Andrzej KARATKIEWICZ
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI

Analiza współbieżnych systemów dyskretnych: zbiory uporczywe vs. symulacja współbieżna

Dr inż. Andrzej KARATKIEWICZ

Dr inż. Andrzej Karatkiewicz ukończył w roku 1993 studia na Wydziale Techniki Obliczeniowej Mińskiego Radiotechnicznego Instytutu. Obronił pracę doktorską w Białoruskim Państwowym Uniwersytecie Informatyki i Radiotechniki w roku 1998. Od roku 2000 jest adiunktem w Instytucie Informatyki i Elektroniki na Uniwersytecie Zielonogórskim. Jego zainteresowania naukowe dotyczą głównie teorii sieci Petriego oraz zagadnień analizy i weryfikacji współbieżnych systemów sterowania.

e-mail: A.Karatkiewicz@iie.uz.zgora.pl.

Streszczenie

Niezbędnym elementem procesu projektowania systemów cyfrowych jest formalna weryfikacja projektów. W tym zakresie ważne są algorytmy częściowej eksploracji przestrzeni stanów. Tematem artykułu jest opis i porównanie dwóch podstawowych metod takiej eksploracji: podejścia zbiorów uporczywych i współbieżnej symulacji. Jako model formalny wybrane zostały sieci Petriego. Wnioski dotyczą określenia rodzajów sieci, dla analizy których zalecane jest stosowanie każdego z tych podejść.

Słowa kluczowe: sieci Petriego, model formalny, systemy dyskretne, eksploracja przestrzeni stanów, wykrywanie zakleszczeń

Analysis of Parallel Discrete Systems: Persistent Sets vs. Concurrent Simulation

Abstract

Formal verification is one of the necessary steps of the process of digital system design. It is especially important for the parallel systems, such as digital control systems, because wrong interaction between concurrent processes may lead to the undesirable situations, which are difficult to predict. Full exploration of state space presents a complete picture of system behaviour, but such exploration is not practically possible for the large systems. That is the reason, why algorithms of partial state space exploration are important in practice. In this paper two basic methods of such exploration are described and compared: persistent set approach and concurrent simulation approach. We use Petri nets as the main model of a parallel discrete system. In the first section we present an introduction. The second section presents basic definitions of Petri nets and related notions. Section 3 is dedicated to persistent set approach and its concretization, the stubborn set method. It describes the approach, its main possibilities and main related theorems. Fourth section describes concurrent simulation approach and its theoretical background. Section 5 presents experimental results, which demonstrate reducing of the explored state space by two algorithms implementing the described approaches. The last section contains discussion on comparison of the methods and the conclusions. The recommendations on applying both approaches to different kinds of nets are formulated.

Keywords: Petri nets, formal model, discrete systems, deadlock detection

1. Wstęp

Model formalny, opisujący system cyfrowy, abstrahuje od pewnych szczegółów jego struktury i zachowania; mimo to, analiza takich modeli dostarcza istotnych informacji o systemie i pozwala wykryć niektóre błędy projektowe. Raczej nie da się w całości sformalizować wszystkich wymagań do projektu, bo wstępnie one są opisywane w języku naturalnym; ale znaczną ich część można formalnie opisać oraz sprawdzić, czy system im odpowiada.

W przypadku systemów współbieżnych istotna jest analiza poprawności wzajemnego oddziaływania współbieżnych procesów, które zachodzą w systemie. Pewne rodzaje wzajemnego zakłócenia ich działania albo nie powinny zachodzić w dobrze zbudowanym systemie wcale, albo nie powinny się zdarzać poza określonymi sytuacjami. Tego rodzaju właściwości systemu mogą być wykrywane za pomocą metod analizy formalnej.

