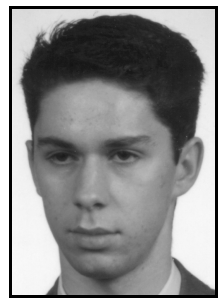
Kamil MIELCAREK
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI

Rozpoznawanie grafów doskonałych na potrzeby automatycznej syntezy układów cyfrowych

Mgr inż. Kamil MIELCAREK

Kamil Mielcarek jest pracownikiem Instytutu Informatyki i Elektroniki Uniwersytetu Zielonogórskiego od czterech lat. Zajmuje się zastosowaniem grafów doskonałych w procesach automatycznej syntezy i optymalizacji układów cyfrowych. Jednocześnie zajmuje się zagadnieniami systemów sieciowych opartych o system OpenBSD jak i ich bezpieczeństwem. Zajmował się też systemami czasu rzeczywistego opartymi o systemy BSD i Linux.



e-mail: K.Mielcarek@iie.uz.zgora.pl

Streszczenie

Niniejszy artykuł ma na celu wskazanie potencjalnej użyteczności algorytmów do rozpoznawania grafów doskonałych. Zagadnienia prezentowane poniżej dają się rozwiązać w czasie wielomianowym, gdzie dla grafów w ogólności problem złożoności należy do klasy problemów NP-zupełnych. Zastosowanie grafów doskonałych jako pośredniego modelu formalnego jest o tyle użyteczne, że znacząca większość zależności, opisujących rzeczywiste układy sekwencyjne, daje w wyniku grafy należące do podklasy grafów doskonałych. Badania w tej dziedzinie są prowadzone w wielu ośrodkach światowych mają na celu przy-spieszenie procesów automatycznej analizy i syntezy układów dyskretnych. Referat jest związany z wykorzystaniem sieci Petriego do opisu układów cyfrowych, a w szczególności rekonfigurowalnych sterowników logicznych. Pokazano przykład analizy dyskretnej przestrzeni stanów lokalnych automatu współbieżnego, modelującego program sterownika logicznego przedstawionego w języku SFC. Do analizy wykorzystano metody badania grafów doskonałych. Pokazano sposób sprawdzenia, czy graf zgodności (niezgodności) jest grafem doskonałym.

Słowa kluczowe: grafy doskonałe, algorytmy rozpoznawania grafów doskonałych, teoria grafów, automatyczna synteza sterowników.

Recognizing of perfect graphs for automatic synthesis of integrated digital circuits

Abstract

This paper should to point out potential strength of perfect graph algorithms for automated synthesis of digital circuits. Typically known problems are NP-complete but using perfect graphs complexity is decreasing to polynomial. Studies in this matter shows that plenty of dependencies describing real sequential circuits can be described using perfect graphs. There shown the analysis of digital controller described in SFC. In analysis was used methods for testing perfect graphs.

Keywords: perfect graphs, perfect graphs recognizing algorithms, graph theory, automatic synthesis of digital controllers.

1. Wstęp

Współczesne metody projektowania układów bardzo szeroko stosują formalne modele pośrednie, do wewnętrznej i zewnętrznej reprezentacji struktur układów logicznych. Powszechnie stosowane są rozwiązania na bazie grafów czy sieci Petriego. Prace badawcze ukazują, że większość grafów otrzymywanych podczas automatycznej syntezy z wykorzystaniem modeli opartych na sieciach Petriego lub grafach stanów należy do specyficznej klasy grafów - grafów doskonałych [3]. Algorytmy używane do analizy grafów, często będących algorytmami heurystycznymi, dają wyniki zbliżone do optymalnych. Uzyskanie wyników dokładnych prowadzi do zastosowania algorytmów o złożoności należących do klasy algorytmów NP-zupełnych, co wiąże się z ogromnymi nakładami czasowymi (o ile jest to technicznie możliwe). Wykonanie tych samych operacji z użyciem algorytmów operujących na grafach doskonałych daje możliwość osiągnięcia tych samych wyników z zastosowaniem algorytmów o złożoności wielomianowej. Konieczne jest wypracowanie metod rozpoznawania grafów doskonałych oraz ich wykorzystanie w sposób umożliwiający dalszą optymalizację wyników.

