Monika WIŚNIEWSKA, Remigiusz WIŚNIEWSKI, Marian ADAMSKI
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI

Usage of Hypergraph Theory in Decomposition of Concurrent Automata

Mgr inż. Monika WIŚNIEWSKA

Ukończyła studia na Wydziale Elektrycznym Uniwersytetu Zielonogórskiego, o specjalności Inżynieria Komputerowa. Obroniła pracę magisterską w 2003 r. Od 2004 r. jest słuchaczem studiów doktoranckich, specjalność informatyka. Jej zainteresowania naukowe to analiza systemów dyskretnych z wykorzystaniem hipergrafów.

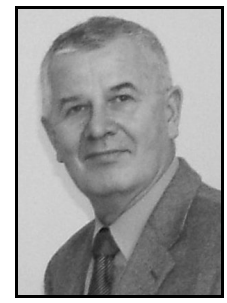
e-mail: M.Wisniewska@weit.uz.zgora.pl

Mgr inż. Remigiusz WIŚNIEWSKI

Mgr inż. Remigiusz Wiśniewski jest absolwentem Uniwersytetu Zielonogórskiego (2003). Ukończył studia o specjalności Inżynieria Komputerowa. W roku 2000 odbył przemysłową praktykę studencką w firmie Aldec Inc. w Stanach Zjednoczonych. Od roku 2003 pracuje jako asystent na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego.

e-mail: R.Wisniewski@iie.uz.zgora.pl

Prof. dr hab. inż. Marian ADAMSKI



Profesor zwyczajny, dyrektor Instytutu Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Zainteresowania badawcze obejmują projektowanie systemów cyfrowych realizowanych w postaci mikrosystemów cyfrowych oraz formalnych metod programowania sterowników logicznych. Członek IEEE, IEE, ACM, PTI oraz PTETiS.

e-mail: M.Adamski@iie.uz.zgora.pl

Abstract

Hypergraphs are useful mathematical tools for a compact representation of relations among local states in the state space of distributed, concurrent control automata (concurrent state machines). Therefore, application of hypergraphs is more efficient and intuitive than traditional solutions. For this reason we propose their application during the design process of reconfigurable logic controllers. It makes it possible to decompose an SFC or a related control interpreted Petri net into parallel or sequentially related modules.

Keywords: hypergraph, transversals, local states of automata, Petri nets.

Równoległa dekompozycja automatów współbieżnych z wykorzystaniem hipergrafów

Streszczenie

Hipergrafy są dogodnym narzędziem matematycznym, umożliwiającym zwartą reprezentację relacji współbieżności lub relacji następstwa w przestrzeni stanów lokalnych cyfrowego automatu współbieżnego. Z tego względu zaproponowano ich wykorzystanie w projektowaniu rekonfigurowanego sterownika logicznego. Hipergraf pozwala w przejrzysty sposób opisywać nie tylko relację współbieżności miedzy stanami lokalnymi, lecz także poglądowo przedstawia ich przynależność do tego samego stanu globalnego. Ułatwia to dekompozycję diagramu SFC lub równoważnej mu interpretowanej sieci Petriego sterowania, na moduły, na przykład szeregowe lub równoległe. W artykule przedstawiono sposób dekompozycji równoległej cyfrowych układów współbieżnych, opisanych z wykorzystaniem sieci Petriego przeprowadzanej za pośrednictwem dekompozycji hipergrafów. Celem dekompozycji jest podział rekonfigurowanego sterownika logicznego na współbieżne moduły, z których każdy może być optymalizowany i syntezowany z wykorzystaniem klasycznej teorii automatów cyfrowych. Sposób dekompozycji sieci Petriego z wykorzystaniem kolorowania grafu współbieżności lub wyszukiwania pokrycia klikami dopełnienia grafu współbieżności (a tym samym grafu niewspółbieżności, czyli grafu następstwa), jest już znany. Opracowując nową metodę, wzięto pod uwagę fakt, że hipergraf współbieżności miejsc sieci Petriego oprócz informacji o relacji między każdą parą miejsc przekazuje dodatkowe dane o istniejących w nim klikach, odpowiadających wcześniej wyznaczonym stanom globalnym. Metoda dekompozycji równoległej automatów współbieżnych zostanie zilustrowana przykładem. Pokazane zostaną niezbędne kroki, jakie są niezbędne do wykonania podziału sterownika logicznego z wykorzystaniem hipergrafów.

