STUDIA INFORMATICA 2012
Volume 33 Number 2A (105)

Marek IWANIAK, Włodzimierz KHADZHYNOV
Politechnika Koszalińska, Wydział Elektroniki i Informatyki

# WYKORZYSTANIE SIECI PETRIEGO DO MODELOWANIA TRANSAKCJI ROZPROSZONYCH

**Streszczenie**. W artykule przedstawiono próbę wykorzystania zwyczajnej sieci Petriego do utworzenia i zbadania modelu protokołu 3PC (ang. *three-phase commit*), czyli protokołu trójfazowego zatwierdzania. Wprowadzone zostały podstawowe pojęcia dotyczące sieci Petriego. Wyjaśniono istotę transakcji w tradycyjnych i rozproszonych bazach danych oraz opisano działanie protokołu trójfazowego zatwierdzania. Przedstawiono sieć Petriego odwzorowującą działanie protokołu 3PC wraz z grafem osiągalnych rozwiązań oraz badaniem właściwości utworzonej sieci.

**Słowa kluczowe**: sieć Petriego, transakcje rozproszone, protokół trójfazowego zatwierdzania, 3PC

# USAGE OF PETRI NETS FOR DISTRIBUTED TRANSACTIONS MODELING

**Summary**. In this work the attempt of using ordinary Petri Net to model and study Three-Phase Commit protocol (3PC) is presented. Brief overview of Petri Nets is introduced. The nature of typical and distributed transactions are explained. 3PC protocol actions are described. The Petri Net of 3PC protocol followed by reachability analysis and study of net properties is presented.

**Keywords**: Petri Net, distributed transactions, 3PC, Three-Phase Commit protocol

## 1. Wstęp

Teoria i praktyka rozproszonych baz danych są bardzo złożonymi zagadnieniami. Jednak przez wiele lat badań nie udało się wykształcić standardu, który wspomógłby projektowanie, implementacje i wdrażanie rozwiązań opartych na rozproszonych bazach danych. W szczególności brak obowiązującego standardu dla środowiska heterogenicznego.

Teoria sieci Petriego znajduje zastosowanie w modelowaniu i analizie procesów współbieżnych. Strukturę i działanie sieci Petriego można przedstawić w postaci algebraicznej i przetwarzać ją z wykorzystaniem metod numerycznych.

Poszukiwanym rozwiązaniem jest możliwość: projektowania, modelowania i weryfikowania procesów zachodzących w rozproszonej bazie danych za pomocą sieci Petriego. Tak opracowana i sprawdzona sieć mogłaby posłużyć do nadzorowania przepływu danych w węzłach rozproszonej bazy danych lub do generowania konfiguracji zapewniających realizację zaprojektowanych procesów.

W artykule tym przedstawiono próbę wykorzystania sieci Petriego do odwzorowania działania protokołu 3PC i odpowiedź na pytanie: czy zwykła sieć Petriego jest wystarczającym narzędziem do odwzorowania złożonego procesu, jakim jest protokół zatwierdzania transakcji rozproszonej?

## 2. Sieci Petriego

Teoria matematyczna sieci Petriego została stworzona przez Carla Adama Petriego. Znajduje ona szerokie zastosowanie w analizie oraz modelowaniu. Przykładowymi obszarami, jakie mogą podlegać analizie, są procesy: współbieżne, przepływu roboczego [5] lub podejmowania decyzji [6].

Podstawową i najczęściej omawianą siecią Petriego jest sieć zwyczajna (ang. *ordinary*), określana również jako sieć klasy pozycja/tranzycja. Jej graficzną reprezentacją jest graf dwudzielny, składający się z dwóch typów węzłów łączonych łukami. Węzły te to:

- miejsca (pozycje, ang. *places*) – reprezentowane przez okręgi,
- tranzycje (przejścia, ang. *transitions*) – reprezentowane przez prostokąty lub kreski.

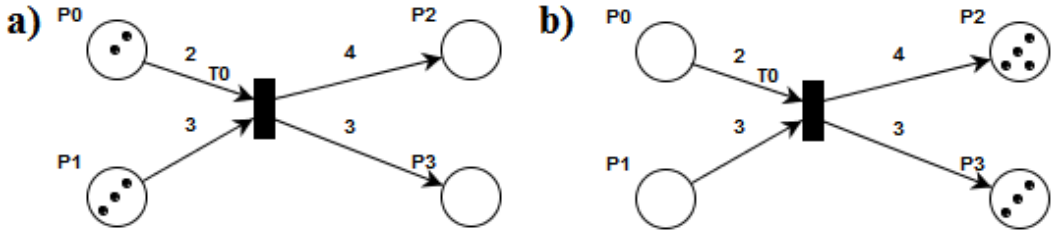
  Sieć Petri'ego można przedstawić jako trójkę N = (P , T , D), gdzie:
  - P jest zbiorem miejsc |P| = m,
  - T jest zbiorem tranzycji |T| = n,
  - D jest macierzą incydencji (ang. *incidence matrix*) o rozmiarze m x n, macierz ta opisuje relacje zbiorów miejsc oraz tranzycji,
  - $D^-$ jest macierzą preincydencji (ang. *pre-incidence matrix*) o rozmiarze m x n, zawiera elementy $d^-_{ij} = w(i,j)$ określające wagę łuku wejściowego tranzycji *j*, czyli łuku bezpośrednio łączącego miejsce *i* z tranzycją *j* (takim, że P x T -> N),
  - $D^+$ jest macierzą postincydencji (ang. *post-incidence matrix*) o rozmiarze m x n, zawiera elementy $d^-_{ij} = w(i,j)$ określające wagę łuku wyjściowego tranzycji *j*, czyli łuku bezpośrednio łączącego tranzycje *j* z miejscem *i* (takim, że T x P -> N ),

– D = $[D^- - D^+]$ jest macierzą incydencji (ang. *incidence matrix*) o rozmiarze m x n, macierz ta tworzona jest przez odjęcie macierzy postincydencji oraz preincydencji, zawiera elementy $d_{ij} = d^-_{ij} - d^+_{ij}$. Elementy te powstają z odjęcia wag łuków wyjściowych danej tranzycji od wag łuków wejściowych.