2. Grafy doskonałe i ich rozpoznanie

Cechy grafu doskonałego

Bogactwo podklas grafów doskonałych pozwala wykorzystać różne podklasy w różnych miejscach lub sytuacjach automatycznego procesu syntezy. Rozpoznawanie grafów doskonałych do konkretnego zastosowania należy poprzedzić wnikliwymi badaniami, które z potencjalnie możliwych algorytmów należy zastosować. Specyfika algorytmów oraz potencjalne użycie wcześniej otrzymanych wyników w dalszych częściach procesów może dodatkowo podnieść efektywność stosowanych algorytmów. Z uwagi na potencjalnie najszersze zastosowania praktyczne przedstawiony zostanie sposób rozpoznania podklas grafów doskonałych, takich jak grafy trójkątne, porównywalności oraz grafów interwałowe. Na wstępnie należy jednak pamiętać, co odróżnia graf doskonały od innych grafów. Można wykazać, że dla dowolnego grafu zachodzą następujące zależności [5, 6]:

$$\omega(G) \le \chi(G) \tag{1}$$

$$\alpha(G) \le \kappa(G) \tag{2}$$

czyli najmniejsza liczba kolorów potrzebna do właściwego pokolorowania wierzchołków grafu G jest równa lub większa od największego pełnego podgrafu w grafie G. Jednocześnie najmniejsza liczba pełnych podgrafów potrzebnych do pokrycia wierzchołków grafu G jest równa lub większa niż rozmiar największego zbioru stabilnego w G [6].

Nierówności (1) i (2) powodują, że problemy kolorowania właściwego oraz minimalnego pokrycia klikami stają się problemami o wysokim stopniu złożoności obliczeniowej. Jeżeli zamiast nierówności (1) i (2) dla rozpatrywanych grafów wystąpią równości, to wyżej wymienione problemy mogą być rozwiązane w czasie wielomianowym (dla grafów doskonałych).

Nieskierowany graf G jest grafem doskonałym (ang. perfect graph), jeśli spełnia trzy ekwiwalentne własności (Lovász 1972):

$$\omega(G_A) = \chi(G_A) \qquad (\text{dlą } A \subseteq V) \tag{3}$$

$$\alpha(G_A) = \kappa(G_A) \qquad (\text{dlą } A \subseteq V) \tag{4}$$

$$\omega(G_A)\alpha(G_A) \ge |A| \quad (\text{dlą } A \subseteq V) \tag{5}$$

dla nieskierowanego grafu G = (V,E). Jednocześnie graf G jest grafem doskonałym, wtedy i tylko wtedy, gdy jego dopełnienie jest grafem doskonałym.

Grafy trójkątne

Rozpoznawanie grafów trójkątnych można przeprowadzić dwuetapowo, poprzez obliczenie doskonałego schematu eliminacji wierzchołków (*ang*. ***P**erfect **V**ertex **E**limination **S**cheme* - ***PVES***) lub schematem doskonałym (*ang*. *perfect scheme*) metodą przeszukiwania leksykograficznego grafu w szerz, a następnie sprawdzić czy wygenerowany schemat jest schematem doskonałym.

Rozpatrując wejściowy graf $G = (V, E)$ będący grafem nieskierowanym oraz $\sigma = \{v_1, v_2, \ldots, v_n\}$ będącym pewnym uporządkowaniem wierzchołków możemy mówić o doskonałym schemacie eliminacji wierzchołków, jeśli sąsiedztwo węzła v_i jest podstawą

do wyznaczenia podgrafu pełnego. Inaczej mówiąc każdy zbiór $X_i = \{v_j \in \mathrm{Adj}(v_i) \mid j > i\}$ jest grafem pełnym.

