Effizient chemisch rechnen durch deterministische Reaktionssysteme mit Regelpriorisierung

T. Hinze R. Faßler T. Lenser N. Matsumaru P. Dittrich

Friedrich-Schiller-Universität Jena
Arbeitsgruppe Biosystemanalyse
Ernst-Abbe-Platz 1–4, D-07743 Jena

{hinze,raf,thlenser,naoki,dittrich}@minet.uni-jena.de

29. August 2007

Zusammenfassung

Vielteilchensysteme sind Forschungsgegenstand in Natur- und Lebenswissenschaften. Ihr molekulares Wechselwirkungsgefüge lässt sich sowohl strukturell untersuchen als auch zur Konstruktion chemischer Computer nutzen. Mathematische Beschreibungen erfolgen entweder algebraisch oder analytisch. Während die Abbildung der diskreten Arbeitsweise durch Termersetzungsmechanismen weitreichende Analogien zu Automatenmodellen aufzeigt, gestatten zeit- und wertkontinuierliche Ansätze eine effiziente Parametrisierbarkeit des dynamischen Verhaltens. Motiviert durch das Ziel, beide Sichtweisen in ein gemeinsames Modellgerüst zu integrieren, stellen wir zunächst eine rein algebraische Beschreibung reaktionsfähiger Vielteilchensysteme mittels P-Systemen Π_{PR} vor, die statt der maximal-parallelen Abarbeitung eine konsequente Priorisierung der Reaktionsregeln verwenden. Dadurch wird ein deterministisches Systemverhalten erreicht und zugleich die Möglichkeit einer Masseerhaltung geschaffen. Zusätzlich erleichtert dies den Zugang zur analytischen Methodik z.B. durch Reaktionskinetiken und Fixpunktbestimmungen. Am Beispiel des Rucksackproblems zeigen wir anschließend eine automatengestützte Transformation eines auf dynamischer Programmierung beruhenden Lösungsansatzes in Systeme des Typs Π_{PR}, deren Zeitverhalten für eine Probleminstanz simuliert wird.

1 Einführung

Konzepte des Molecular Computing werden seit mehr als zehn Jahren untersucht. Einen Schwerpunkt bildet hierbei die Frage nach Programmierparadigmen, die die massive Datenparallelität weitmöglichst zur Zeiteffizienzsteigerung einsetzen.

Während spezifische Ansätze wie das DNA-Computing submolekulare Strukturen in die Algorithmierung einbeziehen [3], verfolgen künstliche Chemien eine allgemeinere Herangehensweise [2], die auch durch P-Systeme, Modelle des Membrane Computing, aufgegriffen wird [4]. Der nichtdeterministischen Natur chemischer Reaktionsnetzwerke stehen jedoch analytisch-deterministische Beschreibungsmodelle ihres dynamischen Verhaltens gegenüber, die sich auf Reaktionskinetiken und daraus resultierende Differentialgleichungssysteme stützen. P-Systeme, die diskret nach dem Vorbild regelgesteuerter Termersetzungen funktionieren, zeichnen zumeist alle potentiell möglichen Ersetzungspfade nach und betrachten diese als gleichwertig. Diese maximal-parallele Arbeitsweise impliziert Nichtdeterminismus. Die Aufgabe, nichtdeterministische Berechnungsmodelle geeignet zu determinisieren, erweist sich als sinnvoll, um analytische und algebraische Beschreibungen rechentechnisch ineinander transformieren zu können und für die darauf jeweils aufsetzende Untersuchungsmethodik zugänglich zu machen. Bereits das Determinisieren endlicher Automaten zählt zur Klasse der NP-vollständigen Probleme.

