AGH

AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Teoria Współbieżności
zadanie 8 - sieci Petri

Przemysław Popowski
Informatyka WI, III rok
Gr 7, wtorek 18:30

Spis treści

1 Cel ćwiczenia

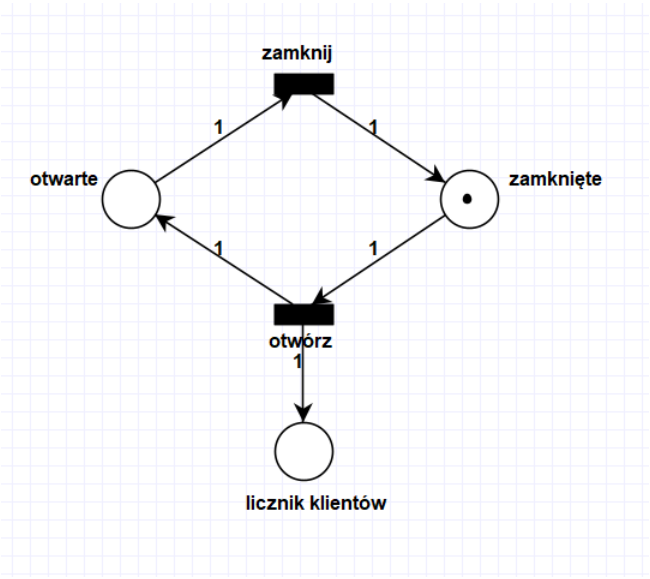
Celem ćwiczenia było zapoznanie się z podstawowymi własnościami sieci Petriego poprzez tworzenie i analizowanie różnych przykładów sieci (m.in. maszyny stanów, wzajemnego wykluczania, różnych wariantów producenta i konsumenta, wariantu z zakleszczeniem np. zastój meksykański).

2 Zadanie 1

2.1 Treść zadania

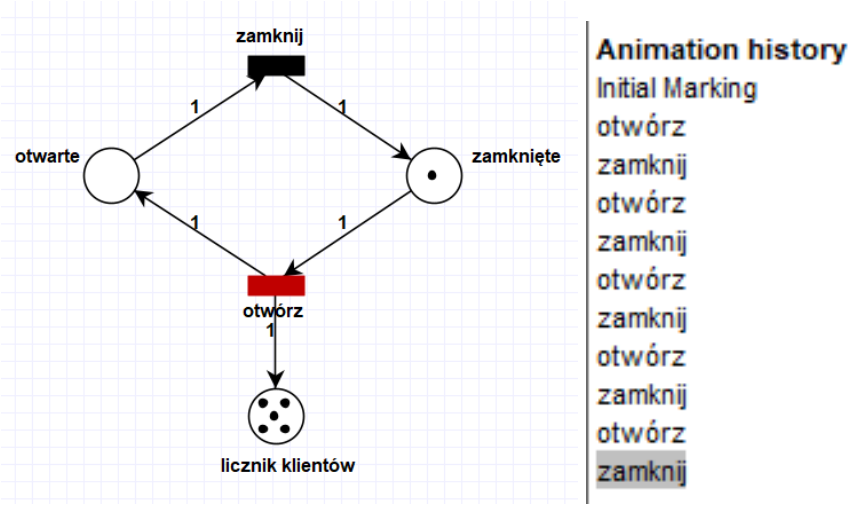
Wymyślić własną maszynę stanów, zasymulować przykład i dokonać analizy grafu osiągalności oraz niezmienników.

2.2 Rozwiązanie

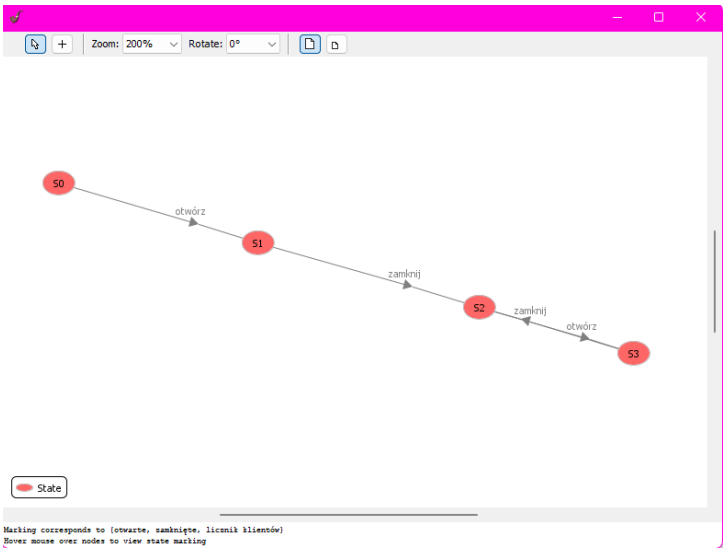


Stworzona przeze mnie sieć ma za zadanie symulować otwieranie i zamykanie drzwi w sklepie, wraz z licznikiem klientów. Każdy token w "licznik klientów"oznacza ile razy zostały otwarte, a następnie zamknięte drzwi.