Grafy trójkątne charakteryzują się występowaniem takiego węzła. Jeżeli graf nie jest kliką to ma on dwa takie węzły. Drugim etapem (powiązanym z poprzednim) jest przeprowadzenie weryfikacji, czy wygenerowane przyporządkowanie σ, jest *PVES*. Do tego celu używana jest lista A(u) w której gromadzone są wierzchołki, które należy sprawdzić czy są incydentne z wierzchołkiem u. Samo sprawdzenie jest opóźnione aż do momentu gdy $u = \sigma(i)$. Tej techniki używa się dlatego, że w $\sigma^{-1}(v)$ iteracji nie następuje przeszukanie węzłów sąsiadujących z u. Powoduje to, że większa część algorytmu może być implementowane w jednym przebiegu przeszukującym węzły sąsiadujące z v. Instrukcja skoku zostanie wykonana j-1 razy, gdzie j oznacza liczbę komponentów grafu G.

Etap 1
przypisz etykietę ∅ do każdego wierzchołka;
dla $i \leftarrow n$ do 1 krok (-1) wykonaj
{
  wybór: wziąć nieponumerowany wierzchołek
    v o największej etykiecie;
  $\sigma(i) \leftarrow$ v; komentarz: przypisuje do v numer i
  odśwież: dla każdego nieponumerowanego
    wierzchołka $w \in$ Adj(v) wykonaj dodaj
    i do etykieta(w);
}
Etap 2
dla wszystkich wierzchołków v zrób: A(v) $\leftarrow \emptyset$;
dla $i \leftarrow 1$ do n wykonaj
{
  $v \leftarrow \sigma(i)$;
  $X \leftarrow \{x \in \mathrm{Adj}(v) \mid \sigma^{-1}(v) < \sigma^{-1}(x)\}$;
  jeśli $X = \emptyset$ to wtedy skocz do linii 8;
  $u \leftarrow \sigma(\min\{\sigma^{-1}(x) \mid x \in X\})$;
  połącz X - {u} z A(u);
  jeśli A(v) - Adj(v) $\neq \emptyset$ to wtedy zwróć „fałsz";
}

Rys. 1. Algorytm rozpoznawania grafów trójkątnych; [6]
Fig. 1. A triangulated graph recognizing algorithm; [6]

Grafy porównywalności

Rozpoznanie przynależności grafu do podklasy grafów porównywalności przeprowadza się przez próbę zorientowania tranzytywnego wejściowego grafu. Dokonuje się tego przez przypisanie wymuszeń orientacji grafu nieskierowanego, według pewnej relacji. Dokładna definicja wymuszeń, oznaczana jako relacja Γ, jest zapisywana jako

$$ab\ \Gamma\ a'b' \Leftrightarrow \{a = a' \text{ oraz } bb' \notin E \quad \text{lub} \quad b = b' \text{ oraz } aa' \notin E \tag{6}$$

Istnieje zależność powstała przez połączenie w ciąg krawędzi w relacji Γ między sobą, nazywana *klasą implikacji*. Mówimy więc, że krawędzie *ab* oraz *cd* należą do tej samej klasy implikacji tylko gdy istnieje sekwencja

$$ab = a_0b_0\ \Gamma\ a_1b_1\ \Gamma\ \ldots\ \Gamma\ a_kb_k = cd \quad \text{dla } k \geq 0 \tag{7}$$

Tego rodzaju sekwencja nazywana jest łańcuchem Γ z *ab* do *cd*.

Inicjuj: $i \leftarrow 1$; $E_i \leftarrow E$; $F \leftarrow \emptyset$
Arbitralnie wybierz krawędź $x_iy_i \in E_i$
Oblicz klasę implikacji B_i zbioru E_i zawierającą x_iy_i
jeśli $B_i \cap B_i^{-1} = \emptyset$ to wtedy
  dodaj B_i do F
w przeciwnym razie
  zwróć: „G nie jest grafem porównywalności", STOP
Zdefiniuj: $E_{i+1} \leftarrow E_i - \{B_i \cup B_i^{-1}\}$
jeśli $E_{i+1} = \emptyset$ to wtedy
  $k \leftarrow i$
  zwróć F, STOP
w przeciwnym razie
  $i \leftarrow i + 1$
  idź do arbitralnego wyboru krawędzi

Rys. 2. Algorytm rozpoznawania grafów porównywalności; [6]
Fig. 2. A comparability graph recognizing algorithm; [6]

Jeżeli podejmie się próbę zorientowania tranzytywnego grafu zaczynając od krawędzi np. *ab* a w wyniku stosowania relacji Γ otrzyma się wymuszenie orientacji krawędzi *ba* można stwierdzić, że graf nie należy do podklasy grafów porównywalności.