Vor diesem Hintergrund soll eine algebraisch-deterministische Beschreibung reaktionsfähiger Vielteilchensysteme im Kontext von P-Systemen entstehen. Es bieten sich zwei Determinisierungsstrategien an: Stochastizität und Regelpriorisierung. Stochastische Modelle ermöglichen die Auswahl des wahrscheinlichsten Pfades. Eine konsequente Regelpriorisierung durch eine vollständige Ordnung in Fortführung der Ideen aus [5] abstrahiert den Aspekt, dass chemische Reaktionen mit unterschiedlichen Aktivierungsenergien behaftet sein können. Zugleich beseitigt man damit eine Unzulänglichkeit maximal-parallel arbeitender P-Systeme, die auftreten kann, wenn die Anzahl der Moleküle nicht ausreicht, um trotz vorhandener Ausgangsstoffe alle darauf anwendbaren Reaktionen gleichzeitig absättigen zu können. Werden diese dennoch ausgeführt, wird dem System mehr Material entnommen als darin vorrätig ist, was das Kriterium der Masseerhaltung verletzt. Nachfolgend werden P-Systeme vom Typ Π_{PR} eingeführt, eine Transformationsvorschrift endlicher Automaten angegeben und eine Lösung des Rucksackproblems demonstriert. Wir zeigen damit über Implementierungen logischer Gatter hinaus, wie man rechenintensive Probleme durch chemische Prinzipien lösen kann, was Anwendungsperspektiven in synthetischer Biologie, Molecular und Organic Computing eröffnet.

2 Multimengenbasierte Systemdefinition

Sei A eine beliebige Menge und $\mathbb{N}$ die Menge der natürlichen Zahlen einschl. null. $\mathcal{P}(A)$ verkörpert die Potenzmenge von A. Eine Multimenge über A ist eine Abbildung $F : A \longrightarrow \mathbb{N} \cup \{\infty\}$. $F(a)$ gibt die Vielfachheit des Vorkommens von $a \in A$ in F an. Multimengen lassen sich als elementweise Aufzählung der Form $\{(a_1, F(a_1)), (a_2, F(a_2)), \ldots\}$ darstellen, wobei $\forall (a, b_1), (a, b_2) \in F : b_1 = b_2$. Der Support $\mathrm{supp}(F) \subseteq A$ von F ist definiert durch $\mathrm{supp}(F) = \{a \in A \mid F(a) > 0\}$. Eine Multimenge F über A ist leer gdw. $\forall a \in A : F(a) = 0$. Seien F_1 und F_2 Multimengen über A. F_1 ist eine Teilmenge von F_2, notiert $F_1 \subseteq F_2$, gdw. $\forall a \in A : (F_1(a) \leq F_2(a))$. Die Multimengen F_1 und F_2 sind gleich gdw. $F_1 \subseteq F_2 \wedge F_2 \subseteq F_1$. Die Schnittmenge $F_1 \cap F_2 = \{(a, F(a)) \mid a \in A \wedge F(a) = \min(F_1(a), F_2(a))\}$, die

Multimengensumme $F_1 \uplus F_2 = \{(a, F(a)) \mid a \in A \wedge F(a) = F_1(a) + F_2(a)\}$ sowie die Multimengendifferenz $F_1 \ominus F_2 = \{(a, F(a)) \mid a \in A \wedge F(a) = \max(F_1(a) - F_2(a), 0)\}$ bilden Multimengenoperationen. Der Term $\langle A \rangle = \{F : A \longrightarrow \mathbb{N} \cup \{\infty\}\}$ beschreibt die Menge aller Multimengen über A.

Ein P-System zur Beschreibung von reaktionsfähigen Vielteilchensystemen mit konsequenter Regelpriorisierung ist ein Tupel

$$\Pi_{\mathrm{PR}} = (V, T, [\,]_1, L_0, R),$$

in welchem V das Arbeitsalphabet und $T \subseteq V$ das Terminalalphabet bilden. $[\,]_1$ bezeichnet den einzigen Reaktionsraum in Anlehnung an die bei P-Systemen übliche Notation. Die Multimenge $L_0 \subset V \times (\mathbb{N} \cup \{\infty\})$ erfasst den Initialinhalt des Systems. Jedes Molekülexemplar wird hierbei als Symbolobjekt ohne räumliche Position betrachtet. Symbolobjekte implizieren keine zusätzlichen Restriktionen für weiterführende Systemimplementierungen, da sie keine Vorgaben für innere Strukturen setzen. Die endliche Menge $R = \{r_1, \ldots, r_k\}$ fasst den verfügbaren Satz chemischer Reaktionen zusammen, wobei sich jede Reaktion $r_i \in \langle E_i \rangle \times \langle P_i \rangle$ aus der endlichen Multimenge der Ausgangsstoffe $E_i \subset V \times \mathbb{N}$ sowie der endlichen Multimenge der Reaktionsprodukte $P_i \subset V \times \mathbb{N}$ ergibt. Die Vielfachheiten der Elemente entsprechen jeweils den stöchiometrischen Faktoren. Eine Reaktionsregel $r_i = (\{(A_1, a_1), \ldots, (A_h, a_h)\}, \{(B_1, b_1), \ldots, (B_v, b_v)\})$ wird nachfolgend in der chemischen Notation $r_i : a_1\, A_1 + \ldots + a_h\, A_h \longrightarrow b_1\, B_1 + \ldots + b_v\, B_v$ geschrieben. Der Index i jeder Reaktionsregel definiert ihre Priorität bei der Ausführung, die für r_1 am höchsten und für r_k am niedrigsten ist.