2.3 Symulacja

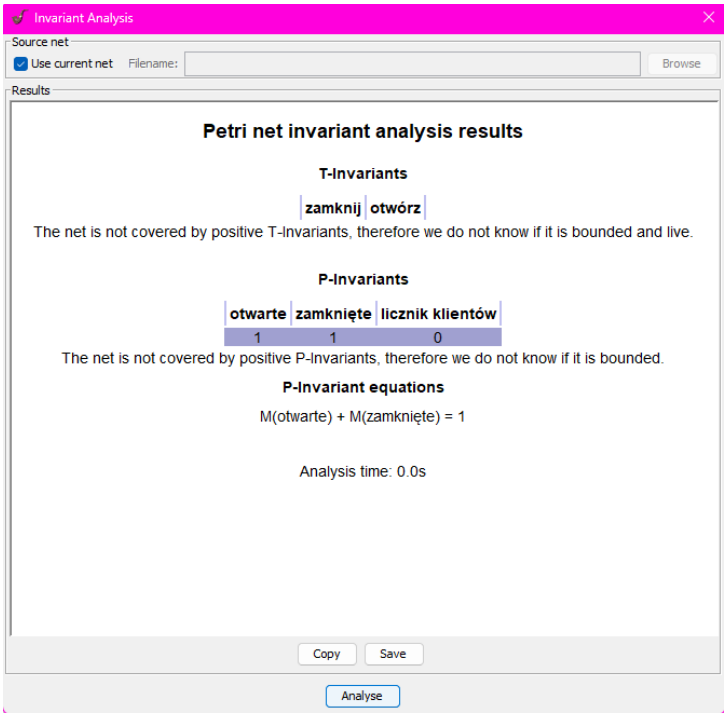


2.4 Analiza grafu osiągalności



Analizując graf osiągalności, zauważyć możemy, przejście ze stanu początkowego (drzwi zamknięte) w S1 (drzwi otwarte), a następnie między stanami S2 i S3 powstał cykl otwierania oraz zamykania drzwi (stany - otwarte/zamknięte). Zgadza się to z ideą początkową, ponieważ licznik klientów tylko na samym początku jest równy 0. Później cały czas się zwiększa, więc nie wrócimy już do stanu początkowego. Dlatego zostajemy uwięzieni między S2 i S3, a do S1 i S0 nie wrócimy.

2.5 Analiza niezmienników



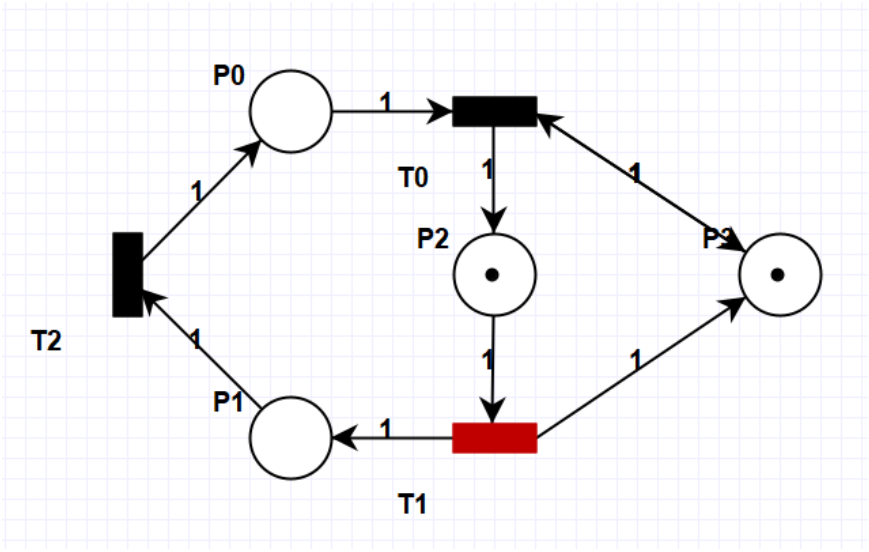
Analizując niezmienniki, możemy wywnioskować, że nasza sieć nie jest odwracalna. Dzieje się tak dlatego, że uruchamiając symulację nasz pojedynczy token z każdym kolejnym cyklem zwiększa swoją liczebność o 1, przez co nie wrócimy już do stanu, w którym zaczęliśmy. Z tego również zauważyć możemy, że nasza sieć nie jest ograniczona, ponieważ "licznik klientów"nigdy nie będzie posiadał skończonej ilości tokenów.

