Verifikation von Spezifikationsmodellen mit Intervall-Petri-Netzen

Vesselka Duridanova, Thorsten Hummel, Olga Fengler, Wolfgang Fengler
Technische Universität Ilmenau
Fakultät für Informatik und Automatisierungstechnik
Fachgebiet Rechnerarchitektur
Email: thorsten.hummel|olga.fengler@tu-ilmenau.de

Zusammenfassung: *Der Entwurf von komplexen eingebetteten Systemen ist durch die Größe und die Vielzahl der unter Echtzeit arbeitenden Komponenten häufig eine große Herausforderung. Dabei spielt die Einhaltung zeitlicher Eigenschaften eine wichtige Rolle. Im Folgenden wird die formale Verifikation von eingebetteten Systemen mit Zeitintervall-Petri-Netzen diskutiert. Message Sequence Charts des zu verifizierenden eingebetteten Systems werden mit Hilfe des Entwurfswerkzeugs „VisualObjectNet++" in Intervall-Petri-Netze umgewandelt und anschließend analysiert.*

1. Einleitung

Die Einhaltung zeitlicher Eigenschaften für komplexe eingebettete Systeme ist häufig von entscheidender Bedeutung für deren erfolgreiche Anwendung. Deren Berücksichtigung in frühen Entwurfsphasen und die konsequente Verifikation in allen Phasen des Entwurfs erhöht die Entwurfsqualität und verkürzt die Entwurfszeiten.
Message Sequence Charts (MSC) [1] werden seit einigen Jahren unterstützend im Entwicklungsprozess für die Spezifikationsmodellierung eingesetzt. Schon im frühen Stadium geben sie einen Überblick über die geplante Lösung der Aufgabenstellung. Durch ihre grafische Form und verschiedene Abstraktionsebenen können sie zur Verdeutlichung der Funktion dienen.
Wir zeigen in diesem Artikel, wie für die zeitliche Verifikation der Entwurfsmodelle zeitbehaftete MSC in Intervall-Petri-Netze[2] transformiert werden. Für die Umwandlung und die anschließende Verifikation der Intervall-Petri-Netze(IPN) wurde das Entwurfswerkzeug „VisualObjectNet++" funktionell erweitert.

2. Erweiterung des Entwurfswerkzeuges „VisualObjectNet++"

Das Entwurfswerkzeug „VisualObjectNet++" wurde an der TU Ilmenau für die Modellierung und Simulation von hybriden Petri-Netzen entwickelt. Der Formalismus der Hybriden Petri-Netze enthält diskrete und kontinuierliche Komponenten[3]. Zu den diskreten Netzelementen gehören auch diskrete Transitionen, die um das Konzept der Zeitintervalle erweitern wurden.
Die Analysemethode von Popova[2] für IPN wurde für die Verifikation von MSC-Diagrammen komplexer eingebetteter Systeme erweitert und implementiert.
Ein MSC-Konverter wurde implementiert, der die automatische Umwandlung von MSC-Diagrammen (textliche Beschreibungsform) in IPN vornimmt.

Erweiterung des Formalismus der Hybriden Petri-Petze um Intervall-Petri-Netze

Zeitbehaftete Petri-Netze wurden eingeführt, um die Zeit, ohne die keine vollständige Anforderungsanalyse und Verifikation eines Systems möglich ist, in die klassischen Petri-Netze zu integrieren. Es gibt zwei Grundarten von zeitbehafteten Petri-Netzen. Einerseits kann einer Transition eine Zeit, die sie zum Feuern benötigt, zugeordnet werden. Andererseits kann jeder Transition ein Zeitintervall zugeordnet werden, innerhalb dessen die Transition feuern kann.
Dies ist bei den hier verwendeten Intervall-Petri-Netzen der Fall. Eine schaltfähige Transition darf nur innerhalb dieses Intervalls schalten und muss spätestens nach Intervallende geschaltet haben, es sei denn, sie verliert in der Zwischenzeit ihre Schaltfähigkeit. Erlangt die Transition später ihre Schaltfähigkeit zurück, beginnt der Ablauf der im Intervall festgelegten Zeit von vorn. Das Intervall wird hierzu durch ein Tupel (a,b) von rationalen Zahlen, mit $0 \le a \le b \le \infty$, $a < \infty$ dargestellt. Die Erweiterung eines Petri-Netzes um das Konzept der Transitions-Intervallzeiten kann wie folgt formuliert werden:

Z = (P,T,F,V,m_0,I) sei ein Intervall Petri-Netz wenn gilt:

(P,T,F,V,m_0) ist ein Petrinetz mit $I : T \to Q_0 \times (Q_0 \cup \{\infty\})$ mit $\forall t \;/\; t \in T \to \left(\underline{I}(t) \le \overline{I}(t) \wedge \underline{I}(t) < \infty\right)$.
Dabei bezeichnet man I als Zeitfunktion des Intervall Petrinetzes Z. $\underline{I}(t)$ bezeichnet man als Startzeit und $\overline{I}(t)$ als Stoppzeit der Transition t.

Eine Transition eines Intervall-Petri-Netzes ist schaltfähig, wenn alle üblichen Vor- und Nachbedingungen erfüllt sind und der Wert der Zeitfunktion J mindestes dem Wert der Startzeit $\underline{I}$ entspricht.