Analiza współbieżnych systemów dyskretnych, takich jak systemy sterowania logicznego, w dużej mierzy sprowadza się do analizy sieci Petriego, opisujących ich strukturę. Dotyczy to zarówno specyfikacji bezpośrednio bazujących na sieci Petriego, jak i innego rodzaju specyfikacji, które jednak mogą być modelowane przez sieci Petriego. Analiza sieci Petriego w tych zastosowaniach polega głównie na sprawdzeniu, czy sieć jest żywa, bezpieczna i powtarzalna (*poprawnie zbudowana*), lub znalezieniu osiągalnych zakleszczeń albo stwierdzeniu, że takowych nie ma. Innym często spotykanym zadaniem analizy jest *analiza osiągalności*, czyli sprawdzenie, czy dane znakowanie jest osiągalne.

Choć istnieje wiele metod analizy sieci Petriego, najwięcej wiedzy o zachowaniu systemu daje eksploracja przestrzeni stanów. Rozmiar takiej przestrzeni zależy od rozmiaru sieci w sposób wykładniczy, co oznacza, że już analiza sieci o rozmiarze kilkudziesięciu miejsc i tranzycji metodą „brute force" może sprawiać duże trudności. Dlatego powstało wiele algorytmów częściowej eksploracji przestrzeni stanów, w której ilość zbadanych stanów może być wielokrotnie mniejsza, niż ilość stanów osiągalnych.

Ogólnie można powiedzieć, że metody częściowej analizy przestrzeni stanów w różny sposób postępują ze zjawiskiem *przeplotu*. Zjawisko to polega na tym, że istnienie w systemie kilku współbieżnych procesów powoduje istnienie dużej liczby osiągalnych globalnych stanów, z których większość, z reguły, nie jest istotna dla analizy zachowania systemu. Oto prosty przykład: niech sieć ma dwie współbieżne gałęzie i nie ma żadnej synchronizacji między nimi; niech każda gałąź ma n stanów. Łącznie osiągalnych stanów jest n^2, ale globalny stan końcowy ewolucji sieci byłby ten sam, gdyby każda z gałęzi została symulowana osobno, a w trakcie takiej symulacji starczyłoby zbadać tylko $2n$ stanów. Takie podejście można rozszerzyć na dużo ogólniejsze przypadki.

2. Sieci Petriego

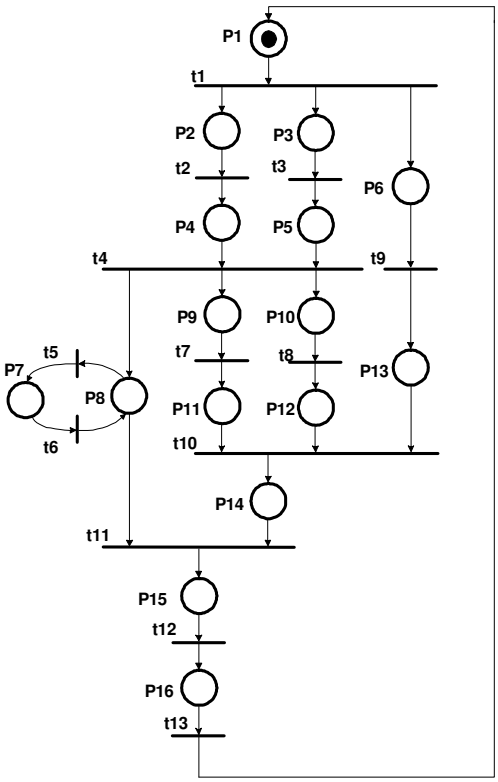
Sieć Petriego definiuje się jako czwórkę uporządkowaną $\Sigma=(P,T,F,M_0)$, gdzie: P – skończony zbiór miejsc, T – skończony zbiór tranzycji ($P \cap T = \varnothing$), $F \subset (P\times T) \cup (T\times P)$, M_0 –znakowanie początkowe. Sieć Petriego może być przedstawiona jako dwudzielny graf skierowany (Rys. 1).

Stan sieci Petriego, nazywany *znakowaniem*, jest funkcją M: $P \rightarrow \mathrm{N} \cup \{0\}$. Znakowanie może być przedstawiono poprzez *znaczniki*, które mogą się znajdować w miejscach sieci. Miejsce, w którym znajduje się co najmniej jeden znacznik, nazywa się *oznakowanym*. Jeśli wszystkie wejściowe miejsca tranzycji są oznakowane, to taka tranzycja jest *aktywna* i może zostać *wykonana*. Wykonanie tranzycji zabiera jeden znacznik z każdego z miejsc wejściowych i dodaje jeden znacznik do każdego z miejsc wyjściowych tranzycji. *Zakleszczeniem* (ang. *deadlock*) nazywa się znakowanie, w którym żadna tranzycja nie jest aktywna.

