II Konferencja Naukowa KNWS'05
"Informatyka- sztuka czy rzemiosło"
15-18 czerwca 2005, Złotniki Lubańskie

MODELOWANIE SIECI PETRIEGO Z WYKORZYSTANIEM RELACYJNEJ BAZY DANYCH

Małgorzata Kołopieńczyk

Instytut Informatyki i Elektroniki, Uniwersytet Zielonogórski
65-246 Zielona Góra, ul. Podgórna 50

e-mail: m.kolopienczyk@iie.uz.zgora.pl

STRESZCZENIE

Celem artykułu jest przedstawienie metody zapisu sieci Petriego w relacyjnej bazie danych. Relacyjna baza danych to obecnie najpopularniejszy sposób gromadzenia danych. Jasno sformułowane relacje pozwalają na przejrzyste opisanie złożonych struktur. Ponadto model relacyjnej bazy danych pozwala na zapisanie w strukturze bazy nawet bardzo rozbudowanych sieci Petriego.

1. WPROWADZENIE

Obserwowany ciągły postęp w elektronice pozwala na implementację coraz to bardziej skomplikowanych i rozbudowanych sterowników logicznych PLC (Programmable Logic Controllers). Wraz z rosnącymi możliwościami technologicznymi wciąż poszukiwane są nowe metody projektowania takich układów, w tym przede wszystkim metody wspierane technikami komputerowymi. Wśród nowoczesnych technik projektowania sterowników logicznych wymienić można np. język Grafcet [5] oraz języki wyspecyfikowane w międzynarodowej normie IEC:1131-3 [4].

W celu weryfikacji poprawności oraz w celu zbadania własności ilościowych i jakościowych projektowanego układu stosowane są formalne metody opisu układu. Mimo wielorakości metod projektowania sterowników, najczęściej stosowanym formalizmem na etapie weryfikacji są sieci Petriego [1, 3, 6]. Sieć Petriego stanowi uogólnioną, pośrednią formę w projektowaniu i badaniu właściwości sterowników.

W niniejszym artykule przedstawiono sposób zamodelowania układu logicznego w języku sieci Petriego przy wykorzystaniu mechanizmów relacyjnej bazy danych. Relacyjne bazy danych to obecnie najpopularniejszy sposób gromadzenia danych dla aplikacji [7, 9]. Wysoko sformatowana postać danych oraz jasno sprecyzowane relacje umożliwiają na przejrzyste opisanie nawet skomplikowanych struktur, umożliwiając jednocześnie na automatyczne

przetwarzanie danych przez programy komputerowe. Wykorzystanie baz danych pozwala także na zwiększenie wydajności przetwarzania - wysoka skalowalność systemów zarządzania bazami danych RDBMS (Relational Database Management Systems) pozwala na zakodowanie nawet bardzo złożonych układów, techniki indeksowania i partycjonowania pozwalają na szybkie przetwarzanie ogromnych ilości danych, dając nieporównywalnie większą wydajność w stosunku do przetwarzania standardowych plików komputerowych oraz niwelując ograniczenia związane z koniecznością zapamiętania całej struktury układu w pamięci operacyjnej.

Zapisanie układu sterowania w relacyjnej bazie danych pozwala ponadto na symulację projektowania układu bezpośrednio przy wykorzystaniu standardowych języków zapytań, w tym przede wszystkim języka SQL [8]. Daje to możliwość badania własności sterownika bezpośrednio przez projektanta stosującego standardowy język zapytań i mającego bezpośredni wpływ na operacje wykonywane na modelu.

2. SIECI PTRIGO

Sieci Petriego [2, 3] umożliwiają badanie zjawisk współbieżnych zachodzących w systemach wzajemnie warunkujących się w czasie i przestrzeni warunków i zdarzeń. Oprócz projektowania i programowania sterowników logicznych, znajdują one zastosowanie także w różnych gałęziach przemysłu, w zadaniach planowania i sterowania przepływem produkcji, w syntezie oprogramowania systemowego itp.