### 2.1. Znakowanie sieci i odpalanie tranzycji

Znakowaną sieć Periego można przedstawić jako czwórkę PN = (P, T, D, $M_0$), gdzie $M_0$ : P->{0,1,2,..} jest znakowaniem początkowym sieci, oznaczającym rozkład żetonów (znaczników, markerów, tokenów) w miejscach sieci.

Dla danego znakowania w sieci mogą zachodzić zdarzenia dynamiczne. Jeżeli w miejscach wejściowych danej tranzycji zgromadzi się liczba żetonów odpowiadająca wadze łuków wejściowych tej tranzycji, to tranzycja ta może zostać odpalona (ang. *fired*). Po odpaleniu danej tranzycji ze wszystkich jej miejsc wejściowych zgodnie z wagami łuków wejściowych żetony zostają usunięte. Do miejsc wyjściowych tranzycji żetony zostają dodane zgodnie z wagami łuków wyjściowych. Proces ten przedstawia rys. 1




Rys. 1. a) tranzycja gotowa do odpalenia, b) tranzycja po odpaleniu
Fig. 1. a) transition ready to be fired, b) transtion after firing

W zależności od znakowania początkowego zbiór osiągalnych rozwiązań danej sieci może być dla każdego znakowania inny. Zbiory możliwych znakowań tworzy się i prezentuje za pomocą grafu skierowanego nazywanego również drzewem osiągalności.

Drzewo osiągalności reprezentuje w skończony sposób kolejne odpalenia tranzycji oraz zmiany znakowania zwane również wykonaniem sieci. Korzeniem drzewa jest znakowanie początkowe. Węzłami drzewa są znakowania osiągalne, a łuki reprezentują odpalenie danej tranzycji. Liśćmi drzewa są znakowania końcowe lub powtórzone.

Graf osiągalnych rozwiązań można przygotować, analizując odpalenia kolejnych tranzycji. Analiza ta może zostać uproszczona dzięki algebraicznym własnościom sieci. Kolumny macierzy D określają zmianę znakowania sieci po odpaleniu tranzycji *j*. Kolejne znakowania możemy obliczyć wg następującego równania:

$$M_k = M_{k-1} + e[t_j]D,$$

gdzie: k = 1, 2, 3,…, a $e[t_j]$ jest wektorem odpalenia zawierającym 1 w pozycji odpowiadającej *j*-tej tranzycji.

### 2.2. Właściwości sieci

Właściwości sieci to:

- Ograniczoność (ang. *boundness*) – sieć jest ograniczona, gdy dla każdego z miejsc możemy określić skończoną liczbę żetonów występujących w danym miejscu. Wówczas zbiór możliwych znakowań jest skończony i można go przeszukiwać przeglądem zupełnym.
- Bezpieczeństwo (ang. *safeness*) – jest to uszczegółowiony przypadek ograniczoności sieci. Miejsce jest bezpieczne, gdy dla dowolnego znakowania liczba znaczników w danym miejscu wynosi 0 lub 1. Jeżeli wszystkie miejsca spełniają ten warunek, to można stwierdzić, że sieć jest bezpieczna. Właściwość ta jest istotna w przypadku sieci projektowanych do implementacji sprzętowej, gdzie miejscu odpowiada tranzystor przyjmujący w danym momencie 0 lub 1.
- Konfliktowość (ang. *conflictuality* ) – do konfliktu odpaleń tranzycji dochodzi wówczas, gdy dla danego znakowania początkowego przygotowana jest więcej niż jedna tranzycja lub, gdy odpalenie jednej lub więcej tranzycji uniemożliwia odpalenie pozostałych tranzycji [2].
- Zachowawczość (ang. *conservativness*) – sieć jest zachowawcza, jeżeli suma żetonów w każdym znakowaniu jest stała. Właściwość tę można udowodnić w przypadku prostych sieci.

### 2.3. Sieci czasowe

W czasowych sieciach Petriego dla tranzycji można zdefiniować pewne opóźnienia czasowe. Jeżeli dana tranzycja zostanie wzbudzona (w miejscach wejściowych pojawi się odpowiednia liczba żetonów), odpalenie tranzycji nastąpi dopiero po minięciu zadanego opóźnienia. Jeżeli dwie tranzycie czasowe współdzielą pewne miejsce wejściowe, wówczas zaistnieć może szczególna sytuacja. W czasie gdy wzbudzona tranzycja oczekuje na odpalenie, a druga wzbudzona tranzycja usunie żetony ze wspólnego miejsca wejściowego, odpalenie oczekującej tranzycji nie nastąpi.

### 2.4. Sieci kolorowe

Kolorowana sieć Petriego (Coloured Petri Net) jest grafem złożonym z miejsc, tranzycji i łuków, w którym poruszają się znaczniki przenoszące wartości określonego typu. Znacznikom określonego typu przypisywane są kolory. W każdym miejscu sieci może znajdować się wiele różnych znaczników, których typ musi być jednak zgodny z typem tego miejsca. Wzbudzenie tranzycji w bieżącym znakowaniu sieci zależy zarówno od obecności znaczników w miejscach wejściowych tranzycji, jak i od wartości przypisanych do tych znaczników.