Formal: Die Transition t eines Intervall-Petri-Netzes Z =(P,T,F,V,m_0,I) ist dann schaltfähig, wenn gilt:

1. $t^- \le m$
2. $\underline{I}(t) \le J(t)$.

Der Zustand $z = (m, J)$ geht durch den Zeitablauf $\tau \in Q_0$ in einen neuen Zustand $z = (m', J')$ über (Notation: $z = (m, J) \xrightarrow[\tau]{e} z = (m', J')$), wenn gilt:

1. $m = m'$
2. $\forall t\left(t \in T \wedge J(t) \neq \# \to J(t) + \tau \le \overline{I}(t)\right)$
3. $J'(t) = \begin{cases} J(t) + \tau, & falls \quad t^- \le m \\ \# \quad , & sonst \end{cases}$

Wird die Stoppzeit $\overline{I}(t)$ einer schaltfähigen Transition erreicht, so gerät diese Transition in den so genannten Schaltzwang und wird zum Feuern gezwungen.

Der Zustand $z=(m,J)$ geht durch das Feuern einer Transition $\hat{t}$ in einen neuen Zustand $z=(m',J')$ über (Notation: $z=(m,J)\xrightarrow{\hat{t}}z'=(m',J')$), wenn gilt:

1. $\hat{t}$ ist feuerbereit bei $z=(m,J)$
2. $m'=m+\Delta\hat{t}$
3. $J(\hat{t})=\begin{cases}\#\ ,falls & t^- \not\leq m'\\ J(t),falls & t^- \leq m \wedge t^- \leq m'\\ 0\ ,sonst & \end{cases}$

Bild 1 zeigt das Hauptfester des Werkzeugs „VisualObjectNet++" mit einem diskreten Intervall-Petri-Netz.

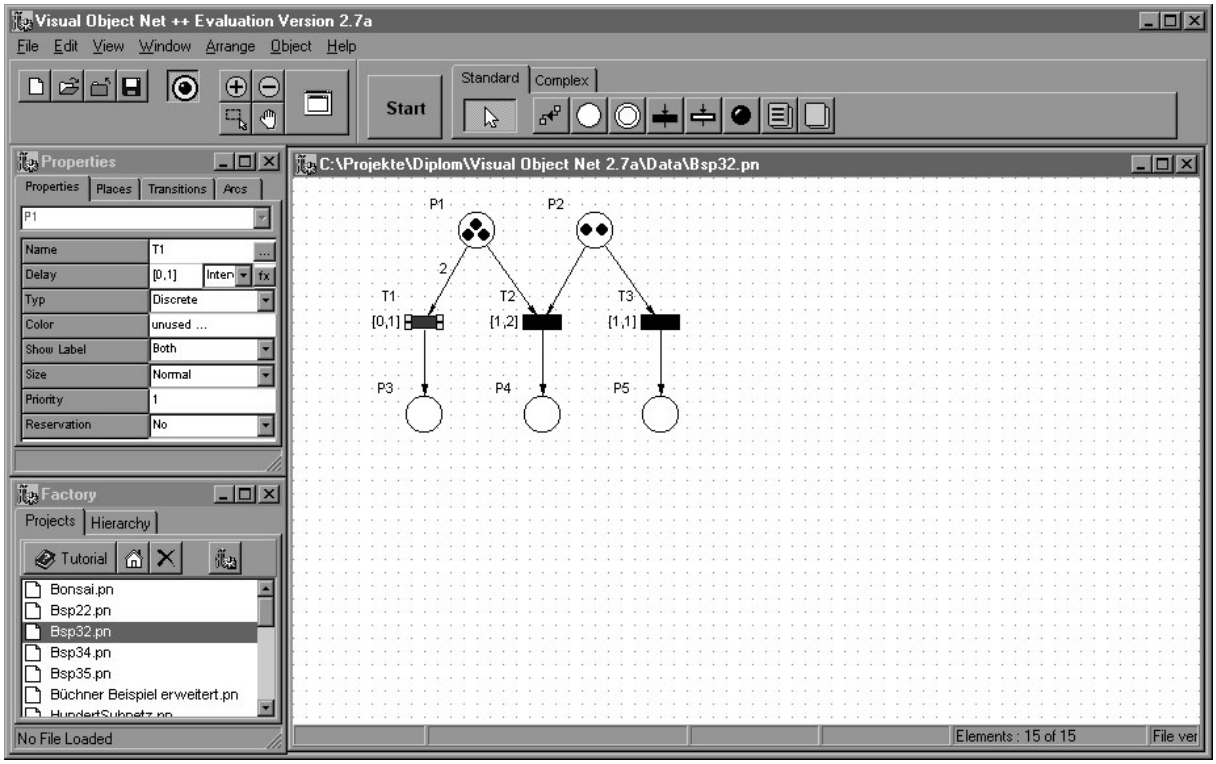

Bild 1. Das Entwurfswerkzeug „VisualObjectNet++" mit einem IPN-Beispiel

Bei der Realisierung in „VisualObjectNet++" wurde festgelegt, dass die Intervalle jeder intervallbehafteten Transition wie allgemein üblich in eckigen Klammern angegeben und mit Kommata voneinander getrennt werden**.** Die Zeitintervalle können feste oder variable Werte aus dem ganzen oder reellen Zahlenbereich annehmen. Unendliche Werte sind zugelassen und können als „i" für infinite bzw. ∞ angegeben werden. Im Simulator wird für unendlich (∞) eine sehr große Zahl ($10*e^{11}$) eingesetzt. Der Schaltpunkt t intervallbehafteter Transitionen ist in den angegebenen Grenzen variabel. Bei jedem erneuten Schalten hat t einen anderen Wert, der durch eine Zufallsfunktion erzeugt wird.