Sieć jest *żywa*, jeżeli dla każdej tranzycji t, z każdego znakowania osiągalnego ze znakowania M_0, jest osiągalne znakowanie, w którym tranzycja t może zostać wykonana. Sieć jest *bezpieczna*, jeśli w każdym z osiągalnych znakowań żadne miejsce nie zawiera więcej niż jeden znacznik. Sieć jest *ograniczona*, jeśli istnieje granica górna ilości znaczników w każdym miejscu sieci dla wszystkich osiągalnych znakowań. Sieć jest *powtarzalna*, jeśli znakowanie M_0 jest osiągalne z każdego osiągalnego znakowania.

Sieć Petriego jest nazywana *s-siecią*, jeśli w znakowaniu M_0 zawiera ona tylko jeden znacznik. Sieć Petriego jest nazywana *rozszerzoną siecią swobodnego wyboru* (*EFC-siecią*), jeżeli dowolne dwa miejsca sieci, które mają wspólna tranzycję wyjściową, mają równe zbiory tranzycji wyjściowych. Sieć Petriego jest

nazywana *α-siecią*, jeśli jest ona EFC-siecią i jednocześnie s-siecią.



Rys. 1. Sieć Petriego (wzięta z [1])
Fig. 1. A Petri net (taken from [1])

3. Zbiory uporczywe

Największą rodziną metod częściowej analizy przestrzeni stanów są *metody zbiorów uporczywych* (ang. *persistent set methods*). *Zbiorem uporczywym* (ang. *persistent set*) [2] tranzycji w znakowaniu M nazywa się taki zbiór T_P aktywnych tranzycji, że dla każdej możliwej sekwencji $Mt_1M_1t_2M_2\ldots M_{n-1}t_nM_n$, składającej się tylko z tranzycji nie należących do T_P, t_n w M_{n-1} jest niezależny od wszystkich tranzycji należących do T_P (gdzie *tranzycje niezależne* to są tranzycie, które nie mogą aktywować i dezaktywować siebie nawzajem; szczegóły patrz w [2]).

Twierdzenie 1. [2] Eksploracja stanów, przy której w każdym z eksplorowanych stanów są symulowane tylko tranzycie należące do jednego z uporczywych zbiorów, pozwala osiągnąć każde z zakleszczeń, osiągalnych w systemie.

Za najbardziej zaawansowaną metodę z rodziny zbiorów uporczywych jest uważana *metoda zbiorów upartych* (ang. *stubborn set method*) A. Valmariego.

Zbiorem upartym (ang. *stubborn set*) [3], [4], [5] T_S tranzycji sieci Petriego w znakowaniu M jest taki zbiór T_S, że: (1) każda nieaktywna tranzycja w T_S ma takie nieoznakowane miejsce wejściowe p, ze wszystkie wejściowe tranzycie miejsca p należą do T_S; (2) żadna aktywna tranzycja, należąca do T_S, nie ma wspólnych miejsc wejściowych z tranzycją z poza T_S (aktywną lub nieaktywną); (3) T_S zawiera aktywną tranzycję.

Twierdzenie 2. [2] Zbiór wszystkich aktywnych tranzycji, należących do zbioru upartego, jest zbiorem uporczywym.

Definicja zbioru upartego jest konstruktywną, więc z definicji tej i z twierdzenia 2 wynika, ze metoda zbiorów upartych daje możliwość prostego obliczenia zbiorów uporczywych. Poniżej pod skrótem ZGO jest rozumiany zredukowany graf osiągalności, skonstruowany przy pomocy metody zbiorów upartych.

Twierdzenie 3. [3], [4], [5] ZGO zawiera wszystkie zakleszczenia danej sieci osiągalne ze znakowania początkowego.