## 3. Przetwarzanie transakcji w bazach danych

Zazwyczaj bazą danych nazywamy przechowywaną w formie cyfrowej kolekcję odpowiednio zorganizowanych danych. Przechowywane dane zwykle odwzorowują rzeczywiste obiekty. Za dbałość o poprawność, bezpieczeństwo i efektywny dostęp do przechowywanych danych najczęściej odpowiada DBMS (ang. *database managment system*), czyli system zarządzania bazą danych. Struktura bazy danych jest zbyt skomplikowana, by można było z nią pracować bez wykorzystania DBMS. Najczęściej DBMS odpowiada za udostępnienie usługi dostępu do bazy danych w sieci komputerowej. Za pośrednictwem lokalnego systemu operacyjnego DBMS dba o bezpieczne przechowywanie danych i pośredniczy we wszystkich operacjach, które użytkownicy wykonują na danych.

Przechowywane dane są w pełni przydatne tylko wtedy, gdy są pewne i spójne. Każda zmiana wykonywana na danych w bazie powoduje przejście do zupełnie nowego stanu bazy danych. Kolejne zmiany mogą występować równocześnie lub generować błędy. O zapewnienie poprawności i spójnego stanu bazy danych dba mechanizm transakcji.

Transakcja jest operacją, która po poprawnym wykonaniu gwarantuje zachowanie spójnego stanu bazy danych. Transakcje mogą składać się z dowolnej liczby operacji zapisu i odczytu oraz operować na wielu tabelach. Zmiany wprowadzane przez transakcje muszą zostać wykonane w całości lub w całości zostać wycofane.

Właściwościami gwarantującymi poprawne przetwarzanie transakcji są reguły ACID:

- Atomowość (ang. *atomicity*) – oznacza, że każda transakcja zostanie wykonana w całości lub wcale.
- Spójność (ang. *consistency*) – oznacza, że po zakończeniu transakcji system będzie spójny. Jeżeli w trakcie zatwierdzania transakcji okaże się, że naruszone zostały więzy integralności lub inne ograniczenia gwarantujące spójność i poprawność danych, to zmiany te nie zostaną zapisane.
- Izolacja (ang. *isotation*) – poziom izolacji określa możliwość wzajemnego odczytywania danych zmienianych przez współbieżnie wykonywane transakcje. Na podstawie wybranego poziomu izolacji można określić, jakich anomalii możemy spodziewać się w trakcie przetwarzania transakcji.
- Trwałość (ang. *durability*) – oznacza, że system potrafi uruchomić się po awarii i udostępnić spójne, nienaruszone i aktualne dane zapisane w ramach zatwierdzonych transakcji.

DBMS rejestruje wszystkie etapy przetwarzanych transakcji w dzienniku transakcji. W trakcie odtwarzania bazy danych po awarii odtwarzane są tylko zmiany wprowadzone

przez zatwierdzone transakcje, ponieważ tylko one gwarantują uzyskanie spójnego stanu bazy danych.

W trakcie przetwarzania transakcji w pojedynczej bazie danych może wystąpić wiele anomalii, takich jak: wzajemne blokowanie, konflikty, zakleszczenia. Pojedynczy DBMS, w zależności od swojej konfiguracji i możliwości, stara się niwelować i usuwać występujące anomalie.

## 4. Przetwarzanie transakcji w rozproszonych bazach danych

Rozproszona baza danych jest zbiorem wielu logicznie powiązanych ze sobą baz danych rozproszonych w sieci komputerowej. Różnorodność natury rozproszonej bazy danych jest opisana w [1]. Zaletami rozproszenia danych mogą być np.: zwiększenie wydajności (dane przetwarzane są w wielu maszynach), dzięki powieleniu danych, oraz odporność systemu na awarie jednej z lokalizacji. Przez zastosowanie przetwarzania rozproszonego rośnie złożoność wszystkich aspektów tradycyjnie związanych z bazą danych. Do aspektów tych należą: projektowanie systemu, programowanie aplikacji, fragmentacja danych w poszczególnych lokalizacjach, sterowanie współbieżnością i zarządzanie transakcjami, zapobieganie zakleszczeniom i sterowanie blokadami.

Jednym z aspektów jest podział danych przechowywanych w poszczególnych lokalizacjach. Jeżeli wszystkie lokalizacje (węzły) przechowują daną relację, ale zakres danych przechowywanych w każdej lokalizacji jest inny, wówczas mówimy o poziomej dekompozycji relacji. Jeżeli pewna relacja jest przechowywana w jednej centralnej lokalizacji i łączona jest z lokalnym fragmentem innej relacji, to wówczas mówimy o pionowej dekompozycji relacji.

Kolejnym aspektem jest zarządzanie transakcjami. Transakcje zachodzące w rozproszonej bazie danych możemy podzielić ze względu na czas trwania. Transakcje długotrwałe nazywane są sagami. Transakcje długotrwałe zazwyczaj związane są z procesem biznesowym, wymagającym reakcji lub decyzji użytkownika. Realizacja ich w tradycyjnym systemie transakcyjnym powodowałaby długotrwałe blokowanie zasobów. Zapewnienie takim transakcjom mechanizmu zatwierdzania/wycofywania realizowane jest na poziomie implementacji tego procesu w aplikacji. W logice aplikacji zaszywa się transakcje kompensacyjne, pozwalające na etapowe wycofywanie zmian wprowadzonych w trakcie realizacji procesu biznesowego. Transakcje takie są specyficzne dla danej aplikacji, zarządzanie nimi leży poza kompetencjami DBMS. Z przykładem modelu długotrwałej transakcji rozproszonej można zapoznać się w [7].