Umwandlung von Message Sequence Charts (MSC) in Intervall-Petri-Netze

Generell beschreiben die MSC eine Funktionalität oder ein Verhalten, dabei steht die Interaktion in Form von Nachrichten im Vordergrund. Der Nachrichtenaustausch zwischen verschiedenen Instanzen, im objektorientierten Umfeld auch Objekte genannt, erfolgt meist asynchron. Die Instanzen können Softwareprozesse, Hardware oder auch Benutzer darstellen. In einem MSC werden zwei Darstellungsdimensionen unterschieden, die *vertikale* Dimension, welche einer klassischen Zeitachse entspricht, während *horizontal* die betrachteten Objekte spezifiziert werden. Die Zeitkonstrukte in zeitbehafteten MSC sind mit Nachrichten verbunden. Dabei werden diese Zeitbedingungen neben der Lebenslinie bei Senden oder Empfangen einer Nachricht notiert. Diese Zeitbedingungen stellen dabei die einzuhaltende Zeitspanne zwischen zwei Nachrichten dar. An Stelle der Verwendung von Zeitangaben ist auch die Verwendung von Zeitintervallen möglich.
Der Message Sequence Chart Konvertierer von „VisualObjectNet++“ ist in der Lage, MSC-Diagramme von allen MSC-Entwurfsumgebungen zu konvertieren, die sich an die textlichen Beschreibungsvorschriften des Z.120–Standards der International Telecommunication Union (ITU) halten und diese Beschreibung in einer ASCII-Datei liefern.
Die Umwandlung übernimmt ein Parser, der die Schlüsselworte aus der textlichen Beschreibung der MSC in Elemente der Intervall-Petri-Netze übersetzt. Der Parser bearbeitet in mehreren Durchgänge die textliche Beschreibung des MSC-Modells, bevor er ein neues Projekt in „VisualObjectNet++” anlegt und das konvertierte Intervall-Petri-Netz im Hauptfenster darstellt.

3. Formale Analyse des Intervall-Petri-Netzes des Spezifikationsmodells

Die Umwandlung des MSC-Spezifikationsmodells in ein IPN-Modell dient der frühzeitigen Verifikation der Systemeigenschaften.
In erster Linie kann eine Analyse der strukturellen Eigenschaften vorgenommen werden, die durch die Theorie der klassischen Platz-Transitions-Petri-Netze angeboten wird. Die Analyse des Modells liefert Angaben über folgende Eigenschaften: Erreichbarkeit und Nichterreichbarkeit von Systemzuständen, Beschränktheit von Systemressourcen (z.B. Speicher) oder Verklemmungen im System, d.h. Synchronisationsfehler zwischen Teilmodulen, die zu unerwünschten Stillstand im Gesamtsystem führen.
Um den Zustand eines Intervall-Petri-Netzes beschreiben zu können, muss im Gegensatz zu den klassischen Petri-Netzen neben den Markierungen auch die Zeitkomponente berücksichtigt werden, da sich das Netzverhalten bei unterschiedlichen Intervallzeiten grundlegend verändern kann. Dazu wird neben den Markierungen zu jeder Transition auch die Zeit J gespeichert, die seit dem Erhalten der Schaltfähigkeit (Aktivität) vergangen ist. Ist eine Transition nicht schaltfähig oder verliert sie ihre Schaltfähigkeit, so wird die Zeitmessung gestoppt und die Transition wird als inaktiv vermerkt. (siehe [2]).

Der Zustand ist also ein 2-Tupel, wobei das erste Element eine erreichbare Markierung des Netzes, einen Zustand unter Vernachlässigung der zeitlichen Komponente, darstellt. Im zweiten Element werden die Zeiten angegeben, die eine Transition bereits aktiviert ist, bzw. # für eine nicht aktivierte Transition.

$$s = ((1,0,1,0,1)\,,\begin{pmatrix} 0 \\ \sharp \\ \sharp \\ 0 \end{pmatrix})$$

ist der Startzustand des Netzes in Bild 2.