Jak widać z powyższego, metoda zbiorów upartych pozwala wykryć wszystkie osiągalne zakleszczenia. W niektórych przypadkach jej możliwości okazują się większe. Poniżej są przedstawione niektóre autorskie rezultaty dotyczące wykorzystania metody zbiorów upartych w analizie formalnej.

Twierdzenie 4. [6], [7], [8] Silnie spójna α-sieć jest żywą i bezpieczną wtedy i tylko wtedy, gdy jej ZGO jest grafem silnie spójnym i zawiera więcej niż jeden wierzchołek.

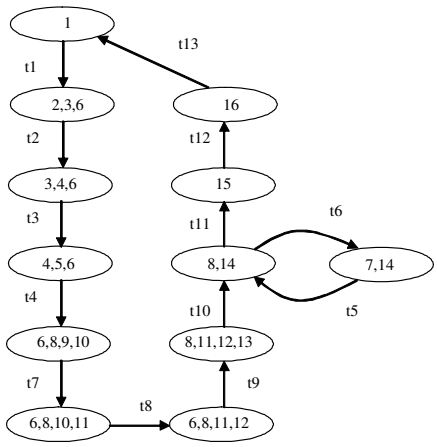
Oznacza to, że metoda zbiorów upartych pozwala sprawdzić, czy dana α-sieć jest poprawnie zbudowana. Sieci należące do klasy α, są typowe dla zastosowań w sterowaniu logicznym. Szczegóły analizy takich sieci są opisane w [6].

Twierdzenie 5. [9] Ograniczona s-sieć jest żywa wtedy i tylko wtedy, gdy jej ZGO jest grafem silnie spójnym i dla każdej tranzycji sieci ZGO zawiera co najmniej jeden odpowiadający jej łuk.

Twierdzenie 6. [7], [9] ZGO ograniczonej sieci Petriego zawiera znakowanie, z którego nie jest osiągalne żadne zakleszczenie, wtedy i tylko wtedy, gdy pełny graf osiągalności sieci zawiera takie znakowanie.

Sprawdzenie takiej właściwości jak istnienie stanów, z których nie jest osiągalne żadne zakleszczenie, może być prawie tak samo przydatne w praktyce, jak i wyznaczenie osiągalnych zakleszczeń. Na przykład, jeśli sieć opisuje zachowanie systemu, który powinien mieć stan terminalny, stwierdzenie, że może zaistnieć sytuacja, w której żaden z takich stanów nie jest osiągalny, oznacza wykrycie błędu polegającego na możliwości zapętlenia się.

Rys. 2 przedstawia zastosowanie metody zbiorów upartych dla sieci z Rys. 1, będącej s-siecią. Pełny graf osiągalności tej sieci zawiera 29 znakowań.



Rys. 2. ZGO dla sieci z Rys. 1
Fig. 2. RRG for the net from Fig. 1

4. Symulacja współbieżna

Innym podejściem do częściowej eksploracji przestrzeni stanów jest symulacja współbieżna. Współbieżna symulacja również pozwala unikać zjawiska przeplotu, ale w zasadniczo inny sposób.

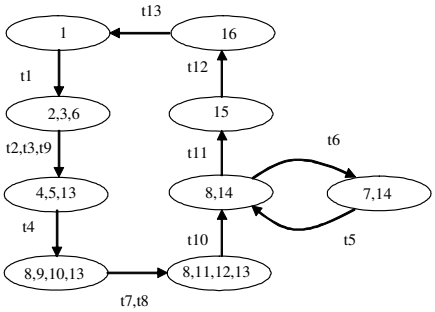
Zasada symulacji *maksymalnie współbieżnej* (ang. *maximally concurrent*) została przedstawione w pracy [10]: „*Zawsze wybieraj maksymalny zbiór niezależnych tranzycji*". Niestety, taka metoda nie zawsze wykazuje wszystkie osiągalne zakleszczenia. Zaawansowany algorytm współbieżnej symulacji został przedstawiony w pracy [10]. Algorytm ten wykrywa wszystkie osiągalne zakleszczenia, ale może być zastosowany tylko dla takich sieci, które mogą być pokryte przez składowe automatowe.