Natomiast analizując niezmienniki miejsc, w całości symulacji jest miejsce tylko na jeden token, zatem nie ma szans aby co jeden krok wykonywał się cały cykl, potrzebne są minimum dwa kroki.

3 Zadanie 2

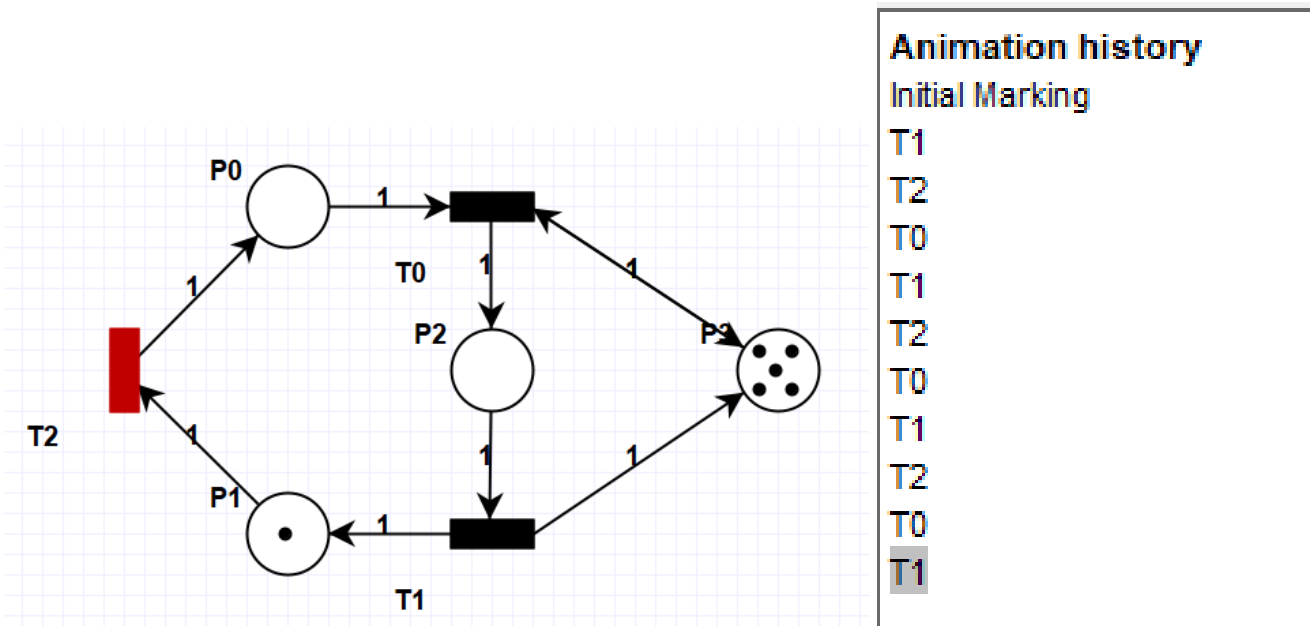
3.1 Treść zadania

Zasymulować sieć jak poniżej:



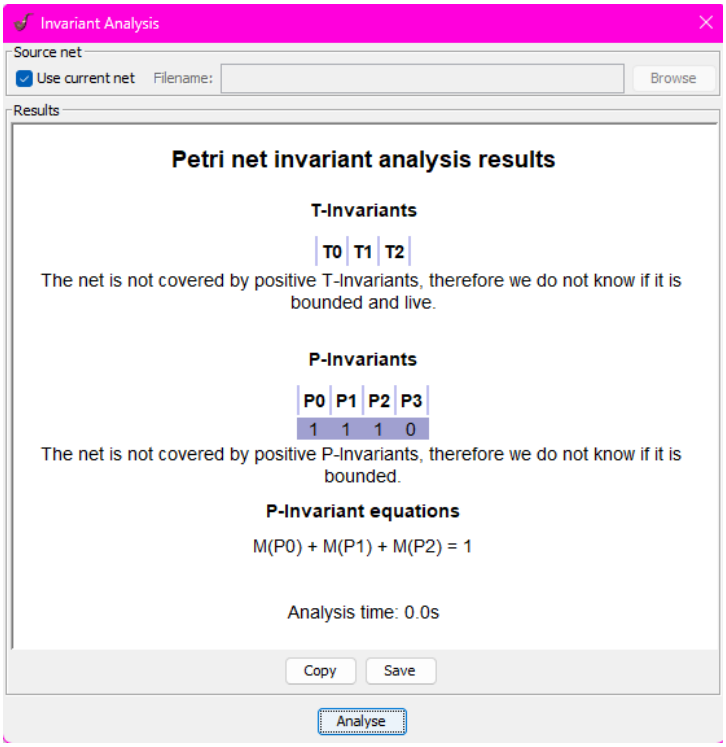
Dokonać analizy niezmienników przejść. Jaki wniosek można wyciągnąć o odwracalności sieci? Wygenerować graf osiągalności. Proszę wywnioskować z grafu, czy siec jest żywa. Proszę wywnioskować czy jest ograniczona. Objaśnić wniosek.