Transakcje krótkotrwałe realizowane są na poziomie systemu DBMS. Z punktu widzenia aplikacji/użytkownika wykonanie rozproszonej transakcji nie powinno się różnić od wykona-

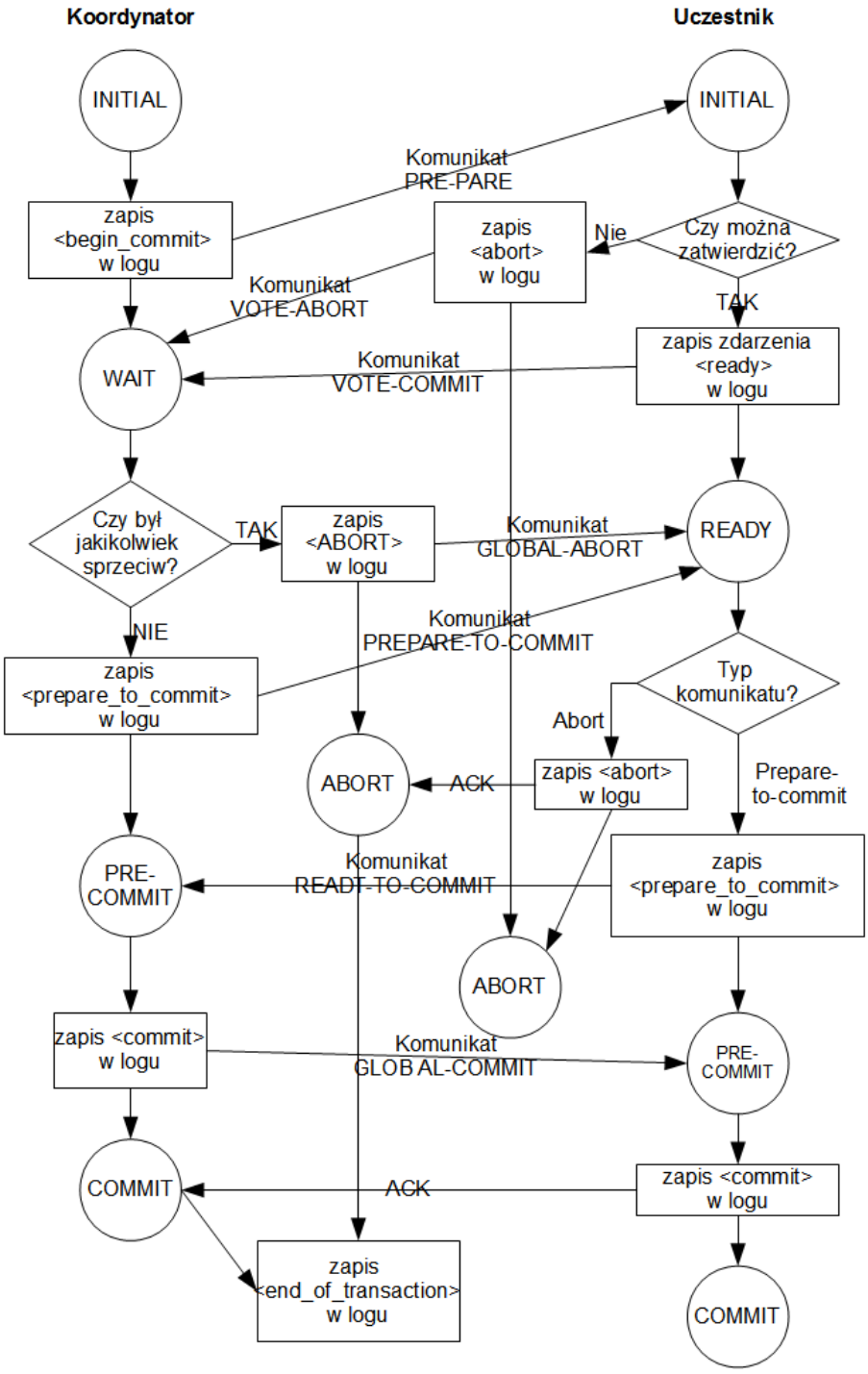
nia transakcji w systemie tradycyjnym. Temat współbieżnego wykonywania transakcji rozproszonych w zgodności z postulatami ACID obejmuje aspekty wymagające rozważenia.

Samodzielna baza danych ma zaimplementowanego TM (ang. *transaction manager*) zwanego menadżerem transakcji, dbającego o prawidłowe zarządzanie stanem bazy danych oraz komunikację z klientami. W rozproszonej bazie danych pojawia się dodatkowa specjalistyczna komunikacja pomiędzy innymi menadżerami transakcji. Komunikacja ta zachodzi pomiędzy menadżerem transakcji, pełniącym rolę koordynatora, a pozostałymi menadżrami transakcji, pełniącymi rolę uczestników transakcji. Sposób komunikacji między koordynatorem a uczestnikami definiuje określony protokół rozproszonego zatwierdzania. Klasycznym przykładem takiego protokołu jest zatwierdzanie dwufazowe (2 Phase Commit – 2PC). Protokół ten, zgodnie z nazwą, realizuje dwie fazy: głosowania i zatwierdzania. Faza głosowania następuje po przesłaniu przez koordynatora zapytania o możliwość zatwierdzenia danej transakcji rozproszonej. Komunikaty zwrotne od uczestników do koordynatora zawierają informacje o gotowości danego uczestnika do zatwierdzenia transakcji. Po zebraniu głosów od wszystkich uczestników następuje faza zatwierdzania lub wycofywania. Jeżeli koordynator otrzymał od wszystkich uczestników potwierdzenie gotowości, wysyła komunikat o globalnym zatwierdzeniu transakcji. Jeżeli którykolwiek uczestnik zagłosuje przeciw zatwierdzeniu transakcji lub w ogóle nie prześle komunikatu (np. z powodu awarii), wówczas koordynator prześle komunikat o globalnym odrzuceniu tej transakcji. Podejmowanie decyzji o zatwierdzeniu tylko przy jednomyślnym głosowaniu jest podstawowym sposobem zapewnienia atomowości transakcji rozproszonej.