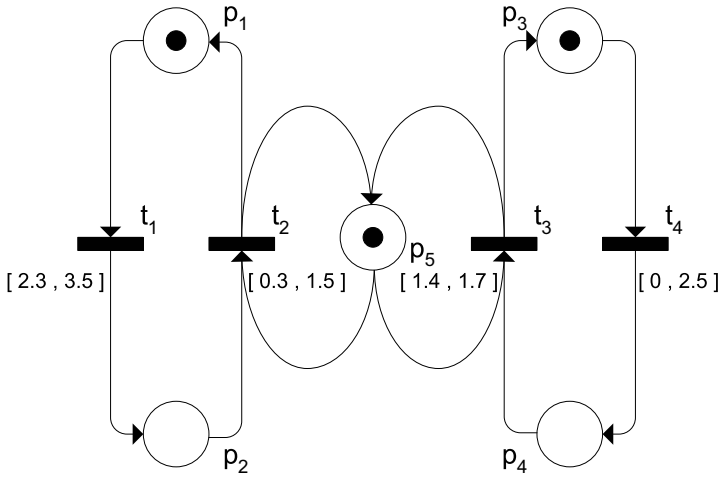

Bild 2: Ein Intervall-Petri-Netz

Die Zahl der möglichen Folgezustände eines Intervall-Petri-Netzes ist wegen der kontinuierlichen Zeit weit höher, als bei Netzen ohne Zeiterweiterung.
Zwei aus unendlich viele mögliche Folgezustände sind zum Beispiel:

$$s_1 = ((1,0,1,0,1)\,,\begin{pmatrix} 1.3 \\ \sharp \\ \sharp \\ 1.3 \end{pmatrix}) \text{ oder } s_2 = ((0,1,1,0,1)\,,\begin{pmatrix} \sharp \\ 0 \\ \sharp \\ 2.2 \end{pmatrix})$$

Ein Zustandsübergang wird wie folgt dargestellt: $s_0 \xrightarrow{t_1} s_1$.

Die Menge der Zustände und der Zustandsübergänge ist bei einem IPN per Definition somit unendlich.
Durch folgende Annahmen wird die Menge der Zustände auf eine endliche Menge reduziert:
Die Church-Rosser-Eigenschaft: Ein Zustandswechsel, s_1 nach s_2, vollzieht sich durch Schalten und Ablauf einer gewissen Zeit oder auch durch Ablauf der Zeit und anschießendes Schalten.
Wenn zwei Transitionen in einem Zustand gleichzeitig schaltfähig sind, so stehen sie in Konflikt und haben dieselbe Startzeit.
Jeder erreichbare Zustand des zeitkontinuierlichen Intervall-Petri-Netzes ist in einer ganzzahligen Zeit erreichbar, d.h. es kann die Einschränkung vorgenommen werden, dass das Netz nur zu Zeitpunkten schaltet, die sich durch natürliche Zahlen darstellen lassen.
∞ wird als Obergrenze verboten.

Netze, welche die genannten Eigenschaften besitzen, haben eine endliche Anzahl von Zuständen und sind damit analysierbar. Der Erreichbarkeitsgraph ähnelt dem einfacher Petri-Netze, Zustandübergänge sind zusätzlich von den diskreten Zeitpunkten abhängig. Für komplexe Netze besteht hier die Gefahr, dass der Erreichbarkeitsgraph explodiert, d.h. alle mögliche Zustände und Zustandsübergänge im Netz sind in einer endlichen Rechenzeit nicht bestimmbar.
Die Einführung von Zustandsklassen, die Folgen von Zuständen und Zustandsübergängen, bzw. Pfaden im Erreichbarkeitsgraphen beschreiben, führt zu einer komprimierten Beschreibung der möglichen Entwicklungstendenzen im Netz. Die Konstruktion des gesamten Erreichbarkeitsraumes bleibt jedoch auch unter Einführung von Zustandsklassen für komplexe Systeme in der Regel ein nicht polynomiales Problem.

Die Analysemethode

Die entwickelte Analysemethode, basierend auf Popova [2], gibt die Möglichkeit, einzelne Entwicklungsprozesse im Netz, d.h. Pfade im Erreichbarkeitsraum, zu verfolgen und unter vertretbaren Rechenaufwand Prozesseigenschaften zu analysieren. Für einen angegebenen Anfangszustand und eine Transitionssequenz, welche im realen System eine Kette von Ereignissen, Nachrichten und Aktionen darstellt, wird die Dynamik im Netz untersucht.
Die Analysemethode verläuft in zwei Schritten. Für die betrachtete Transitionssequenz werden zuerst die Folgezustände, ohne Berücksichtigung der Zeitintervalle ausgerechnet. Dadurch wird bestimmt, ob auch ohne Zeitrestriktionen eine solche Aktionsfolge möglich ist, d.h. ob die Transitionen in der angegebenen Reihenfolge schalten können und welche Folgezustände dabei entstehen.
Im zweiten Schritt der Analyse wird unter Berücksichtigung der Transitionsintervalle die Zustandsklasse im Netz für die betrachtete Transitionssequenz gebildet. Jede Klasse enthält alle erreichbaren Zustände mit derselben Markierungskomponente. Die einzelnen Zustände einer Klasse unterscheiden sich durch ihre Zeitkomponenten. Änderungen in der Markierung führen zum Klassenwechsel. Die zweite Komponente der Klasse markiert die ablaufende Zeit aller aktiven Transitionen, die noch nicht geschaltet haben. Die dritte Klassenkomponente stellt ein System von Ungleichungen dar, das die Zeitvorgeschichte kapseln, die zu diesem Zustand geführt hat und die Zeitrestriktionen unter denen der Zustand weiter beibehalten wird.