3.2 Symulacja



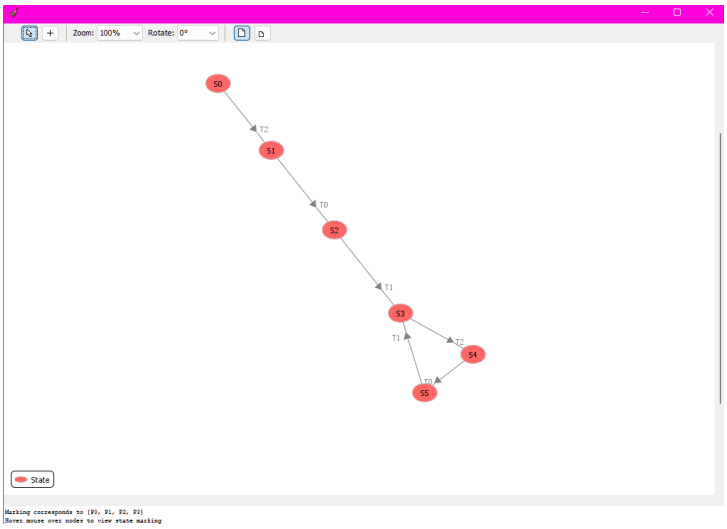
Przeprowadzona została przykładowa symulacja składająca się z kroku startowego i 10 następnych kroków, po której wykonaniu w stanie P3 znalazło się 5 tokenów.

3.3 Analiza niezmienników przejść



Analizując niezmienniki przejść, możemy zauważyć brak pozytywnych przejść. Widok ten podpowiada nam, że sieć nie jest odwracalna. Działa to trochę na podobnej zasadzie jak wymyślony przeze mnie przykład w zadaniu 1. W momencie uruchomienia symulacji, W P3 pojawia się coraz więcej tokenów, co powoduje, że nie jesteśmy w stanie wrócić do stanu początkowego. Również sieć nasza nie jest ograniczona, przez to że jesteśmy w stanie osiągnąć nieskończoną wartość w P3.