### 4.1. Protokół trójfazowego zatwierdzania transakcji rozproszonych

Proces trójfazowego zatwierdzania transakcji rozproszonych wygląda następująco:

1. W pliku dziennika koordynator tworzy informacje o rozpoczęciu zatwierdzania („begin_commit”) i przesyła wiadomość „prepare” (przygotuj) do uczestników transakcji rozproszonej. Następnie przechodzi do stanu oczekiwania na odpowiedzi od uczestników (WAIT).
2. Uczestnicy podejmują decyzje, czy są gotowi do zatwierdzania, i zapisują w swoich plikach dziennika decyzje: gotowy (*ready*) lub odwołaj (*abort*). Następnie przesyłają do koordynatora swoje głosy za zatwierdzeniem (*vote commit*) lub przerwaniem (*vote abort*) transakcji rozproszonej. W zależności od podjętej decyzji dany uczestnik przechodzi do stanu odrzucenia transakcji rozproszonej (ABORT) lub stanu gotowości (READY) do zatwierdzenia transakcji rozproszonej.




Rys. 2. Algorytm przedstawiający realizację protokołu trójfazowego zatwierdzania z jednym uczestnikiem

Fig. 2. Three Phase Commit protocol algorithm with one participant

3. Koordynator sprawdza głosy otrzymane od wszystkich uczestników.
   a. Jeżeli wystąpił choć jeden głos za odrzuceniem transakcji lub choć jeden z uczestników nie nadesłał swojego głosu, koordynator podejmie decyzję o odrzuceniu transakcji. Zapisuje on w pliku dziennika podjętą decyzję o przerwaniu i przesyła do uczestników komunikat o wycofaniu transakcji (*global abort*). Koordynator przechodzi do

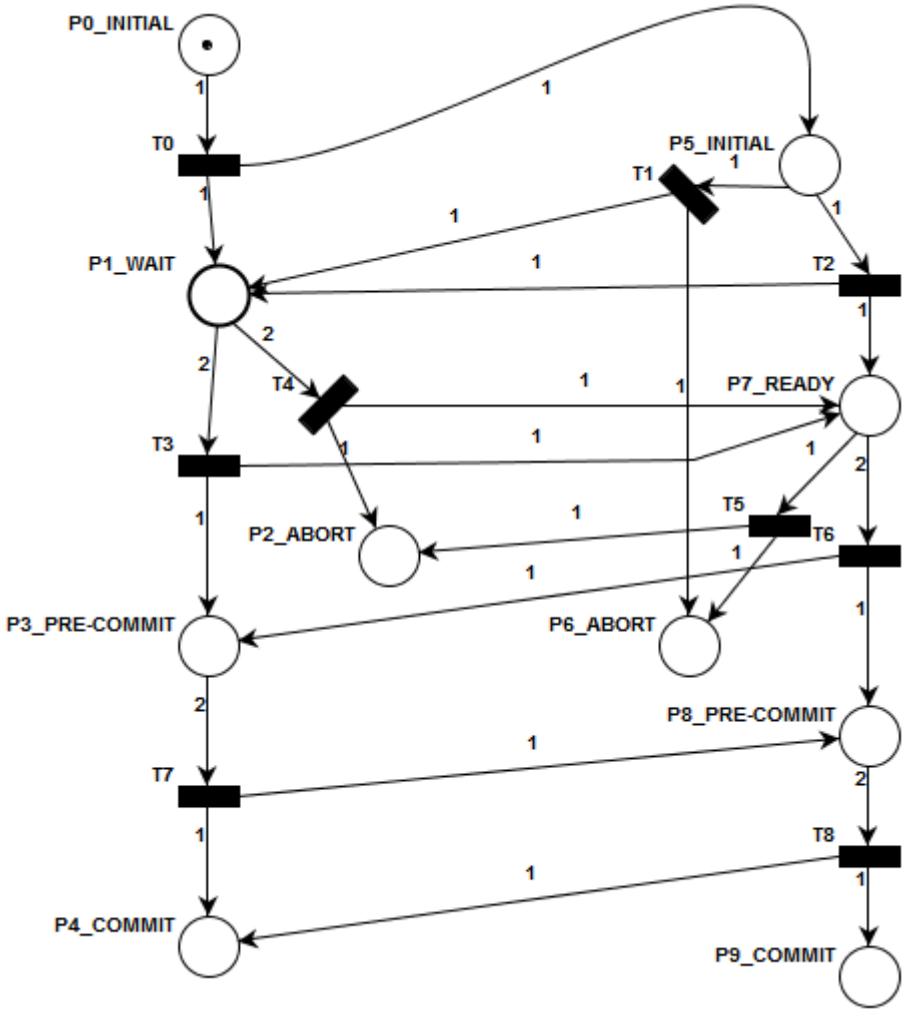
stanu przerwania transakcji rozproszonej (ABORT) oraz zapisuje zakończenie transakcji w dzienniku.

b. Jeżeli wszyscy uczestnicy odpowiedzieli i wszystkie głosy potwierdzają gotowość do zatwierdzenia transakcji, wówczas koordynator podejmuje decyzję o przejściu do kolejnej fazy zatwierdzania. Zapisuje on w dzienniku informację o przygotowaniu do zatwierdzania. Do uczestników przesyłany jest komunikat polecający przygotowanie do zatwierdzania (*prepare-to-commit*). Koordynator przechodzi do stanu przed zatwierdzeniem (*pre-commit*).