Für das Netz von Bild 3 und die Transitionssequenz t1,t2 ergeben sich folgenden Zustandsklassen:

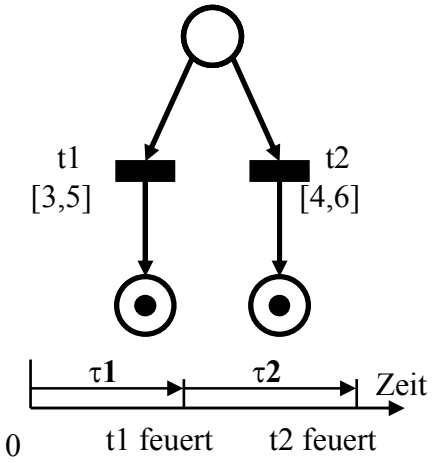

Bild 3: Ein IPN (Endzustand) mit den zugehörigen Zustandsklassen

$$C_0 = \left\{ (2,0,0) \wedge J\begin{bmatrix} t1 \\ t2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_1 \end{bmatrix} \wedge (0 \le \tau_1 \le 5) \right\}$$

$$C_0 \xrightarrow{t1} C_1$$

$$C_1 = \left\{ \begin{array}{l} (1,1,0) \wedge J\begin{bmatrix} t1 \\ t2 \end{bmatrix} = \begin{bmatrix} \tau_2 \\ \tau_1 + \tau_2 \end{bmatrix} \wedge (3 \le \tau_1 \le 5) \\ \wedge (0 \le \tau_2 \le 5) \wedge (0 \le \tau_1 + \tau_2 \le 6) \end{array} \right\}$$

$$C_1 \xrightarrow{t2} C_2$$

$$C_2 = \left\{ \begin{array}{l} (0,1,1) \wedge J\begin{bmatrix} t1 \\ t2 \end{bmatrix} = \begin{bmatrix} \# \\ \# \end{bmatrix} \wedge (3 \le \tau_1 \le 5) \\ \wedge (0 \le \tau_2 \le 5) \wedge (4 \le \tau_1 + \tau_2 \le 6) \end{array} \right\}$$

Die Methode der Klassenbildung wird ausführlich in [4] beschrieben.
Das Verfahren basiert auf die Untersuchung der Schaltfähigkeit aller Transitionen mit vorschreitender Zeit. Das Ungleichungssystem zu jeder Klasse kann unter Formulierung von Zusatzbedingungen mit den Methoden aus der linearen Optimierung gelöst werden.
Dadurch lassen sich für die angegebene Schaltsequenz mehrere Zeitkennziffern ermitteln:

- Worst-Case-Zeit: kürzeste und längste Zeitdauer,
- Die Einhaltung einer gegebenen Zeiteinschranke (Deadline),
- Start- und Stoppzeiten für die Intervalle einer bzw. mehrerer Transitionen, bei denen die Schaltsequenz eine minimale Dauer nicht überschreitet,
- Start- und Stoppzeiten für die Intervalle einer bzw. mehrerer Transitionen, bei denen ein bestimmter Zustand im Netz nicht erreicht werden kann (Lifelock).

Implementierung der IPN-Analyse

Die Ermittlung der oben aufgezählten Kennziffern ergibt ein Problem aus der Theorie der linearen Optimierung: für das Ungleichungssystem wird eine optimale Lösung im Raum der Zeitparameter gesucht. In der Praxis verwendet man für die Lösung derartige Optimierungsproblemen bei komplexen Systemen, mit einer großen Zahl von Variablen x_i zwei grundlegende Algorithmen: der Simplexalgorithmus und die Interior-Point-Methode [5,6].
Beide Methoden wurden bezüglich ihrer Anwendbarkeit für komplexe Systeme ausgewertet.
„Die Simplexmethode ist ein iteratives Verfahren bei dem eine Anfangslösung so lange schrittweise verbessert wird, bis eine optimale Lösung gefunden ist oder die Nichtlösbarkeit erkannt wird“ [5].
Der Interior-Point-Algorithmus benötigt weniger Iterationen als die Simplexmethode, braucht jedoch pro Iteration mehr Rechenzeit. Damit kann nicht garantiert werden, dass der Interior-Point-Algorithmus schneller arbeitet. Allgemein wird jedoch angenommen, dass er vor allem bei großen Problemen schneller sind [9,10].
Die Zielstellung war, eine Programmbibliothek herauszufinden, die das Optimierungsproblem auch bei sehr komplexen Systemen in vertretbarer Rechenzeit bewältigt. Nach gründlicher Auswerteanalyse fiel die Wahl auf die Programme LPAKO und LPABO von OrLab [11]. **LPAKO v4.8f** ist ein „Linear Programming Solver“ welcher die duale Simplexmethode zur Lösung verwendet. **LPABO v5.9f** ist ein „Linear Programming Solver“ welcher die infeasible primal-dual Interior-Point-Methode verwendet.
Beide Programme können sowohl das MPS-Format [8] als auch ein eigenes Dateiformat importieren. Das verwendete eigene Dateiformat ‚F‘ ist der Normalform linearer Optimierungsaufgaben sehr ähnlich und kann sehr einfach in jedes Programm implementiert werden. Die Ungleichungen können in der Form $Ax \le b$, $Ax = b$ oder $Ax \ge b$ eingegeben werden und werden intern in Normalform, also in das Format $Ax \le b$ umgewandelt.
Die Analysemethode wurde, wie bereits erwähnt, innerhalb des Werkzeuges „VisualObjectNet++“ implementiert. Die Analyse für eine ausgewählte Transitionssequenz läuft nach folgendem Schema ab:
Zuerst werden die Matrizen des Ungleichungssystems für die lineare Optimierung gebildet. Anschließend wird aus diesen Matrizen die Eingabedatei für das ausgewählte Optimierungsprogramm erzeugt und dieses Programm über einen DLL-Function-Call ausgeführt. Abschließend folgt entweder eine Datenauswertung sowie Anzeige der gewonnenen Daten oder eine Fehlerbehandlung.