Słowa kluczowe: hipergraf, transwersala, stany lokalne automatu, sieć Petriego.

1. Introduction

The size of the logic devices grows up very fast nowadays. Therefore the prototyping process consumes more and more time. One of the most important stages of the design process is the logic synthesis. It takes not only a lot of time but – which is even more important – huge CPU and memory resources. Current synthesis methods are based on the usage of traditional graph theory [1]. Such a solution was perfect for small devices where the complexity of the design is relatively low. However, when the size of contemporary devices attains millions of gates, many algorithms that are based on graphs have to be modified.

A solution to this problem can be a hypergraph which is much more efficient than any classic undirected graph [1, 2, 3]. The usage of hypergraph theory permits to store and reduce information about the design needed for further synthesis. Moreover, current analysis methods of the hypergraphs are much faster compared with traditional solutions.

In the article a new decomposition method of concurrent automata is presented. The system is designed as a Petri Net which is decomposed using hypergraph theory. The aim of the method is to divide the control automaton into concurrent state machines. It is important that all of the decomposed modules can be afterwards optimized and synthesized with traditional solutions.

2. Main definitions

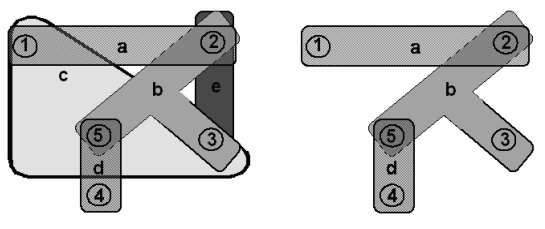
In this section, for the reader's convenience, we recall some definitions related to Petri nets [4, 5] and hypergraphs [1, 2] we need later to explain the decomposition of the automata into concurrent state machines.

2.1. Hypergraphs

A hypergraph is a generalization of a graph. Its edges - known as hyperedges - can connect any number of vertices [1, 2, 3], while classic graph can connect only two vertices.

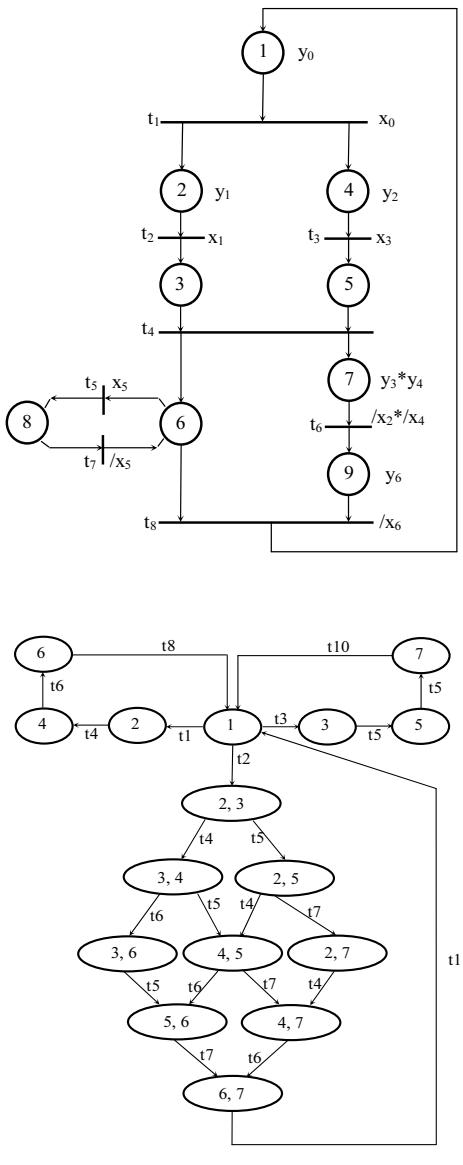
Formally, a hypergraph is a pair (V, E) where V is a set of vertices, and an E is a set of edges. Hyperedges are arbitrary sets of vertices, and can therefore contain an arbitrary number of vertices. Figure Fig. 2 shows the graphical representation of a hypergraph. One of the most popular representations of hypergraph is incidence matrix where vertices are represented by columns and hyperedges by rows of the matrix.

