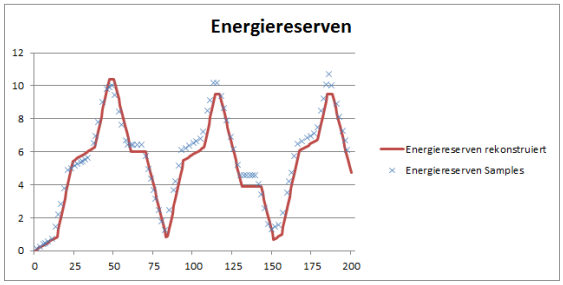

Abbildung 4.11.: Verlauf Energiereserven + Samples

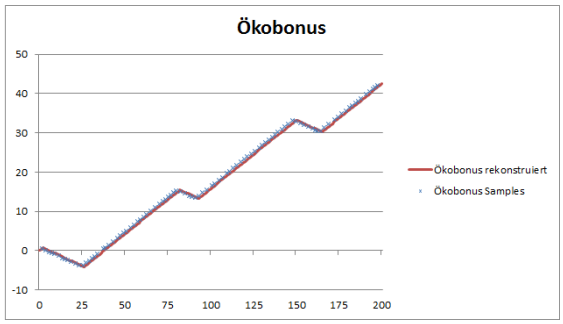

Abbildung 4.12.: Verlauf Ökobonus + Samples

Die Graphen 4.11 und 4.12 zeigen, dass zwischen den Samples und den rekonstruierten Daten nur minimale Abweichungen auftreten und somit ist zumindest teilweise gezeigt, dass die Verhaltensrekonstruktion für dieses Modell erfolgreich war. Wenn nun noch gezeigt wird, dass der zweite Ansatz ebenfalls korrekte Ergebnisse liefert, ist gezeigt, dass die Verhaltensrekonstruktion mit Hilfe der neu entwickelten Erweiterung für dieses Modell korrekt arbeitet.

Der zweite Analyseansatz erfordert nun, dass der erzeugte Pfad mit den protokollierten Daten der Zustandsübergänge verglichen wird. Die folgenden Tabellen zeigen nun diese Gegenüberstellung.

Zeit	Zustand
1,82	VP_KV_KE
12,58	VP_KVN_KE
21,73	VP_KV_KE
22,89	VP_KV_EE
35,69	VP_KVN_EE
46,17	VPN_KVN_EE
46,61	VPN_KV_EE
61,67	VPN_KVN_EE
70,31	VPN_KV_EE
76,92	VPN_KV_KE
78,88	VP_KV_KE
84,35	VP_KVN_KE
92,80	VP_KVN_EE
95,37	VP_KV_EE
109,43	VP_KVN_EE
116,28	VPN_KVN_EE
118,21	VPN_KV_EE
132,91	VPN_KVN_EE
142,26	VPN_KV_EE
150,19	VPN_KV_KE
150,56	VP_KV_KE
157,65	VP_KVN_KE
165,40	VP_KVN_EE
166,49	VP_KV_EE
180,23	VP_KVN_EE
188,04	VPN_KVN_EE
191,22	VPN_KV_EE

Tabelle 4.7.: tatsächliche Zustandswechsel Modell 1

Zeit	Zustand
2	VP_KV_KE
14	VP_KVN_KE
24	VP_KV_KE
26	VP_KV_EE
38	VP_KVN_EE
47	VPN_KVN_EE
50	VPN_KV_EE
61	VPN_KVN_EE
70	VPN_KV_EE
82	VPN_KV_KE
83	VP_KV_KE
84	VP_KVN_KE
93	VP_KVN_EE
94	VP_KV_EE
107	VP_KVN_EE
114	VPN_KVN_EE
117	VPN_KV_EE
131	VPN_KVN_EE
143	VPN_KV_EE
150	VPN_KV_KE
151	VP_KV_KE
156	VP_KVN_KE
165	VP_KVN_EE
167	VP_KV_EE
179	VP_KVN_EE
185	VPN_KVN_EE
188	VPN_KV_EE

Tabelle 4.8.: Zustandsübergänge des verwendeten Pfades

Der Vergleich der Daten zeigt zunächst, dass der mit dem Lösungsalgorithmus erzeugte Pfad hinsichtlich des Zustandsverlaufs mit den Daten übereinstimmt, welche bei der Erzeugung der Spur gesammelt wurden. Es bestehen jedoch Abweichungen in den Zeitpunkten, zu welchen diese Zustandsübergänge auftraten. Diese sind zumeist jedoch mini-

mal und lassen sich teilweise durch die Diskretisierung der Zeitschritte im Lösungsalgorithmus erklären. Die größeren Abweichungen, also die Abweichungen von zwei bis fünf Zeiteinheiten, lassen sich unter anderem durch die erlaubten Abweichungen der kontinuierlichen Variablen erklären. Da sowohl der Zustandsverlauf als auch die Zeitpunkte der Zustandsübergänge in einem annehmbaren Rahmen übereinstimmen, lässt sich daraus schließen, dass die Verhaltensrekonstruktion für dieses Modell ebenfalls erfolgreich war. Da damit beide Analyseansätze positive Ergebnisse liefern, wurde gezeigt, dass das erste der gesetzten Ziele, die Rekonstruktion von einfachen Modellen, erfolgreich erreicht wurde.