Poniższe twierdzenie opisuje inny sposób współbieżnej symulacji sieci, który to sposób pozwala znaleźć wszystkie osiągalne zakleszczenia ograniczonej sieci o dowolnej strukturze.

Twierdzenie 7. [12] Niech dla danej sieci Petriego w znakowaniu M $U = \{T_{P1}, T_{P2}, \ldots, T_{Pn}\}$ jest zbiorem parami rozłącznych zbiorów uporczywych. Symulacja w każdym z eksplorowanych znakowań każdego z kroków $\Delta_k = (t_1^k, t_2^k, \ldots, t_n^k)$, gdzie $t_i^k \in T_{Pi}$, pozwala osiągnąć każde z osiągalnych zakleszczeń danej sieci.

Z twierdzenia 7 wynika miedzy innymi, że maksymalnie współbieżna symulacja pozwala wykrywać wszystkie osiągalne zakleszczenia w ograniczonych EFC-sieciach, ponieważ w takich sieciach dla każdego znakowania, w którym są aktywne tranzycje, istnieją takie parami rozłączne zbiory uporczywe, że w każdym zbiorze aktywnych niezależnych tranzycji dokładnie jedna tranzycja będzie należeć do jednego z tych zbiorów uporczywych.

Rys. 4 przedstawia graf, który może być skonstruowany dla sieci, przedstawionej na Rys. 1, przy pomocy symulacji opisanej przez twierdzenie 7.



Rys. 4. Graf skonstruowany dla sieci z Rys. 1 zgodnie z procedurą, opisaną w twierdzeniu 7
Fig. 4. A graph constructed for the net from Fig. 1 according to the method described by Theorem 7

5. Wyniki eksperymentów

W ramach pracy magisterskiej [13], której promotorem był autor, zostały zaimplementowane metody upartych zbiorów, symulacji współbieżnej oraz pełnej eksploracji przestrzeni stanów. Zostały one zastosowane do szeregu sieci Petriego. Wyniki tych eksperymentów przedstawione są w tabeli 1. Kolumna „Parametry" przedstawia ilość miejsc × ilość tranzycji sieci. Trzy ostatnie kolumny podają ilość znakowań w grafach osiągalności, skonstruowanych przez algorytmy pełnej eksploracji przestrzeni stanów, upartych zbiorów i symulacji współbieżnej (wg. twierdzenia 7), odpowiednio.

Tab. 1. Porównanie ilości znakowań, zbadanych przez opisane metody
Tab. 1. Comparison of number of markings explored by the described methods

№	Parametry	Osiągalne znakowania	Uparte zbiory	Symulacja współbieżna
1	5×3	7	7	7
2	9×8	14	7	6
3	11×7	36	12	6
4	16×13	78	14	8
5	19×12	53	48	50
6	26×26	514	72	28
7	10×10	17	9	8
8	9×8	13	7	6
9	20×18	19	15	14
10	30×28	28	25	24
11	30×20	21	18	16

6. Wnioski

Zarówno z badań teoretycznych, jak i eksperymentalnych, wynika, że obydwie metody okazują się skuteczne dla tego samego rodzaju sieci. Są to sieci duże, o znacznym stopniu rozgałęzienia i znacznym stopniu niezależności współbieżnych procesów. Spośród takich sieci zdarzają się przykłady, z którymi jedna z tych metod radzi sobie znacznie lepiej od drugiej. W najgorszym przypadku jednak obydwie metody są wykładnicze czasowo i pamięciowo. Przy czym łatwo można podać przykład, dla którego metoda współbieżnej symulacji okazuje się wykładnicza pamięciowo, metoda zbiorów upartych zaś – liniowa; odwrotny przykład najprawdopodobniej nie istnieje. Dla sieci, dla których metoda zbiorów upartych okazuje się mało skuteczna, symulacja współbieżna z reguły pozwala nieznacznie dodatkowo zredukować przestrzeń badanych stanów, ale stosowanie współbieżnej symulacji dla takich sieci raczej się nie opłaca ze względu na większą czasochłonność tej metody.