Die Arbeitsweise von Π_{PR} wird induktiv durch Fortschreibung der Systemkonfigurationen L_t über die Zeit t ausgehend von L_0 nach folgendem Schema definiert:

$$\begin{aligned} L_{t+1} \quad = \quad & \big\{(a, \alpha_{a,k}) \mid \forall a \in V \ \wedge \ \alpha_{a,0} = |L_t \cap \{(a, \infty)\}| \ \wedge \ \beta_{a,i} = |E_i \cap \{(a, \infty)\}| \\ & \wedge\, \gamma_{a,i} = |P_i \cap \{(a, \infty)\}| \ \wedge \ i \in \{1, \ldots, k\} \\ & \wedge\, \alpha_{a,i} = \begin{cases} \alpha_{a,i-1} + \gamma_{a,i} - \beta_{a,i} & \text{falls } \forall a \in V \,:\, \alpha_{a,i-1} \geq \beta_{a,i} \\ \alpha_{a,i-1} & \text{sonst} \end{cases} \big\} \end{aligned}$$

Die Anwendung einer Reaktionsregel r_i erfolgt dadurch, dass alle benötigten Ausgangsstoffe in der geforderten Molekülzahl aus L_t entnommen und die zugehörigen Reaktionsprodukte hinzugefügt werden. Hierzu geschieht eine Überprüfung, ob die Reaktion abgesättigt werden kann, d.h., ob alle benötigten Ausgangsstoffe auch vorliegen. Zu diesem Zweck werden für jeden Stoff $a \in V$ die Maßzahlen $\alpha_{a,j}$ und $\beta_{a,j}$ bestimmt, die angeben, wieviele Moleküle vorhanden ($\alpha_{a,j}$) bzw. wieviele zur Umsetzung der Reaktion notwendig sind ($\beta_{a,j}$). $\gamma_{a,j}$ spezifiziert entsprechend die im Falle einer Reaktion hinzukommende Molekülanzahl des Stoffes a. Bei der Absättigungsüberprüfung geht das Iterationsschema hierarchisch vor und testet zuerst die Reaktion r_1, führt diese ggf. aus, prüft danach die Reaktion r_2 usw. bis zur Reaktion r_k. Die Abarbeitung von Π_{PR} endet, sobald keine Reaktionsregel mehr anwendbar ist. Allgemein lässt sich die von Π_{PR} erzeugte Ausgabe durch die Elementarsprache

$$L(\Pi_{\mathrm{PR}}) = \mathrm{supp}\left(\biguplus_{t=0}^{\infty} (L_t \cap \{(w, \infty) \mid w \in T\})\right) \subseteq T$$

beschreiben, wobei die Frage $L(\Pi_{\mathrm{PR}}) = \emptyset$? zum Akzeptorverhalten korrespondiert.