Bild 4.a zeigt das Eingabefenster für die Analyse. Das Formular „Intervall-Petri-Netz-Analyse“ wird über die Toolbar (Bild 1) oder über das Menü Extras des Simulationspanels (Bild 4.b) gestartet.

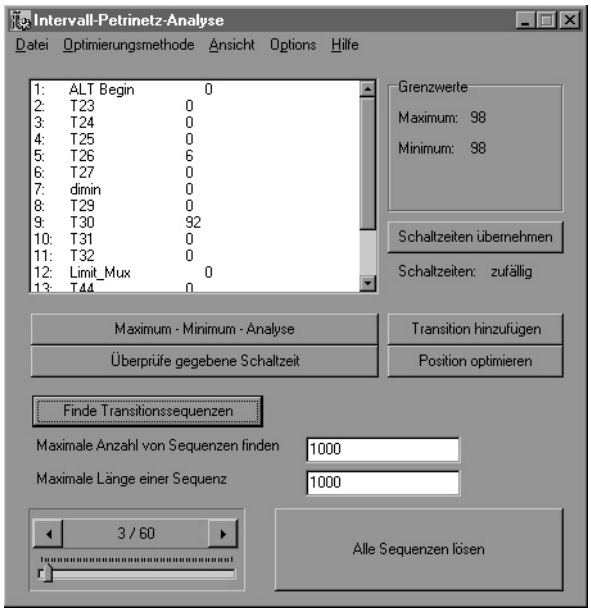

Bild 4.a: Das Fenster für die IPN-Analyse in VisualObjectNet++

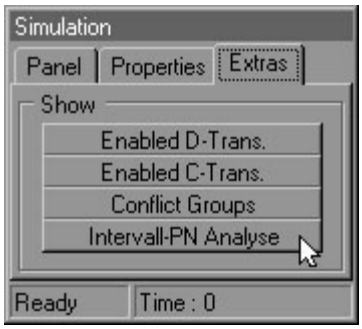

Bild 4.b: Integration der Analyse-methode in den Simulator

Die eigentliche Grenzwertanalyse wird über den Befehl „Maximum-Minimum-Analyse“ ausgeführt. Für den Test, ob eine gegebene Transitionssequenz in einer bestimmten Zeit schalten kann, wurde eine zusätzliche Funktion implementiert, welche über den Befehl „Überprüfe gegebene Schaltzeit“ gestartet wird.
Um das Systemverhalten simulieren zu können, ist es möglich, die Ergebnisse der Analyse den jeweiligen Transitionen direkt zuzuweisen, um somit von einer zufälligen Schaltzeit auf die in der Analyse ermittelten Schaltzeit umzuschalten zu können (Befehl „Schaltzeiten übernehmen“).
Mit dem Befehl „Transition hinzufügen“ lässt sich sehr einfach die Reihenfolge der Transitionen in einem Transitionssequenz ändern, neue Transitionen hinzufügen oder Transitionen löschen.
Die wichtigsten Auswahlparameter für die Analyse sind die Optimierungsmethode und die Transitionssequenz. Für die Bildung einer Transitionssequenz gibt es zwei Möglichkeiten. Die Transitionssequenz kann durch den Anwender selbst auf verschiedenen im Werkzeug implementierten Wegen bestimmt werden. Der direkte Weg z.B. führt über das Editorfenster des Petrinetzes. Im Popupmenü des Editors (Bild 1.) wurde eine Methode namens „Add to Analyse“ implementiert, welche beim Auswählen einer beliebigen Transition diese in die Transitionssequenz einträgt. Es wurde ferner eine Methode zur automatischen Generierung aller in einem Netz vorhandenen schaltfähigen Transitionssequenzen entwickelt. (Befehl „Finde Transitions-sequenzen“), Man kann anschließend entweder eine bestimmte Transitionssequenz aus der Liste aller gefundenen schaltfähige Transitionssequenzen auswählen und analysieren oder eine Maximum-Minimum-Analyse aller Transitionssequenzen durchführen.
Die Ergebnisse der kompletten Analyse können zusätzlich als Sequenzserie, in Form einer SEQ-Datei, gespeichert bzw. wieder geladen werden.