The minimal number of hyperedges that are connected to all vertices is called as a hyper-graph transversal (Fig. 1). There are several methods to compute transversal of hypergraph. For example, it can be achieved through the reduction of the incidence matrix [1].



Rys. 1. Hipergraf H oraz jego transwersala
Fig. 1. Hypergraph H and its transversal

The complement of a hypergraph *H=(V,E)* is a hypergraph H*=(V,E*), in which E* is a set of all subsets of V that are not in *E*. It means that H* contains all hyperedges that are not present in H.



Rys. 2. Sieć Petriego PN_1 oraz jej graf znakowań
Fig. 2. Petri Net PN_1 and its reachability graph

2.2. Petri nets

A Petri net is one of the representations of discrete distributed systems. It is a labeled oriented graph which consists of places, transitions and directed arcs. A place is marked with a circle while a transition is represented by a line. Such a structure permits for a simple description of describe concurrent processes. To analyze and present a Petri net in a more complex and intuitive form, a reachability graph is created. It represents the reachability of the states in the Petri net. The reachability graph can be used to find erroneous states or states that cannot be reached [6, 7]. An exemplary Petri net, PN_1, and its reachability graph RG_1 are shown in Fig. 2.

3. Main idea of proposed method

3.1. Traditional method of concurrent automata decomposition

In order to present the new idea of Petri net decomposition, the traditional method ought to be presented. The algorithm of control automaton decomposition based on the graph theory can be divided into following steps:

1. ***Creation of the macronet for the initial Petri Net***. In this stage the macronet is created. It is performed using well-known solutions [5, 8] and it is not under the scope of this article.

2. ***Formation of the reachability graph G***. Based on the macronet prepared in the previous step, the reachability graph is created. It contains all states of the Petri Net.

3. ***Formation of the relations of the concurrency***. To perform this stage the cliques in the reachability graph G ought to be found. Further analysis of the graph allows us to find the relations between states in the macronet.

4. ***Computation of the subnets of automata that cover all macroplaces in the macronet***. This step is performed by finding the minimum cover of the reachability graph G.

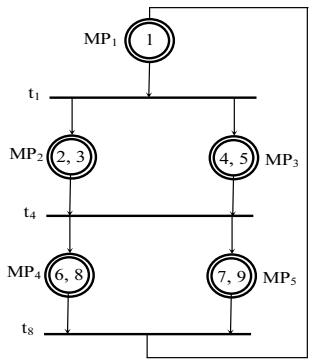
5. ***Replacement of the macroplaces by adequate concurrent state machines***. Finally initial Petri Net can be decomposed into concurrent control automata based on the results achieved in step 4.

The idea of macronet creation and its decomposition was proposed by Andre [5]. It is based on the classical graph theory. In this article the new method of decomposition of macronet is proposed. Instead of creation of the graph, the reachability hypergraph is created. Therefore the hypergraph theory is used to compute the subnets of the initial Petri Net.

3.2. New method of concurrent automata decomposition

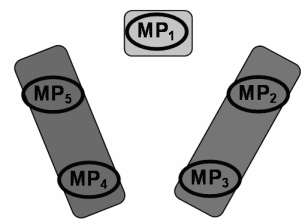
The process of the Petri Net decomposition using hypergraphs can be divided into the following steps:

1. ***Creation of the macronet for the initial Petri Net***. This step is similar to the one shown in the traditional method of decomposition. An exemplary macronet MN_1 for the Petri Net PN_1 is shown in Fig. 3.