3.4 Graf osiągalności

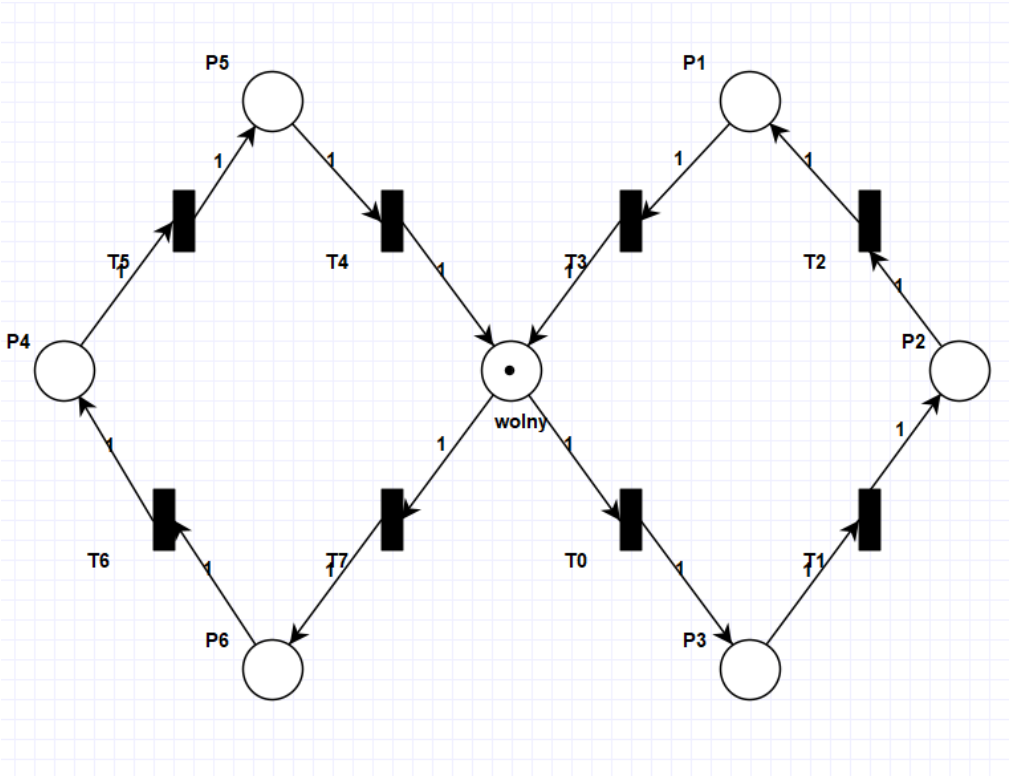


Analizując graf osiągalności, jesteśmy w stanie wywnioskować, że podana sieć jest żywa. Uzasadnione jest to tym, że możemy zawsze skorzystać z każdego podanego w sieci przejścia (T0, T1, T2). Każde z przejść jest zawarte w cyklu, więc będzie zawsze dostępne i nie ulegnie nigdy zablokowaniu bądź nie będzie nieosiągalne.

4 Zadanie 3

4.1 Treść zadania

Zasymulować wzajemne wykluczanie dwóch procesów na wspólnym zasobie. Dokonać analizy niezmienników miejsc oraz wyjaśnić znaczenie równań (P-invariant equations). Które równanie pokazuje działanie ochrony sekcji krytycznej?

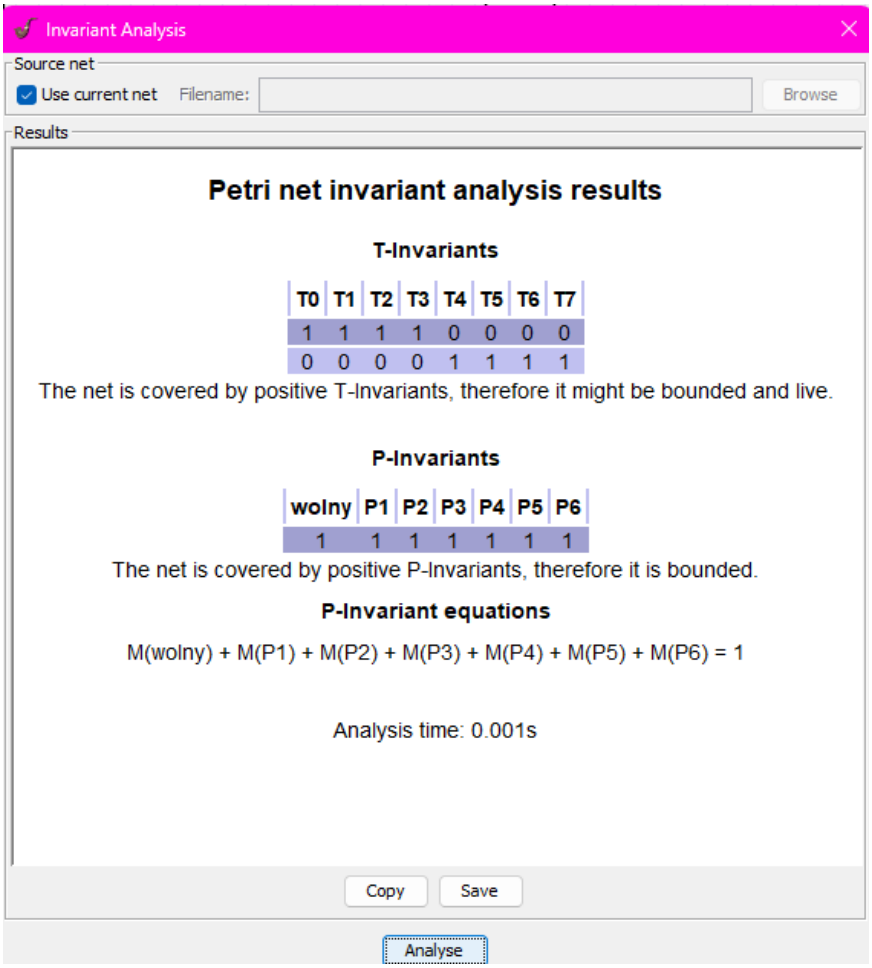


4.2 Symulacja

Animation history

Initial Marking
T0
T1
T2
T3
T7
T6
T5
T4
T7
T6
T5
T4
T0
T1
T2
T3
T7
T6
T5
T4
T7
T6

4.3 Analiza niezmienników miejsc



Z niezmienników miejsc możemy wyciągnąć wniosek, że nasza sieć jest ograniczona, ponieważ każde miejsce w sieci ma swoje równanie bazowe, zatem liczba tokenów w całości symulacji nie ulega powiększeniu ani zmniejszeniu (jedno równanie zawierające wszystkie miejsca i równe 1). Sieć jest bezpieczna, ponieważ w całości jest tylko jeden token, a więc w równaniu bazowym liczba tokenów jest również równa 1.