Grafy interwałowe

Rozpoznawanie grafów interwałowych można oprzeć na założeniu, że graf jest grafem interwałowym, jeśli dopełnienie grafu trójkątnego jest grafem porównywalności [6]. Możliwe jest użycie prezentowanych wcześniej algorytmów. Należy pamiętać, że rozpoznanie przynależności do grafów interwałowych będzie prawdopodobnie używane łącznie z rozpoznawaniem dwóch powyższych podklas grafów doskonałych. Istnieją jeszcze inne algorytmy pozwalające w szybki sposób rozpoznać grafy interwałowe [8].

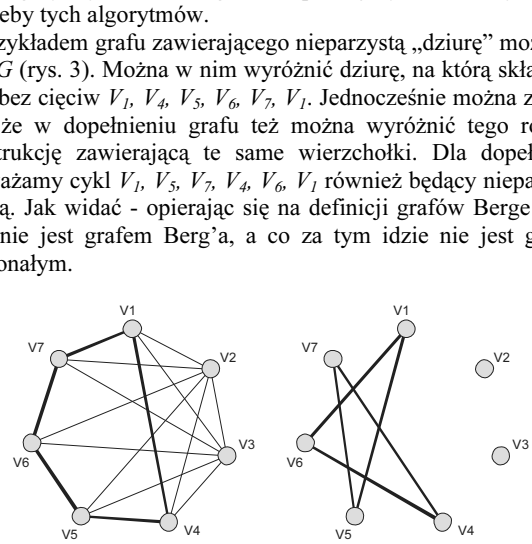
Rozpoznawanie przez wyszukiwanie nieparzystych dziur

Nowsze algorytmy rozpoznawania grafów doskonałych opierają się o metodę poszukiwania nieparzystych dziur (ang. odd-hole) w grafach prostych skończonych [4]. Dowiedziono, że graf jest grafem doskonałym wtedy i tylko wtedy, gdy jest grafem Berge'a. Natomiast graf Berge'a można rozpoznać, gdy ani graf ani jego dopełnienie nie zawierają nieparzystych dziur.

Dziurą nazywa się cykl bez cięciw o długości przynajmniej 4. Jeśli liczba krawędzi dziury jest nieparzysta wtedy mówi się o nieparzystej dziurze, a w przeciwnym wypadku o dziurze parzystej. Jednocześnie brak dziur w grafie powoduje, że graf jest albo prosty albo zawiera podgraf 2-gwiezdny (ang. 2-star) lub dwudzielny, (ang. 2-join). To stwierdzenie jest podłożem wyjścia do algorytmów rozpoznawania, czy graf przynależy do rodziny grafów doskonałych, poprzez rozłożenie grafu G na m podgrafów i stwierdzenie czy każdy z G_i gdzie $i=1,2,\ldots,m$ taki prostszy podgraf jest grafem bez nieparzystych dziur. Należy wspomnieć, że wyszukanie 2-gwiazdy jest problemem, który można rozwiązać w czasie wielomianowym. Rozpoznanie 2-gwiazdy opartej o wierzchołku u i v, w czasie wielomianowym polega na sprawdzeniu czy zostaną rozłączone niesąsiadujące wierzchołki x i y, przez usunięcie wszystkich sąsiadów u i v z wyjątkiem x i y. Natomiast wielomianowy algorytm wyszukiwania podgrafu 2-dzielnego można znaleźć w [10, 11].