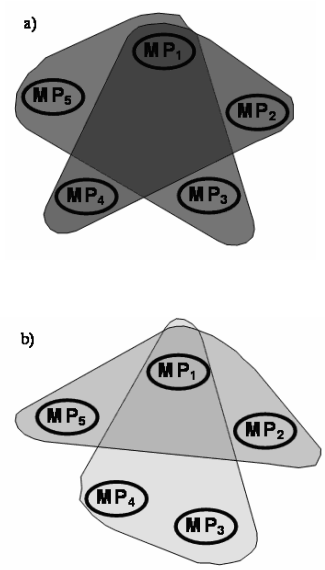
Rys. 3. Makrosieć MN_1
Fig. 3. Macronet MN_1

2. ***Formation of the reachability hypergraph H***. Instead of the graph, reachability hypergraph for the macronet is created. Figure 4 illustrates the reachability hypergraph HR_1 for the macronet MN_1.



Rys. 4. Hipergraf znakowań HR_1 dla makrosieci MN_1
Fig. 4. Reachability hypergraph HR_1 of macronet MN_1

3. ***Computation of the complement H* of the hypergraph and formation of its transversal***. The result of this process is the same as hypergraph coloring. However very often computation of the complement of hypergraph and further finding its transversal is faster and less complex than hypergraph coloring [2].



Rys. 5. Transwersale dopełnienia hipergrafu
Fig. 5. Transversals of hypergraph complement

In our example the complement of hypergraph HR_1 contains four hyperedges: $HR^*=\{a_1,a_2,a_3,a_4\}$, where $a_1=\{MP_1,MP_2,MP_4\}$, $a_2=\{MP_1,MP_2,MP_5\}$, $a_3=\{MP_1,MP_3,MP_4\}$ and $a_4=\{MP_1,MP_3,MP_5\}$. For such a hypergraph two transversals can be found as it is shown in Fig. 5. Both indicates that the minimal number of hyperedges that are connected to all vertices of the hypergraph HR* is equal to two. This means that Petri net PN_1 can be decomposed into the two concurrent automata.

Let's also point out here that there is no need to compute any cliques, which is required in traditional solutions.

4. ***Replacement of the macroplaces by adequate concurrent state machines***. Finally the automata can be decomposed according to the transversal of the hypergraph H*.

4. Conclusion

In the article a new decomposition method of concurrent automata is presented. The main advantage of the presented method is the availability of hypergraph theory usage instead of graph one. Hypergraph contains all information about the relations between hyperedges while in graph all cliques have to be found. Furthermore, the process of finding concurrent automata can be more effective because of hypergraph complement usage. Very often computation of the transversal of hypergraph complement is less complex than its coloring. Let us point out that the results of both solutions, either transversal of hypergraph, complement and hypergraph coloring are here the same.

5. Acknowledgement

This work has been partially supported by the Integrated Regional Operational Programme (Measure 2.6: Regional innovation strategies and the transfer of knowledge) co-financed from the European Social Fund.

6. References

[1] G. De Micheli: Synteza i optymalizacja układów cyfrowych. Wydawnictwo Naukowo-Techniczne, Warszawa 1998.
[2] C. Berge: Graphs and Hypergraph. North-Hols.r Mathematical Library, Amsterdam 1976.
[3] T. Łuba: Synteza układów logicznych. Wyższa Szkoła Informatyki Stosowanej i Zarządzania, Warszawa 2000.
[4] Z. Banaszak, J. Kuś, M. Adamski: Sieć Petriego. Modelowanie, sterowanie i synteza systemów dyskretnych. Wydawnictwo Wyższej Szkoły Inżynierskiej, Zielona Góra 1993.
[5] R. David, H. Alla: Petri Nets & Grafcet. Tools for modelling discrete event systems. Prentice Hall, New York, 1992.
[6] M. Adamski, A. Karatkievich, M. Węgrzyn (ed.): Design of Embeded Control Systems. Springer Science (USA), 2005.
[7] K. Bilinski, M. Adamski, J.M. Saul, E.L. Dagless: Petri-net-based algorithms for parallel-controller synthesis. IEE Proceedings - Computers and Digital Techniques, 1994, Vol. 141, no 6, s. 405-412.
[8] M. Adamski, M. Chodań: Modelowanie układów sterowania dyskretnego z wykorzystaniem sieci SFC. Wydawnictwo Politechniki Zielonogórskiej, Zielona Góra, 2000.

Artykuł recenzowany