Im mniej sieć ma konfliktów, tym lepiej metoda symulacji współbieżnej redukuje przestrzeń stanów w porównaniu z metodą zbiorów upartych. Z tego wynika, że metoda symulacji współbieżnej okazuje się najskuteczniejszej dla sieci, należących do klasy *grafów synchronizacji* [14] (sieć № 4 z przedstawionego zestawu eksperymentów).

Istotną zaletą symulacji współbieżnej jest to, że nie występuje w niej tak zwany *problem ignorowania* (ang. *ignoring problem*), stanowiący główną przeszkodę w zastosowaniu podejścia uporczywych zbiorów do rozwiązywania zadań analizy sieci innych, niż wykrywanie zakleszczeń [4], [5]. Problem ten polega na możliwości stałego ignorowania aktywnych tranzycji, nienależących do konstruowanych uporczywych zbiorów. Dlatego też dalszym etapem badań będzie analiza przydatności symulacji współbieżnej do sprawdzania innych właściwości współbieżnych systemów dyskretnych. Są podstawy przypuszczać, że redukcja przestrzeni stanów, dokonywana przy pomocy tej metody, zachowuje żywotność sieci, pojedynczych tranzycji i miejsc, oraz *livelocks* (terminalne silnie spójne składowe grafów osiągalności).

7. Literatura

[1] M. Adamski, M. Węgrzyn, and P. Wolański: A VHDL based approach to logic controllers design. Proc. of Int. Conference PDS. 1998, pp. 9-16.
[2] P. Godefroid: Partial-Order Methods for the Verification of Concurrent Systems: An Approach to the State-Explosion Problem. LNCS. Springer, Vol. 1032, NY, 1996.
[3] A. Valmari: Stubborn Sets for Reduced State Space Generation. Advances in Petri Nets 1990. LNCS. Springer,Vol. 483, Berlin, 1991, p. 491-515,.
[4] A. Valmari: State of the Art Report: Stubborn Sets. Petri Net Newsletter. April 1994, 6-14.
[5] A. Valmari: The State Explosion Problem. Lectures on Petri Nets I: Basic Models. LNCS. Springer, Vol. 1491, 1998, p. 429-528.
[6] A. Karatkevich: Properties and Analysis of α-nets. Informatyka teoretyczna i stosowana. Vol. 5, No. 8, 2003, pp. 53-64.
[7] A. Karatkevich: Dynamic Analysis of Petri Net-Based Discrete Systems. LNCIS 356. Springer Verlag, Berlin Heidelberd, 2007.
[8] A. Karatkevich, M. Adamski and M. Węgrzyn: Rapid Correctness Analysis for Sequential Function Chart. 45. Internationales Wissenschaftliches Kolloquium IWK'2000. Ilmenau, Deutschland, 2000, pp. 679-684.
[9] A. Karatkevich: To Behavior Analysis of a Class of Petri Nets. 27th IFAC/IFIP/IEEE Workshop on Real-Time Programming WRTP'03. Elsevier Ltd, Oxford, 2003, pp. 33-38.
[10] R. Janicki, P. E. Lauer, M. Koutny, R. Devillers: Concurrent and Maximally Concurrent Evolution of Non-Sequential Systems. Theoretical Computer Science, 43, 1986, 213-238.
[11] R. Janicki, M. Koutny: Using Optimal Simulations to Reduce Reachability Graphs. Proceedings of the 2nd International Conference CAV'90. LNCS. Springer, Vol. 531, London, 1991, 166-175.
[12] A. Karatkevich, Analysis of Parallel Discrete Systems: Persistent Sets and Concurrent Simulation. III konferencja naukowa "Informatyka: sztuka czy rzemiosło" KNWS'06. PAK, No. 6bis, 2006. pp. 20-22.
[13] P. Wasielewski: System analizy sieci Petriego z wykorzystaniem symulacji współbieżnej. Praca dyplomowa. Uniwersytet Zielonogórski, 2008.
[14] M.Szpyrka: Sieci Petriego w modelowaniu i analizie systemów współbieżnych. WNT, Warszawa, 2008.

Artykuł recenzowany