Autorzy [4] podają metodę rozpoznawania grafów doskonałych z użyciem dekompozycji 2-gwiazd oraz dekompozycji 2-dzielnej oraz algorytmy tworzenia grafów spełniających inne kryteria na potrzeby tych algorytmów.

Przykładem grafu zawierającego nieparzystą „dziurę" może być graf G (rys. 3). Można w nim wyróżnić dziurę, na którą składa się cykl bez cięciw $V_1, V_4, V_5, V_6, V_7, V_1$. Jednocześnie można zauważyć, że w dopełnieniu grafu też można wyróżnić tego rodzaju konstrukcję zawierającą te same wierzchołki. Dla dopełnienia zauważamy cykl $V_1, V_5, V_7, V_4, V_6, V_1$ również będący nieparzystą dziurą. Jak widać - opierając się na definicji grafów Berge'a, ten graf nie jest grafem Berg'a, a co za tym idzie nie jest grafem doskonałym.

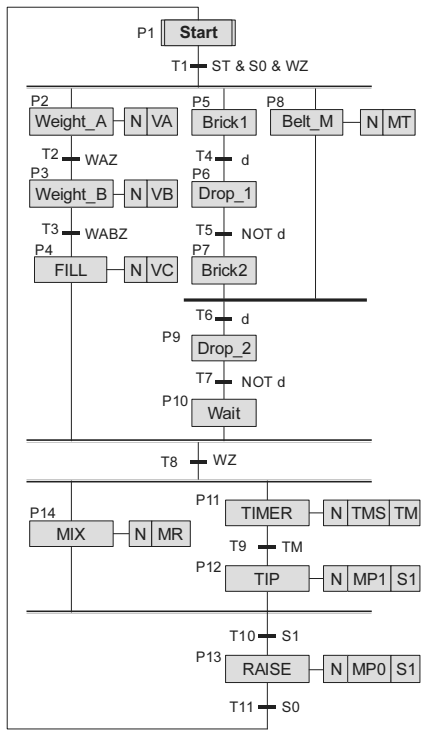


Rys. 3. Graf G (a) oraz jego dopełnienie (b) zawierające nieparzystą „dziurę"
Fig. 3. Graph G (a) and complement (b) containing „odd-hole"

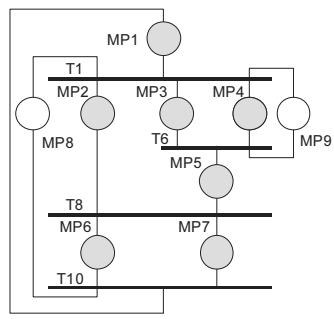
3. Zastosowanie (przykład)

Przedstawione metody, w połączeniu z metodami opisanymi w innych pracach pozwalają na przeprowadzenie uproszczonej przykładowej analizy. Niech dany jest opis układu w postaci grafu

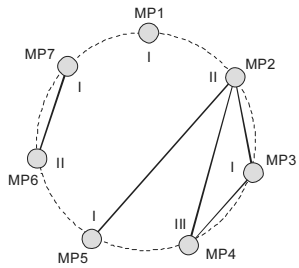
SFC. Za przykład niech posłuży sieć pochodząca z załącznika standardu IEC 1131-3 [7]. Oryginalny sposób interpretacji układowej sterownika rekonfiguralnego przedstawiono w pracach [1, 2].



Rys. 4. Opis układu w postaci SFC
Fig. 4. SFC description of the circuit



Rys. 5. Makrosieć Petriego odpowiadająca modelowi SFC z rys. 4
Fig. 5. Petrie macro-net eqivalent to SFC model from fig. 4



Rys. 6. Graf zgodności - modelu z rys. 4
Fig. 6. Compatibility graph of the model from fig. 4

Po wyznaczeniu relacji współbieżności między miejscami makrosieci, np. analizując graf znakowań makrosieci, uzyskano graf zgodności (współbieżności) między miejscami. Na podstawie analizy stwierdzono, że graf ten jest
- złożony z trzech składowych spójności,
- każda z nich odpowiada składowej automatowej,
- składowe są grafami doskonałymi,
- przynależy do podklasy grafów trójkątnych*,
- przynależy do podklasy grafów porównywalności*.