Um nun die Ergebnisse des Lösungsalgorithmus hinsichtlich des zweiten Ziels zu bewerten, wird ebenfalls dessen Laufzeit mit Laufzeit des ursprünglichen Algorithmus verglichen. Als Vergleichswert wird die im vorherigen Abschnitt genannte Laufzeit von $\sim 3.3 * 10^{-5}$ Sekunden pro bearbeitetem Proxel angenommen. Die Laufzeit, welcher der Lösungsalgorithmus für die Berechnung aller möglichen Pfade benötigt, beträgt 0.47 Sekunden, in denen 24069 Proxel verarbeitet werden. Daraus ergibt sich eine durchschnittliche Laufzeit von $\sim 1.95 * 10^{-5}$ Sekunden pro Proxel. Damit liegt die Laufzeit des weiterentwickelten Lösungsalgorithmus im selben Zeitrahmen wie die Laufzeit des ursprünglichen Algorithmus. Somit wurde gezeigt, dass auch das zweite der gesetzten Ziele für dieses Modell erreicht wurde und die entwickelte Erweiterung in der Lage ist, die an sie gestellten Anforderungen zu erfüllen.

Durch die Analyse der gewonnenen Ergebnisse wurde nun auch für das zweite der gewählten Modelle gezeigt, dass die gesetzten Ziele erreicht wurden. Welche Schlussfolgerungen sich daraus ziehen lassen, dass für die beiden der ausgewählten Modelle gezeigt werden konnte, dass jeweils die beiden Ziele erreicht wurden, wird im nächsten Kapitel näher erläutert werden.

5. Abschluss

5.1. Zusammenfassung

In dieser Arbeit wurde eine Weiterentwicklung der Hidden non-Markovian Reward Models präsentiert, welche es im Gegensatz zu dem aktuellen Stand der Technik auch das Verhalten von Modellen rekonstruieren kann, in welchen durch die kontinuierlichen Variablen, also die Reward-Werte, der diskrete Systemzustand beeinflusst werden kann.

Um die Korrektheit der entwickelten Erweiterung zu zeigen, wurden zwei Testmodelle entworfen. Beide befassen sich mit dem dynamischen Verhältnis zwischen Energieversorger und dem Energieverbraucher und stellen diesen Sachverhalt sowohl aus Verbraucher- als auch Versorgersicht dar.

Anhand dieser beiden Modelle wurden nun Experimente durchgeführt, in denen gezeigt wurde, dass das Verhalten dieser Modelle erfolgreich rekonstruiert werden konnte und dies in einem der Modellgröße angemessenen Zeitrahmen durchgeführt werden kann.

5.2. Fazit

Da für zwei Anwendungsfälle die Funktionalität der Erweiterung aufgezeigt wurde, kann man daraus schlussfolgern, dass die Funktionalität der entwickelten Erweiterung auch für beliebige Modelle ähnlicher und höherer Komplexität gegeben ist. Da durch die Erweiterung auch die Anzahl der potentiellen Anwendungsfelder gestiegen ist, ist es denkbar, dass eine Vielzahl interessanter Anwendungen nun umsetzbar ist. Zudem wurde ein möglicher Anwendungsbereich für Hidden non-Markovian Reward Models aufgezeigt, welcher nur durch zwei simple Modelle angerissen wurde. Der Anwendungsbereich der erneuerbaren Energien besitzt nichtsdestotrotz ein vielversprechendes Potential für weitere Anwendungen.

5.3. Ausblick

Im bisherigen Zustand ist die Beeinflussung des diskreten Systemzustands durch die kontinuierlichen Werte nur mit Hilfe der Transitionen und damit an eine festgelegte Schaltzeit der Transition gebunden. Eine mögliche Erweiterung wäre, dass der Systemzustand zum frühestmöglichen Zeitpunkt gewechselt wird, sobald der gewünschte Vergleichswert erreicht wird. Diese Vorgehensweise würde auch der in Kapitel 3.1 erwähnten ursprünglichen Definition einer Guard-Funktion entsprechen.

Zudem könnte die Betrachtung der kontinuierlichen Werte auf einen längeren Zeitrahmen ausgeweitet werden, so dass zum Beispiel der Zustandswechsel ausgelöst werden kann, wenn ein bestimmte Reward-Wert über einen festgelegten Zeitraum einen Vergleichswert erreicht hat.