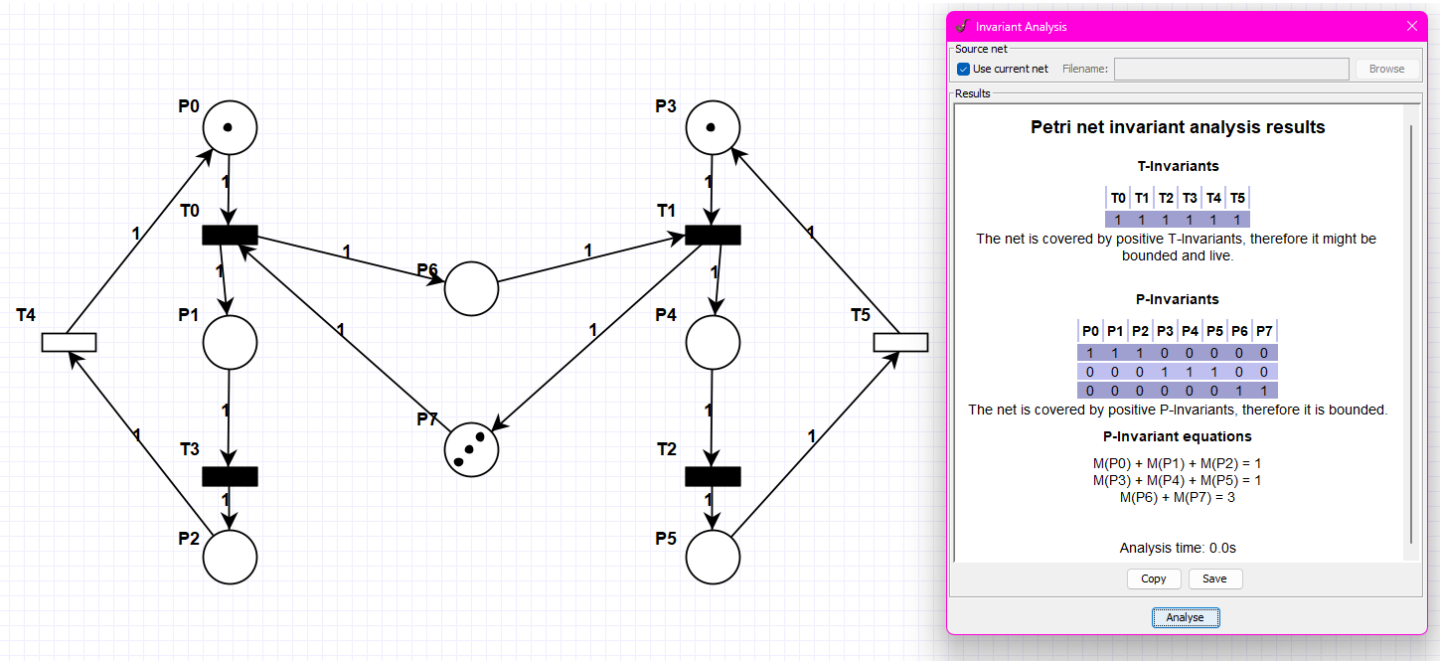
Równanie bazowe wskazuje nam na to, że w danym momencie tylko jeden proces jest w stanie posiadać token. Nie następuje taka sytuacja, że wiele procesów przetrzymuje token w jednej chwili. Czyli w skrócie pokazuje nam sytuację ochrony sekcji krytycznej, a zatem to, o co nam chodziło w zadaniu.

5 Zadanie 4

5.1 Treść zadania

Uruchomić problem producenta i konsumenta z ograniczonym buforem (można posłużyć się przykładem, menu: file, examples). Dokonać analizy niezmienników. Czy siec jest zachowawcza? Które równanie mówi nam o rozmiarze bufora?

5.2 Sieć oraz analiza niezmienników



,

Analizując niezmienniki możemy zauważyć, że sieć jest pokryta w całości pozytywnymi przejściami. W związku z tym, sieć jest żywa, ponieważ wszystkie przejścia mogą być wykonane. Podana sieć jest również odwracalna, gdyż można dojść ponownie do stanu początkowego, z którego zaczęliśmy symulację. Natomiast z niezmienników miejsc możemy wyciągnąć wniosek, że nasza sieć jest ograniczona, ponieważ każde miejsce w sieci ma swoje równanie bazowe, zatem liczba tokenów w całości symulacji nie ulega powiększeniu ani zmniejszeniu. Sieć nie jest bezpieczna, ponieważ w miejscu P6 i P7, a bardziej w ich równaniu bazowym (M(P6) + M(P7) = 3), liczba tokenów nie jest równa 1.