3 Transformation endlicher Automaten in Π_{PR}

Gegeben sei ein nichtdeterministischer endlicher Automat (NEA) $M = (Z, \Sigma, \delta, z_0, F)$ mit nichtleerer endlicher Zustandsmenge Z, Alphabet Σ, Überführungsrelation $\delta \subseteq (Z \times \Sigma) \times Z$, Startzustand $z_0 \in Z$ und Finalzustandsmenge $F \subseteq Z$. Zusätzlich gelte: $Z \cap \Sigma = \emptyset$. M akzeptiere die formale Sprache $L(M)$. Aus M lässt sich wie folgt ein verhaltensadäquates P-System Π_{PR} konstruieren:

$$
\begin{aligned}
\Pi_{\mathrm{PR}} &= (V, T, [_1]_1, L_0, \{r_1, \ldots, r_{|\delta|}\}) \\
V &= Z \cup \Sigma \cup \{\Phi, C\}, \text{ wobei o.B.d.A. } \Phi, C \notin Z \cup \Sigma \\
T &= \{\Phi\} \\
L_0 &= \{(z_0, q) \mid q = |\{(z_0, x, q') \in \delta\}|\} \cup \\
&\quad \{(w, a_w) \mid w \in \Sigma \ \wedge \ a_w = |\{(z_0, w, q') \in \delta\}|\} \cup \\
&\quad \{(C, \tau)\}, \text{ wobei } \tau \in \mathbb{N} \cup \{\infty\} \text{ (wählbarer Laufzeitparameter)} \\
r_{1\ldots m} &: \quad q + A + C \longrightarrow f + \Phi + A \quad \forall (q, A, f) \in \delta \, : \, f \in F \\
r_{m+1\ldots p} &: \quad z_0 + A + C \longrightarrow q' + z_0 + A \quad \forall (z_0, A, q') \in \delta \\
r_{p+1\ldots|\delta|} &: \quad q + A + C \longrightarrow q' + A \quad \forall (q, A, q') \in \delta \, : \, q \in Z \setminus (F \cup \{z_0\})
\end{aligned}
$$

Es gilt: $L(\Pi_{\mathrm{PR}}) = \emptyset$ gdw. $L(M) = \emptyset$. Zusätzlich zu den Molekülen, die die Zustände und Alphabetzeichen von M kodieren, werden zwei neue Molekültypen eingeführt. Φ dient als Marker, der das Erreichen eines Finalzustandes signalisiert. C steht für Clock-Moleküle, deren anfänglicher Vorrat mit jeder Transition schrittweise abgebaut wird. Bei endlicher Initialvorgabe besteht so die Möglichkeit, die Systemkonfigurationen L_t durch absteigendes Sortieren nach der Anzahl C in eine zeitliche Abfolge zu bringen und die während eines Zeitschrittes abgelaufenen Transitionen zu rekonstruieren.

4 Beispielanwendung: Rucksackproblem

Das als NP-vollständig bekannte Rucksackproblem ist durch n natürliche Zahlen $a_1, \ldots, a_n$, welche die Gewichte der korrespondierenden Gegenstände $1, \ldots, n$ repräsentieren, und ein Referenzgewicht $b \in \mathbb{N}$ definiert. Es wird gefragt, ob es eine Auswahl $I \subseteq \{1, \ldots, n\}$ aus diesen Gegenständen gibt, so dass $\sum_{i \in I} a_i = b$. Der in [1] vorgestellte Ansatz zur Lösung des Rucksackproblems mittels dynamischer Programmierung greift auf den gerichteten Graphen $\mathcal{G} = (V, E)$ mit einem $(b + 1) \cdot (n + 1)$-Knotenraster zurück, so dass $V = \{v_{(i,k)} \mid \forall i = 0, \ldots, b \forall k = 0, \ldots, n\}$. Die Kantenrelation $E \subset V \times V$ wird definiert durch $E = \{(v_{(i,k)}, v_{(i,k+1)}) \mid \forall i = 0, \ldots, b \forall k = 0, \ldots, n-1\} \cup \{(v_{(i,k)}, v_{(i+a_i,k+1)}) \mid \forall i = 0, \ldots, b \, : \, i + a_i \leq b \forall k = 0, \ldots, n-1\}$. Das Ergebnis lautet „ja“ gdw. es in $\mathcal{G}$ einen Pfad von $v_{(0,0)}$ nach $v_{(b,n)}$ gibt. Abbildung 1 veranschaulicht $\mathcal{G}$ (oben links) und einen daraus resultierenden Automaten M (oben rechts) für die Probleminstanz $n = 3, a_1 = 3, a_2 = 1, a_3 = 2, b = 3$. Ein daraus resultierendes P-System Π_{PR} besitzt die Gestalt:

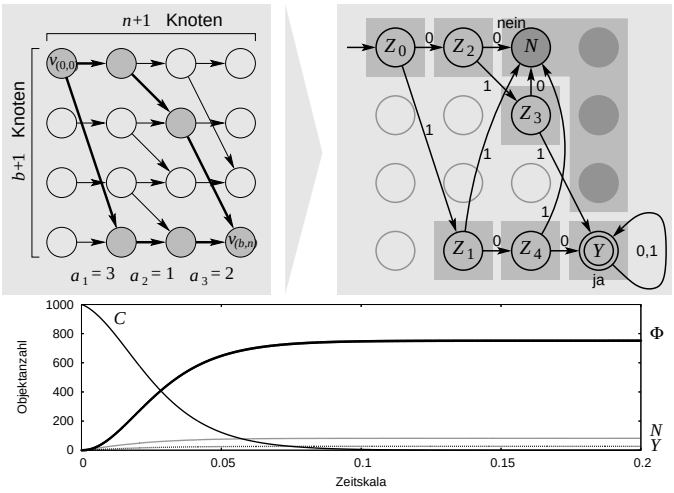

Abbildung 1: Konstruktion von M (oben rechts) aus dem Ansatz zur Lösung des Rucksackproblems mittels dynamischer Programmierung nach [1] (oben links), Simulation des dynamischen Systemverhaltens von Π_{PR} mit Copasi (unten)

$$
\begin{array}{rcl}
\Pi_{\mathrm{PR}} & = & (\{Z_0, Z_1, Z_2, Z_3, Z_4, N, Y, 0, 1, \Phi, C\}, \{\Phi\}, [\,]_1, L_0, \{r_1, \ldots, r_{12}\}) \\
L_0 & = & \{(Z_0, 2), (0, 1), (1, 1), (C, \tau)\}
\end{array}
$$

$$
\begin{array}{lll}
r_1 : Z_3 + 1 + C \longrightarrow Y + \Phi + 1 & r_5 : Z_0 + 1 + C \longrightarrow Z_1 + 1 + Z_0 & r_9 : Z_2 + 1 + C \longrightarrow Z_3 + 1 \\
r_2 : Z_4 + 0 + C \longrightarrow Y + \Phi + 0 & r_6 : Z_0 + 0 + C \longrightarrow Z_2 + 0 + Z_0 & r_{10} : Z_2 + 0 + C \longrightarrow N + 0 \\
r_3 : Y + 1 + C \longrightarrow Y + \Phi + 1 & r_7 : Z_1 + 1 + C \longrightarrow N + 1 & r_{11} : Z_3 + 0 + C \longrightarrow N + 0 \\
r_4 : Y + 0 + C \longrightarrow Y + \Phi + 0 & r_8 : Z_1 + 0 + C \longrightarrow Z_4 + 0 & r_{12} : Z_4 + 1 + C \longrightarrow N + 1
\end{array}
$$

Abbildung 1 (unten) zeigt die Anreicherung der Markermoleküle Φ (Problemlösung „ja“) sowie den Abbau der Clock-Moleküle (initial: $\tau = 1000$). Der Vorrat an alphabetzeichenkodierenden Molekülen bleibt konstant.

Danksagung. Die vorliegende Arbeit wurde aus Mitteln des 6. EU-Rahmenprogrammes für das Forschungsprojekt ESIGNET (Evolving Cell Signalling Networks in Silico, Projektnr. 12789) und aus Mitteln des BMBF (0312704A) gefördert.

Literatur

[1] E. Baum, D. Boneh. *Running dynamic programming algorithms on a DNA computer.* Proc. DNA2, DIMACS **44**:77-86, 1999

[2] P. Dittrich, J. Ziegler, W. Banzhaf. *Artificial Chemistries – A Review.* Artificial Life **7(3)**:225-275, 2001

[3] T. Hinze, M. Sturm. *Rechnen mit DNA – Eine Einführung in Theorie und Praxis.* Oldenbourg-Wissenschaftsverlag München, Wien, 2004

[4] G. Păun. *Computing with Membranes.* Journal of Computer and System Sciences **61(1)**:108-143, 2000

[5] G. Păun. *Membrane Computing: An Introduction.* Springer-Verlag Berlin 2002