Literaturverzeichnis

[1] Claudia Krull and Graham Horton. Hidden non-markovian reward models: virtual stochastic sensors for hybrid systems. In *Proceedings of the Winter Simulation Conference*, page 224. Winter Simulation Conference, 2012.

[2] Claudia Krull and Graham Horton. Hidden non-markovian models: Formalization and solution approaches. In *Proceedings of 6th Vienna International Conference on Mathematical Modelling, Vienna, Austria*, 2009.

[3] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[4] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[5] Robert Buchholz. *Conversive Hidden non-Markovian Models*. PhD thesis, Otto-von-Guericke-Universit¨at Magdeburg, 2012.

[6] Jogesh Muppala, Gianfranco Ciardo, and Kishor S Trivedi. Stochastic reward nets for reliability prediction. *Communications in reliability, maintainability and serviceability*, 1(2):9–20, 1994.

[7] William H Sanders and John F Meyer. A unified approach for specifying measures of performance, dependability and performability. In *Dependable Computing for Critical Applications*, pages 215–237. Springer, 1991.

[8] Graham Horton. A new paradigm for the numerical simulation of stochastic petri nets with general firing times. In *Proceedings of the European Simulation Symposium*, pages 129–136, 2002.

[9] JF Peters, A Skowron, Z Suraj, and S Ramanna. Guarded transitions in rough petri nets. *complement (x)*, 10, 1999.

[10] Anylogic-website http://www.xjtek.com/.

[11] Robert Buchholz, Claudia Krull, Thomas Strigl, and Graham Horton. Using hidden non-markovian models to reconstruct system behavior in partially-observable systems. In *Proceedings of the 3rd International ICST Conference on Simulation Tools*

and Techniques, page 86. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010.

[12] Reinhard German and Christoph Lindemann. Analysis of stochastic petri nets by the method of supplementary variables. *Performance Evaluation*, 20(1):317–335, 1994.

A. Anhang

Zeit	Transition	Zustand
0		NV
9	Niedrig_Hoch	HV
17	Hoch_Niedrig	NV
25	Niedrig_Hoch	HV
40	Hoch_Niedrig	NV
49	Niedrig_Hoch	HV
65	Hoch_Niedrig	NV
73	Niedrig_Hoch	HV
88	Hoch_Niedrig	NV
97	Niedrig_Hoch	HV
113	Hoch_Niedrig	NV
121	Niedrig_Hoch	HV
138	Hoch_Niedrig	NV
146	Niedrig_Hoch	HV
162	Hoch_Niedrig	NV
170	Niedrig_Hoch	HV
185	Hoch_Niedrig	NV
193	Niedrig_Hoch	HV

Tabelle A.1.: Ausgewählter Pfad für Modell 1

Zeit	Transition	Zustand
0		VP_KV_EE
2	Erneuerbar_Kern	VP_KV_KE
14	Verbrauchsstop	VP_KVN_KE
24	Verbrauchsstart	VP_KV_KE
26	Kern_Erneuerbar	VP_KV_EE
38	Verbrauchsstop	VP_KVN_EE
47	Produktionsstop	VPN_KVN_EE
50	Verbrauchsstart	VPN_KV_EE
61	Verbrauchsstop	VPN_KVN_EE
70	Verbrauchsstart	VPN_KV_EE
82	Erneuerbar_Kern	VPN_KV_KE
83	Produktionsstart	VP_KV_KE
84	Verbrauchsstop	VP_KVN_KE
93	Kern_Erneuerbar	VP_KVN_EE
94	Verbrauchsstart	VP_KV_EE
107	Verbrauchsstop	VP_KVN_EE
114	Produktionsstop	VPN_KVN_EE
117	Verbrauchsstart	VPN_KV_EE
131	Verbrauchsstop	VPN_KVN_EE
143	Verbrauchsstart	VPN_KV_EE
150	Erneuerbar_Kern	VPN_KV_KE
151	Produktionsstart	VP_KV_KE
156	Verbrauchsstop	VP_KVN_KE
165	Kern_Erneuerbar	VP_KVN_EE
167	Verbrauchsstart	VP_KV_EE
179	Verbrauchsstop	VP_KVN_EE
185	Produktionsstop	VPN_KVN_EE
188	Verbrauchsstart	VPN_KV_EE

Tabelle A.2.: Ausgewählter Pfad für Modell 2

Selbstständigkeitserklärung

Ich versichere hiermit, dass ich die vorliegende Masterarbeit „Erweiterung von hybriden HnMRM durch Einfluss des kontinuierlichen Teils auf den Diskreten“ selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Jens Schiborowski
Magdeburg, 24.06.2013