Czy sieć jest zachowawcza? Tak, sieć jest zachowawcza, ponieważ liczba tokenów w sieci nie ulega zmianie i pozostaje stała.

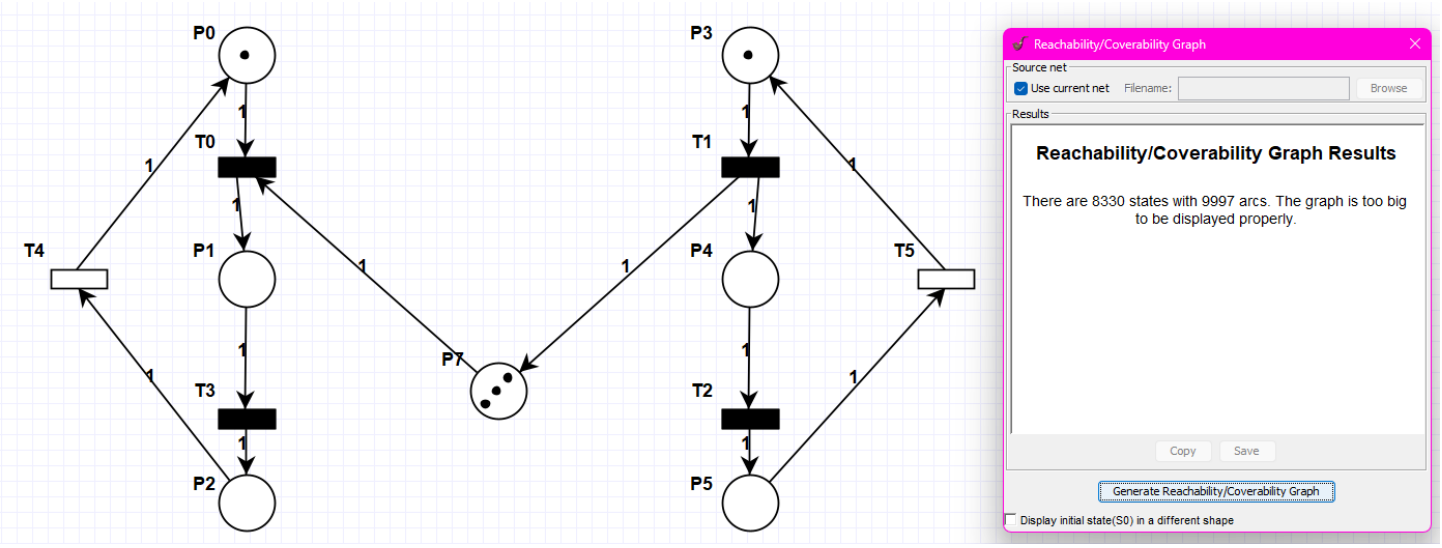
O rozmiarze bufora mówi równanie M(P6) + M(P7) = 3.