4. Uczestnicy zapisują otrzymany komunikat w logach. W przypadku przerwania uczestnicy przechodzą do stanu przerwania (ABORT) i przekazują do koordynatora potwierdzenie otrzymania komunikatu (ACK). W przypadku komunikatu przygotowania do zatwierdzenia uczestnicy przechodzą do stanu przed zatwierdzeniem (*pre-commit*) i przekazują do koordynatora potwierdzenie otrzymania komunikatu i gotowości do dalszego zatwierdzania (*ready-to-commit*).
5. Po otrzymaniu przez koordynatora komunikatu od uczestników o gotowości do zapisu (*ready-to-commit*) w dzienniku zostaje zapisany komunikat o zatwierdzeniu. Koordynator przesyła do wszystkich uczestników polecenie zatwierdzenia (global commit). Koordynator przechodzi do stanu zatwierdzenia transakcji rozproszonej (COMMIT).
6. Uczestnicy zapisują komunikat o zatwierdzeniu do dziennika oraz przesyłają do koordynatora komunikat potwierdzający otrzymanie komunikatu, po czym uczestnicy przechodzą do stanu zatwierdzenia (COMMIT)
7. Koordynator zapisuje zakończenie transakcji w dzienniku.

## 5. Model protokołu trójfazowego jako klasyczna sieć Petriego

Opisany w punkcie 4.1 protokół 3PC przedstawiony został jako sieć Petriego na rys. 3. Przykład został stworzony dla przypadku koordynatora i jednego uczestnika transakcji rozproszonej. Miejsca związane z koordynatorem znajdują się przy lewej krawędzi rysunku, a miejsca związane z uczestnikiem – przy prawej.




Rys. 3. Sieć Petriego odwzorowująca algorytm protokołu 3PC
Fig. 3. Petri Net of 3PC protocol algorithm

Tabela 1

Miejsca i tranzycje związane z koordynatorem

| Miejsca | | Tranzycje | |
|---|---|---|---|
| P0 | INITIAL | T0 | 1) wpis do logu (*begin_commit*)<br>2) wiadomość do uczestników (*prepare*) |
| P1 | WAIT | T4 | 1) wpis do logu (*abort*)<br>2) wiadomość do uczestników (*global-abort*) |
| P2 | ABORT | T3 | 1) wpis do logu (*prepare-to-commit*)<br>2) wiadomość do uczestników (*prepare-to-commit*) |
| P3 | PRECOMMIT | T7 | 1) wpis do logu (*commit*)<br>2) wiadomość do uczestników (*global-commit*) |
| P4 | COMMIT | | |

Tabela 2

Miejsca i tranzycje związane z uczestnikami

| Miejsca | | Tranzycje | |
|---|---|---|---|
| P5 | INITIAL | T1 | 1) wpis do logu (*abort*)<br>2) wiadomość do koordynatora (*vote-abort*) |
| P6 | ABORT | T2 | 1) wpis do logu (*ready*)<br>2) wiadomość do koordynatora (*vote-commit*) |
| P7 | READY | T5 | 1) wpis do logu (*abort*)<br>2) potwierdzenie do koordynatora |
| P8 | PRE-COMMIT | T6 | 1) wpis do logu (*prepare-to-commit*)<br>2) wiadomość do koordynatora (*ready-to-commit*) |
| P9 | COMMIT | T8 | 1) wpis do logu (*commit*)<br>2) potwierdzenie do koordynatora |

### 5.1. Macierze incydencji dla sieci Petriego

Wiersze tabeli 3 zawierają miejsca $P_0$, $P_1$, $P_2$, $P_3$, $P_4$, reprezentujące stany koordynatora, oraz $P_5$, $P_6$, $P_7$, $P_8$, $P_9$, reprezentujące stany uczestnika. Kolumny $t_0 – t_9$ reprezentują tranzycje (przejścia). Macierz preincydencji zawiera wagi łuków wejściowych danej tranzycji. W przypadku gdy brak jest łuku łączącego miejsce *i* z daną tranzycją *j*, wpisujemy liczbę 0. Macierz ta oznaczana jest jako $[D^-]$.

Tabela 3

Macierz preincydencji

| $D^-$ | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ | $t_8$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $P_0$ | INITIAL | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_1$ | WAIT | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 |
| $P_2$ | ABORT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_3$ | PRECOMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 |
| $P_4$ | COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_5$ | INITIAL | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_6$ | ABORT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_7$ | READY | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 0 |
| $P_8$ | PRE-COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| $P_9$ | COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Tabela 4

Macierz postincydencji

| $D^+$ | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ | $t_8$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $P_0$ | INITIAL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_1$ | WAIT | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_2$ | ABORT | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| $P_3$ | PRECOMMIT | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| $P_4$ | COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $P_5$ | INITIAL | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_6$ | ABORT | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $P_7$ | READY | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| $P_8$ | PRE-COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| $P_9$ | COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Tabela 5