4. Anwendungsbeispiel

Im Folgenden soll nun gezeigt werden, wie mit der vorgestellten Methode eine reale Problemstellung gelöst werden kann. Dabei geht es um die Spezifikation und die Untersuchung des Zeitverhaltens eines Mehrkoordinatenmesssystems. Bild 5 stellt das Positionsmesssystem dar, das ein Bestandteil des in [12,13] beschriebenen integrierten Mehrkoordinatenantriebes ist. Mit Hilfe dreier Messachsen X,Y1 und Y2 soll die Lage innerhalb des Systems festgestellt werden.

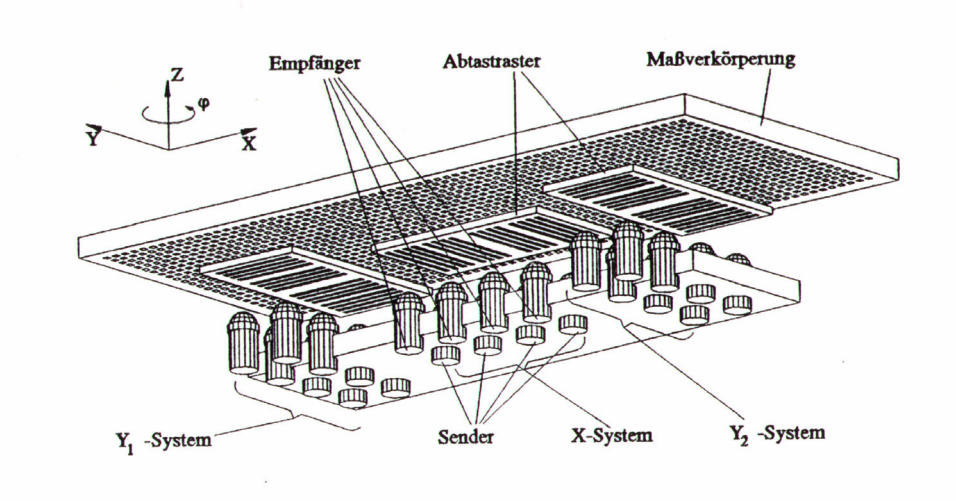

Bild 5: Mehrkoordinatenmesssystem

Durch die Bewegung der Abtastfenster über einem Kreuzraster entstehen an den Empfängern Sinus- und Kosinussignale, die anschließend normiert werden. Durch die Verwendung zweier Abtastraster für die Y-Richtung kann eine Verdrehung des Läufers gemessen und korrigiert werden. Die Messung wird inkrementell durchgeführt. Das erfordert die ständige Auswertung der Position des Läufers relativ zum Anfangspunkt der Messung. Aus den Eingangssignalen wird die Periodenzahl gebildet. Zur Bestimmung der aktuellen Periodenzahl wird ein Zählsystem eingeführt, welches die Periodenzahl beim Überschreiten der Signalperiode je nach Bewegungsrichtung des Läufers erhöht bzw. verringert. Dazu wird zunächst für die aktuellen Signale der zugehörige Quadrant des Vollkreises bestimmt. Anschließend wird, durch Vergleich mit dem Vorzustand des Systems, die zur Messung benötigte Periodennummer gebildet und schließlich die exakte Position interpoliert.
Zur besseren Übersichtlichkeit wurde lediglich die X-Richtung betrachtet.