W sieci Petriego zdarzenia i warunki reprezentowane są abstrakcyjnymi symbolami należącymi do dwóch rozdzielnych alfabetów (zbiorów). Zdarzenia odpowiadają przejściom (tranzycjom), warunki miejscom, a związek przeyczynowo-skutkowy między zdarzeniami i warunkami oraz warunkami i zdarzeniami za pomocą relacji przepływu.

Graficznie sieć Petriego reprezentowana jest jako struktura złożona z prostokątów i kółek połączonych łukami (strzałkami). Prostokąty reprezentują przejścia, kółka – miejsca, a łuki relację przepływu między miejscami i przejściami oraz przejściami i miejscami. Miejsca, z których prowadzą łuki z rozpatrywanego przejścia nazywane są miejscami wyjściowymi przejścia. Analogicznie określane są miejsca wejściowe przejścia oraz przejścia wejściowe i wyjściowe dla miejsc. Miejsca wejściowe przejścia przedstawiają warunki – przyczyny zdarzenia określonego tym przejściem. Miejsca wyjściowe przejścia reprezentują warunki – skutki zdarzenia określonego tym przejściem. Wystąpienie (utrzymanie się) pewnego warunku przedstawione jest znakiem (znacznik, marker) wewnątrz miejsca odpowiadającego temu warunkowi.

Dynamika działania modelowanego systemu dyskretnego znajduje odzwierciedlenie w przesuwaniu się znaczników pomiędzy miejscami sieci. Lokalne zdarzenia i związane

z nimi akcje przedstawiane są realizacją (zadziałaniem, zapaleniem) określonego przejścia. Przejście może być zrealizowane (wystąpić), gdy wszystkie jego miejsca wejściowe zawierają, co najmniej po jednym znaku. W kategoriach zdarzeń i warunków potencjalna realizacja przejścia odpowiada spełnieniu warunków zajścia określonego zdarzenia (wzbudzenie przejścia). Realizacja przejścia modeluje wystąpienie określonego zdarzenia. Realizacji przejścia towarzyszy usunięcie po jednym znaku miejsc wejściowych przejścia i dodanie po jednym znaku do miejsc wyjściowych przejścia. W ten sposób wskazuje się nowe warunki – skutki wystąpienia rozpatrywanego zdarzenia.

Formalnie sieć Petriego definiujemy jako uporządkowaną trójkę $PN = (P, T, F)$ gdzie:

- $P \cap T = \varnothing$,
- $P \cup T \neq \varnothing$,
- $F \subseteq (P \times T) \cup (T \times P)$,
- $\mathrm{dom}(F) \cup \mathrm{cod}(F) = P \cup T$.

F jest relacją przepływu w sieci PN, $\mathrm{dom}(F)$ jest dziedziną relacji F, czyli $\mathrm{dom}(F) = \{x: \underset{y}{\forall} (x, y) \in F\}$, $\mathrm{cod}(F)$ jeset przeciwdziedziną relacji F, czyli $\mathrm{dom}(F) = \{y: \underset{x}{\forall} (x, y) \in F\}$. Elementy zbioru F nazywane są łukami, elementy zbioru P nazywane są miejscami (P-elementami), elementy zbioru T nazywane są tranzycjami (T-elementami).

W modelowaniu sterowników logicznych stosowane są sieci 1-ograniczne, czyli takie, w których pojemność miejsc wynosi 1. Definicja sieci n-ograniczonej jest następująca:

- miejsce $p \in P$ jest n-ograniczone ($n \in \mathrm{N}$) $\Leftrightarrow \underset{M \in [M_0\rangle}{\forall} M(p) \leq n$,
- MPN jest n-ograniczona ($n \in \mathrm{N}$) $\Leftrightarrow \underset{p \in P}{\forall} p$ jest n-ograniczone.