Połączona macierz incydencji

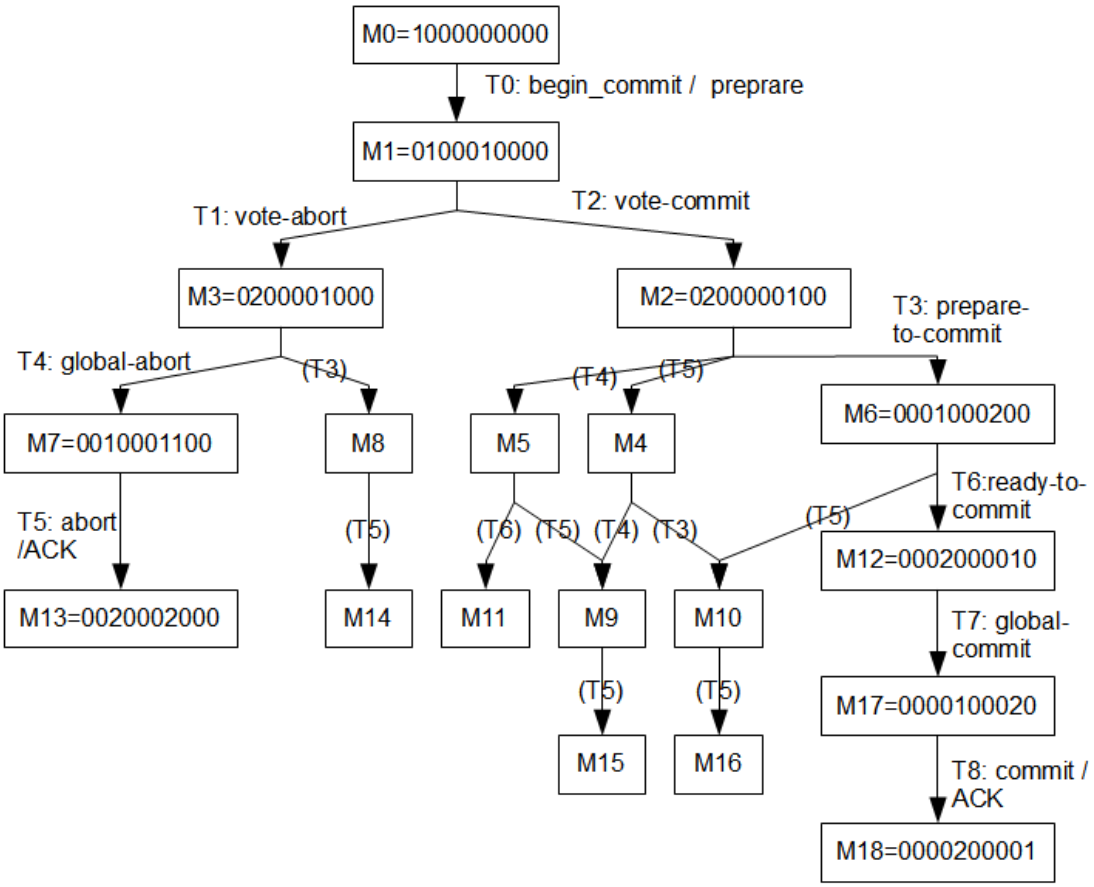
| D | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ | $t_8$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $P_0$ | INITIAL | -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_1$ | WAIT | 1 | 1 | 1 | -2 | -2 | 0 | 0 | 0 | 0 |
| $P_2$ | ABORT | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| $P_3$ | PRECOMMIT | 0 | 0 | 0 | 1 | 0 | 0 | 1 | -2 | 0 |
| $P_4$ | COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $P_5$ | INITIAL | 1 | -1 | -1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $P_6$ | ABORT | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $P_7$ | READY | 0 | 0 | 1 | 1 | 1 | -1 | -2 | 0 | 0 |
| $P_8$ | PRE-COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | -2 |
| $P_9$ | COMMIT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Wiersze tabeli 4 zawierają miejsca $P_0$, $P_1$, $P_2$, $P_3$, $P_4$, reprezentujące stany koordynatora, oraz $P_5$, $P_6$, $P_7$, $P_8$, $P_9$, reprezentujące stany uczestnika. Kolumny $t_0$ – $t_9$ reprezentują tranzycje. Macierz postincydencji zawiera wagi łuków wyjściowych danej tranzycji. W przypadku gdy brak łuku łączącego daną tranzycję *j* z miejsce *i*, wpisujemy liczbę 0. Macierz ta oznaczana jest jako $[D^+]$.

Połączona macierz incydencji wyliczana jest poprzez odjęcia macierzy postincydencji i preincydencji $[D^+ - D^-]$. Opisuje ona dynamikę zmian zachodzących w danej sieci Petriego. Zawiera informacje o tym, ile żetonów zostanie dodanych lub odjętych we wszystkich miejscach po odpaleniu danej tranzycji $t_j$.

## 6. Graf osiągalnych rozwiązań

Znakowanie początkowe to wektor o postaci $M_0 = [10000000000]$. Dla danego znakowania sprawdzamy, która z tranzycji zostanie wzbudzona. Żeton w miejscu P1 spowoduje odpalenie tranzycji $t_0$.




Rys. 4. Drzewo osiągalnych rozwiązań dla sieci Petriego z rys. 3 (niepożądane stany pokazane są bez znakowań)

Fig. 4. Reachability tree for Petri net from Fig. 3 (undesirable states are shown without markings)

Wzór algebraiczny na wyznaczanie kolejnych znakowań po odpaleniu *j*-tej tranzycji wygląda następująco:

$$M_k = M_{k-1} + e[t_j]D,$$

gdzie: $M_k$ – nowe znakowanie, $M_{k-1}$ – poprzednie znakowanie, dla k = 1 jest to znakowanie początkowe, $e[t_j]$ – wektor kolumnowy o rozmiarze *i* z jedną pozycją niezerową w indeksie odpowiadającym odpalanej tranzycje, D – macierz incydencji.