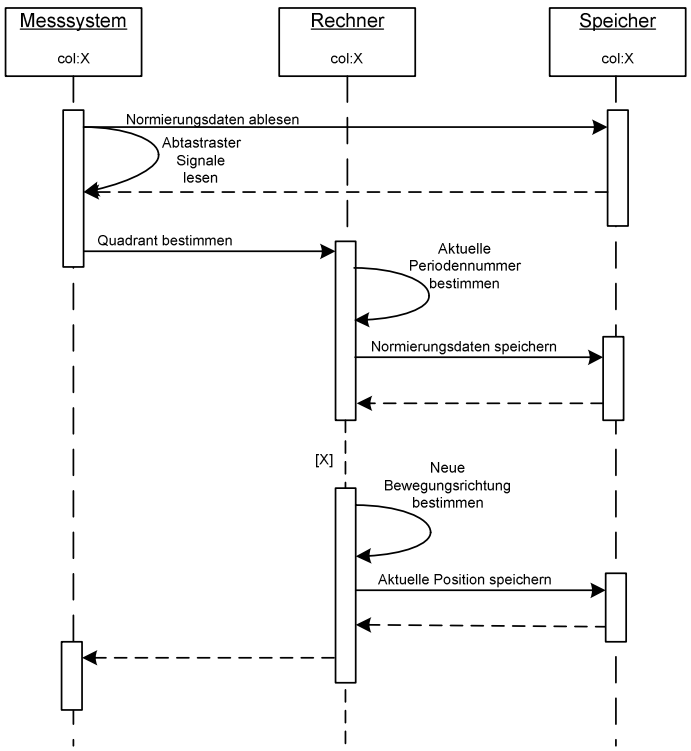

Bild 6.a MSC des Beispielsystems

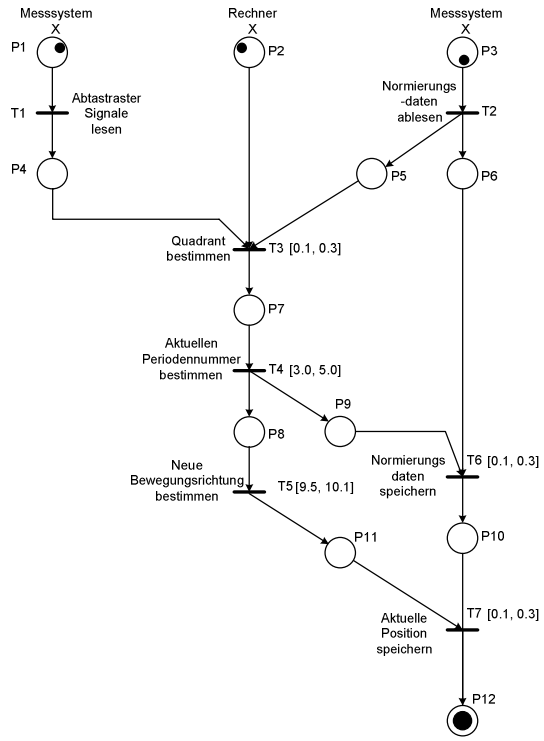

Bild 6.b IPN des Beispielsystems

Mit dem MSC (Bild 6.a) kann das System zunächst spezifiziert werden. Zur Untersuchung des Zeitverhaltens des Systems ist eine Umwandlung in ein Intervall-Petri-Netz (Bild 6.b) notwendig.

Damit kann das zu untersuchende System auf die Einhaltung bestimmter Zeitgrenzen hin analysiert werden. Für eine festgelegt Transitionssequenz und mit den angegebenen Intervallen (in µs) wurde eine Maximum-Minimum-Analyse durchgeführt, die das Worst-Case-Verhalten des Systems bestimmt. Die Analyse ergab ein Maximum von 15,7µs und ein Minimum von 12,7µs.

5. Zusammenfassung und Ausblick

Die Verifikation von Spezifikationsmodellen mit Intervall-Petri-Netzen ermöglicht die frühzeitige Kontrolle der Einhaltung zeitlicher Eigenschaften für komplexe eingebettete Systeme während des Entwurfsprozesses. Das Entwurfswerkzeug „VisualObjectNet++“ wurde für die automatische Umwandlung von Message Sequence Charts in Intervall-Petri-Netze und die formale Analyse von Intervall-Petri-Netzen funktionell erweitert. Dies ermöglicht die Verbindung der Nutzung von üblichen MSC-Entwurfsumgebungen mit den formalen Analysemöglichkeiten für Intervall-Petri-Netze.
Die gewonnen Erkenntnisse sollen für zukünftige Forschungsarbeiten zur formalen Verifikation von SoC-Entwürfen auf Systemebene genutzt werden.

Diese Arbeit wurde von der DFG unter der Kennziffer FE 373/13-2 im Rahmen des Schwerpunktprogramms „Entwurf und Entwurfsmethodik eingebetteter Systeme“ im Thema „Entwurf eingebetteter paralleler Steuerungssysteme für integrierte multi-axiale Antriebssysteme“ unterstützt.

Literatur

[1] ITU-TS. Recommendation Z. 120(11/99) : MSC 2000. Geneva, 1999.
[2] Louchka Popova-Zeugmann: Zeit-Petri-Netze. Dissertation, Humboldt-Universität Berlin, 1989.
[3] Rainer Drath: Modellierung hybrider Systeme auf Basis modifizierter Petri-Netze. Dissertation, TU Ilmenau, 1999.
[4] Marko Döring: Verifikation von Intervall-Petri-Netzen, Diplomarbeit, TU Ilmenau, 2002.
[5] Kurt Marti, Detlef Gröger: Einführung in die lineare und nichtlineare Optimierung. Physica-Verlag Heidelberg, 2000.
[6] Nikos Drakos: The LP-Book, 1993-1996: http://www.isye.gatech.edu/~spyros/LP/LP.html
[7] Stefan Tautscher: Stochastische nonlineare Progammierung: http://www.ani.univie.ac.at/~tautsche/ps/index.html
[8] The Optimization-Tree: Department of Electrical and Computer Engineering, Northwestern University Illinois: http://www.ece.nwu.edu/OTC/
[9] N. Karmarkar: A new polynomial-time algorithm for linear programming. Combinatorica, 4 (1984), pp. 373-395.
[10] H.D. Mittelmann, P. Spellucci: Decision Tree for Optimization Software: http://plato.la.asu.edu/guide.html
[11] Operations Research Laboratory: Department of Industrial Engineering, College of Engineering, Seoul National University: http://orlab.snu.ac.kr/
[12] Th. Hummel: Der Einsatz von hybriden Petri-Netzen für den Entwurf gemischt analog-digitaler eingebetteter Systeme. in: D. Monjau (Hrsg.): 4. GI/ITG/GMM-Workshop: Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen. Meißen, 19.-21.02.2001, Tagungsband Teil 2, S. 49-58. ISBN 3-0000-7440-6
]13] E. Saffert, C. Schäffel, E. Kallenbach: Control of an Integrated Multi-coordinate Drive. *Mechatronics'96*, 18.-20.09.1996, Guimaraes, Portugal, Proceedings Vol. 1, S. 151-156.