3. TABLICE W RLACYJNEJ BAZIE DANYCH

Sieć Petriego opisująca układ sterowania logicznego jest siecią 1-ograniczoną. Wprowadzenie nowej sieci do bazy danych sprowadza się do wypełnienia odpowiednich tabel. Podstawowe elementy sieci to miejsca i tranzycje. W relacyjnej bazie danych do opisu miejsc i tranzycji wykorzystamy dwie tablice, o nazwach `STATE` i `TRANSITION`:

`STATE` - Lista stanów (miejsc, kroków) układu. Dla każdego stanu podajemy podstawowe informacje:

Nazwa pola	Opis
State_Dsc	Opis tekstowy danego stanu układu, np. S15
State_Id	Unikalny identyfikator numeryczny, np. 15

TRANSITION - Lista tranzycji układu. Dla każdej tranzycji podajemy podstawowe informacje:

Nazwa pola	Opis
Trn_Dsc	Opis tekstowy danej tranzycji, np. T8
Trn_Id	Unikalny identyfikator numeryczny, np. 8

W miarę potrzeb tablice te można rozbudować o dodatkowe kolumny informacyjne opisujące inne atrybuty miejsc i tranzycji.

Relacje zachodzące pomiędzy miejscami i tranzycjami opisane zostaną w tablicy RELATION o następującym formacie:

RELATION - Tablica opisująca relacje pomiędzy miejscami i tranzycjami. Opis jest zbliżony do grafu znakowań – opisujemy, co wchodzi i co wychodzi z poszczególnych tranzycji:

Nazwa pola	Opis
Rel_Id	Unikalny identyfikator numeryczny relacji.
Trn_Id	Identyfikator tranzycji
State_Id_In	Identyfikator miejsca wchodzącego do tranzycji
State_Id_Out	Identyfikator miejsca wychodzącego z tranzycji

Dla jednej tranzycji może wystąpić kilka rekordów. Przyjęto, że 0 w polach State_Id_In i State_Id_Out oznacza że pole nie powinno być uwzględniane.

Zbiór tranzycji z prawdziwymi warunkami przejścia opisano za pomocą kolejnej tablicy TRANSACTIONACTIVE.

TransitionActive - tablica zawiera informacje, które tranzycje są aktywne, czyli które z nich mają prawdziwe warunki i pozwalają na aktywację miejsca. Wartości w tej tablicy powinny być ustawiane na podstawie warunków przy tranzycjach, jednak ten element nie jest jeszcze opracowany. Wprowadzamy, więc do tej tablicy wartość true/false, w zależności od tego czy dany warunek ma być prawdziwy czy fałszywy:

Nazwa pola	Opis
Trn_Id	Identyfikator tranzycji
Trn_Active	True/false w zależności od tego, czy warunek na tranzycji jest prawdziwy czy fałszywy

Aktywne stany układu (czyli stany ze znacznikiem) opisane są w tablicy STATEACTIVE:

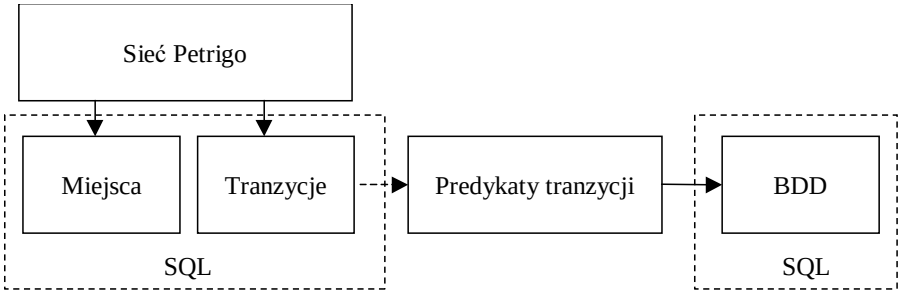
StateActive - Tablica zawiera informacje, które miejsca są aktualnie aktywne. W tablicy tej mamy więc wyjście układu. Miejsca aktywne zmieniają się w miarę wykonywania

kolejnych kroków symulacji w zależności od warunków na tranzycjach, z zachowaniem współbieżności i synchronizacji:

Nazwa pola	Opis
State_Id	Identyfikator miejsca
State_Active	True/false w zależności od tego, czy miejsce jest aktywne czy nieaktywne

4. MODELOWANI WARUNKÓW TRANZYCJI W SQL

W bazie danych nie można w prosty sposób zamodelować złożonych funkcji logicznych wykorzystywanych do opisu warunków tranzycji w sieciach Petriego. Problem ten można rozwiązać zapisując funkcję logiczną w postaci binarnego diagramu decyzyjnego, który następnie zostanie zamodelowany w relacyjnej bazie danych (rys. 1).



Rys. 1. Schemat zapisu sieci Petriego w relacyjnej bazie danych

Binarne diagramy decyzyjne (Binary Decision Diagram) dostarczają efektywnej metody do reprezentowania i przetwarzania funkcji logicznych. Binarny diagram decyzyjny prezentuje zbiór binarnych decyzji. Wynikiem ostatecznej decyzji jest prawda, albo fałsz. BDD przedstawiane są w postaci acyklicznych, skierowanych grafów z korzeniem, gdzie decyzje są związane z wierzchołkami grafu. Binarny diagram decyzyjny składa się z dwóch terminalowych wierzchołków „0” i „1” reprezentujących wartości funkcji logicznej, nieterminalowych węzłów reprezentujących zmienne funkcji logicznej oraz z zbioru łuków. Umownie przyjęto, że łuki łączące węzeł z jego lewym następnikiem odpowiadają wartości zerowej zmiennej decyzyjnej węzła, zaś łuki łączące węzeł z prawym jego następnikiem– wartości jeden zmiennej decyzyjnej [10, 11].

Zaimplementowanie w relacyjnej bazie danych binarnych diagramów decyzyjnych pozwoli na weryfikację wybranych właściwości sieci Petriego.

5. PODSUMOWANIE

Możliwe jest zamodelowaniu sterownika opisanego siecią Petriego za pomocą relacyjnej bazy danych. Istnieje możliwość symulacji sterownika przy pomocy języka SQL. W dalszych pracach planowane jest przeprowadzenie weryfikacji układu oraz badania jego własności jakościowych i ilościowych przy wykorzystaniu SQL i innych mechanizmów relacyjnej bazy danych.

LITERATURA

[1] M. Adamski, M. Chodań: *Modelowanie układów sterowania dyskretnego z wykorzystaniem sici SFC*, Wydawnictwo PZ, 2000

[2] P. H. Starke: *Sieci Petri*, WNT, 1987

[3] W. Reisig: *Sieci Petri*, WNT, 1988

[4] International Electrotechnical Commision: International Standards IEC 1131-3, 1993

[5] R. David, H. Alla: *Petri Nets & Grafcet. Tools for modelling discrete events systems*, Prentice Hall, 1992

[6] M. Kołopieńczyk: Modelling Process of Programmable Logic Controllers, Proc. Modelling and Symulation of Systems, 2002

[7] R. Muller: *Baza danych – język UML w modelowaniu danych*, Mikom, 2000

[8] American National Standards Institute: *Database Language-SQL ANSI X3.135*, 1992

[9] T. J. Teorey: *Database Modelling and Design*, Morgan Kaufmann, 1999

[10] P. Miczulski: *Algorytm konstrowania hierarchicznego grafu znakowań z wykorzystaniem przekształceń na funkcjach logicznych*, RUC, Materiały V Krajowej Konferencji Naukowej, Szczecin, 2000

[11] E. Pastor, O. Roig, J. Cortadella, M. R. Badia: *Petri Nets Analysis Using Boolean Manipulation*, Proc. of 15th International Conference: Application and Theory of Petri Nets, vol. 815 of Lecture Notes in Computer Science, Springer-Verlag, 1994