Dla danego znakowania $M_0$ chcemy wyznaczyć znakowanie $M_1$ przez odpalenie tranzycje $t_0$:

$$e[t_0]*D = [-1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ ],$$

$$M_0=[10000000000].$$

Po dodaniu obu wektorów otrzymamy kolejne znakowanie sieci:

$$M_1=[0100010000].$$

### 6.1. Właściwości sieci

Własności sieci to:

- Ograniczoność – w prezentowanej sieci w danym miejscu występują maksymalnie dwa żetony, więc sieć ta jest ograniczona.
- Bezpieczeństwo – maksymalna pojemność miejsc w prezentowanej sieci wynosi 2, dlatego sieć nie jest bezpieczna.
- Konfliktowość – w prezentowanym modelu do konfliktu dochodzi w miejscach, w których protokół musi podejmować decyzje. Pojawienie się w miejscach P1, P2 i P4 odpowiedniej liczby żetonów powoduje przygotowanie obydwu tranzycji wyjściowych (P1(T1,T2), P5(T3,T4), P7(T5,T6)). W miejscu P5 proces – uczestnika na podstawie wystąpienia błędów związanych z lokalnym fragmentem transakcji rozproszonej lub ich braku – musi zdecydować, czy odpalić tranzycję, która prześle głos za zatwierdzeniem, czy odpalić tranzycję, która prześle głos przeciw zatwierdzeniu (*vote-commit*, *vote-abort*).
- Zachowawczość – prezentowany model sieci jest złożony i nie można stwierdzić, że sieć jest zachowawcza.

## 7. Podsumowanie

W artykule przedstawiono model sieci Petriego opisujący działanie protokołu 3PC. Za pomocą macierzy incydencji wyznaczono drzewo osiągalnych rozwiązań, na którym zidentyfikowano stany niepożądane, lecz osiągalne.

W trakcie badania zaproponowanej sieci Petriego wykazano wiele ograniczeń i problemów. Wynikają one głównie z ograniczeń zwykłej sieci Petriego. Konieczne jest zastosowanie sieci wyższego poziomu, np. sieci kolorowej. Zastosowanie sieci kolorowej pozwoli na wyeliminowanie konfliktów w miejscach decyzyjnych, a zatem na ograniczenie liczby niepożądanych stanów w analizie osiągalnych rozwiązań.

## BIBLIOGRAFIA

1. Tamer Özsu M., Valduriez P.: Principles of Distributed Database Systems, III edition. Springer, 2011.
2. Banaszak Z., Majdzik P., Wójcik R.: Procesy współbieżne: modele efektywności funkcjonowania. Rozdział 2: Modele sieci Petriego. Politechnika Koszalińska, Koszalin 2011, s. 93÷143.
3. Sarkar B. B., Chaki N.: Transaction Management for Distributed Database using Petri Nets. International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM), Vol. 2, 2010, s. 69÷76.
4. Sarkar B., Chaki N.: Virtual Data Warehouse Modeling Using Petri Nets for Distributed Decision Making. Journal of Convergence Information Technology, Vol. 5, No. 5, 2010, s. 8÷21.
5. Pieczonka S.: System zarządzania produkcją oprogramowania – moduł definiowania procesów, [w:] Kozielski S., Małysiak B., Kasprowski P., Mrozek D. (red.): Bazy danych: rozwój metod i technologii. WKŁ, 2008.
6. Dec G., Jędrzejec B., Rząsa W.: Kolorowana sieć Petriego jako model systemu podejmowania decyzji kredytowej. Studia Informatica, Vol. 31 No. 2A(89), Wydawnictwo Politechniki Śląskiej, Gliwice 2010.
7. Mroczkiewicz P.: Model długotrwałych transakcji rozproszonych z uwzględnieniem obsługi zdarzeń, [w:] Kozielski S., Małysiak B., Kasprowski P., Mrozek D. (red.): Bazy danych: struktury, algorytmy, metody. WKŁ, 2006.

Wpłynęło do Redakcji 31 stycznia 2012 r.

**Abstract**

This paper presents the attempt of using ordinary Petri Net to model and study Three-Phase Commit protocol (3PC). The motivation for this work is presented in Introduction. Second chapter introduces basic information about Petri Nets, marked Petri Net, firing of transitions, Timed Petri Net and Colored Petri Net. Third chapter provides the basic concepts about database transactions, followed by overview of distributed transactions processing. Three-Phase Commit protocol actions are described and shown on Fig. 2. Fourth chapter with Fig. 3 introduces The Petri Net of 3PC protocol. The incidence matrixes are presented and described. The reachability graph, based upon incidence matrix, is created and shown on Fig. 4 and followed by a short study of Petri Net properties. The Summary describes detected problem and irregularities. Possible solution for further research is stated.

**Adresy**

Marek IWANIAK: Politechnika Koszalińska, Wydział Elektroniki i Informatyki,
ul. Śniadeckich 2, 75-453 Koszalin, Polska, marek.iwaniak@tu.koszalin.pl.
Włodzimierz KHADZHYNOV: Politechnika Koszalińska, Wydział Elektroniki i Informatyki,
ul. Śniadeckich 2, 75-453 Koszalin, Polska, hadginov@ie.tu.koszalin.pl.