* (w tym przypadku)

W związku z tym jego kolorowanie daje się wykonać w czasie wielomianowym. W rezultacie kolorowania otrzyma się trzy składowe podsieci automatowe:

I: $\{MP_1, MP_3, MP_5, MP_7\} = \{P_1, P_5, P_6, P_7, P_9, P_{10}, P_{11}, P_{12}, P_{13}\}$
II: $\{MP_2, MP_6\} = \{P_2, P_3, P_4, P_{14}, P_{15}\}$
III: $\{MP_4\} = \{P_8, P_{16}\}$

4. Wnioski

Na podstawie analizy większej liczby sieci Petriego, opisującej rzeczywiste układy sterowania można wysnuć przypuszczenie, że relacja uporządkowania przestrzeni stanów lokalnych takiej sieci jest opisana grafem doskonałym. Wniosek ten może potwierdzać pośrednio praca [3], w której przeprowadzono znaczną liczbę testów na grafach opisujących rzeczywiste układy cyfrowe. Wyniki prac eksperymentalnych w tej dziedzinie wskazują, że znacząca większość rzeczywistych układów cyfrowych opisanych grafami należącymi do klasy grafów doskonałych daje się właściwie pokolorować w prosty sposób w czasie wielomianowym.

Badania autora idą w kierunku określenia: 1) klas sieci Petriego, dla których relacja współbieżności między równocześnie oznakowanymi miejscami jest opisana grafem doskonałym, 2) metod rozpoznawania, czy relacja współbieżności w przestrzeni stanów lokalnych dowolnego sterownika logicznego jest opisana grafem doskonałym.

Algorytmy rozpoznawania grafów trójkątnych, porównywalności i interwałowych zostały zrealizowane praktycznie z wykorzystaniem języka C.

5. Literatura

[1] Adamski M., „Metodologia projektowania reprogramowalnych sterowników logicznych z wykorzystaniem elementów CPLD i FPGA", Reprogramowalne Układy Cyfrowe RUC 98, Szczecin 12-13.03.1998, ss. 15-22, 1998
[2] Adamski M., „Aplication Specific logic Controler for Safety Critical Systems", 1999 IFACTriennial congress, Beijing (Chiny) vol. 0, ss. 519-529
[3] Olivier Coudert, „Exact Coloring of Real-Life Graphs is Easy", Synopsys, Inc. 700 East Middlefield Rd., Moutain View, CA 94043, Proc. of the 34th Design Automation Conference DAC,, Anaheim, CA, USA, June 1997, pp. 121-126
[4] Cornuejols G., Xinming Liu, Vuskovic K., "A polynomial algorithm for recognizing perfect graphs", Proceeding 44th Annual IEEE Symposium, 2003
[5] Giovani De Micheli: „Synteza i optymalizacja układów cyfrowych", Wydawnictwa Naukowo-Techniczne, Warszawa 1998
[6] Martin Charles Golumbic: „Algorithmic Graph Theory and Perfect Graphs", Courant Institute of Mathematical Science, New York University, Academic Press 1980
[7] International Electrotechnical Commission, "Programable controllers. Part 3 - Programming Languages IEC 1131-3", Genewa, 1993
[8] Leuker G. S., "Interval graph algorithms. Ph. D. thesis", Princeton University
[9] Mielcarek K, „Grafy doskonałe jako alternatywa dla automatycznej syntezy i optymalizacji układów cyfrowych", Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim: IV Krajowa Konferencja. Kraków, Polska, 2003, Oprogramowanie Naukowo-Techniczne, 2003
[10] M. Conforti, G. Cornuejols, A. Kapoor, K. Vuskovic, „Even-hole-free-graphs, Part II: Recognition algorithm", Journal of Graph Theory 40 (2002) 238-266
[11] G. Cornuejols, W.H. Cunnigham, „Compositions for perfect graphs", Discrete Mathematics 55 (1985) 245-254

Artykuł recenzowany