6 Zadanie 5

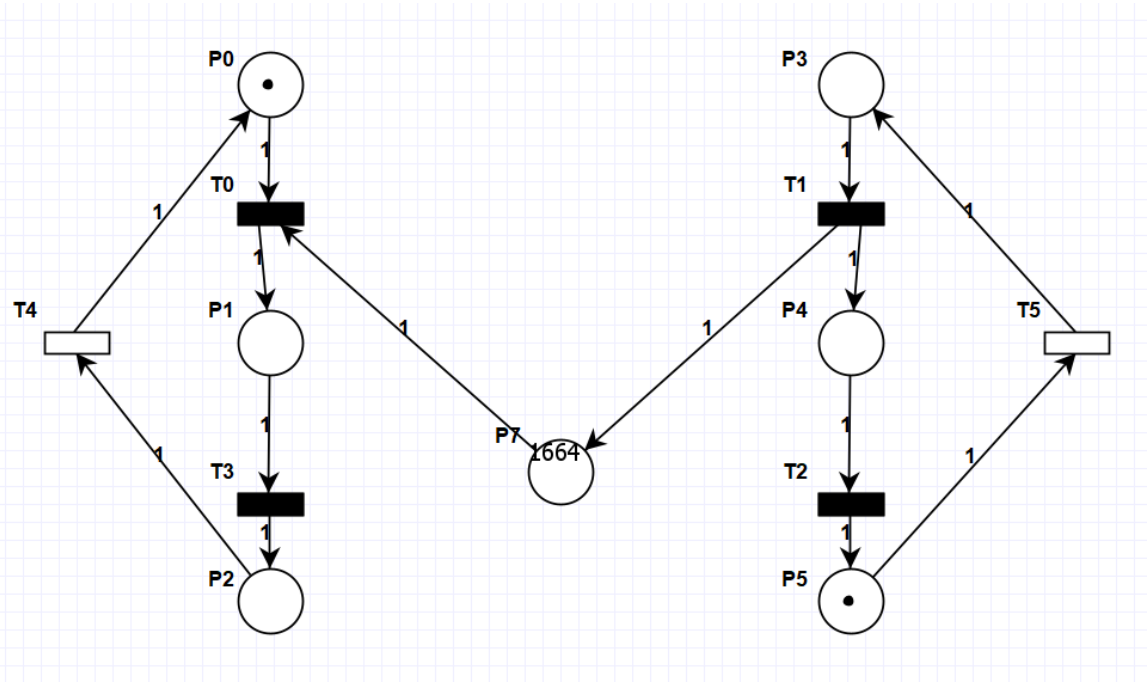
6.1 Treść zadania

Stworzyć symulacje problemu producenta i konsumenta z nieograniczonym buforem. Dokonać analizy niezmienników. Zaobserwować brak pełnego pokrycia miejsc.

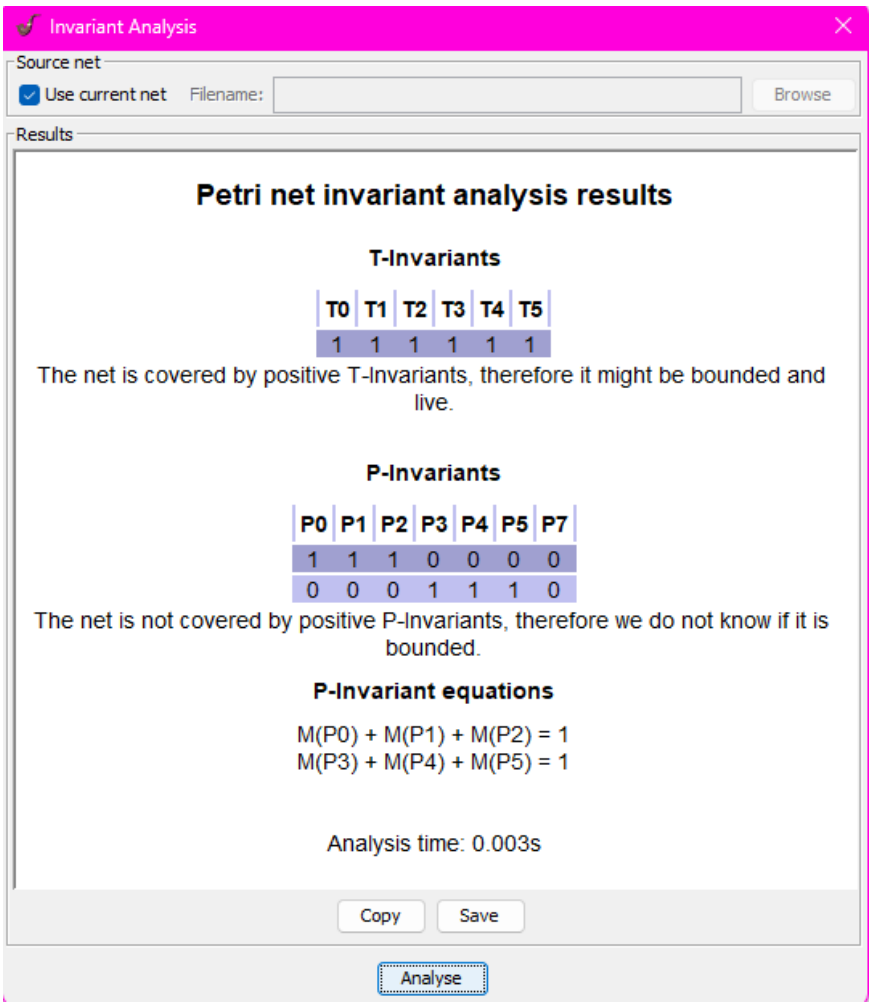
6.2 Utworzona symulacja z nieograniczonym buforem



Poniżej możemy zauważyć powstanie 1664 tokenów w nieograniczonym buforze. Ta liczba będzie rosła w nieskończoność wraz z realizacją kolejnych cykli symulacji.



6.3 Analiza niezmienników



Analizując niezmienniki przejść możemy zauważyć, że sieć jest pokryta w całości pozytywnymi przejściami. W związku z tym, sieć jest żywa, ponieważ wszystkie przejścia mogą być wykonane. Natomiast dana sieć nie jest odwracalna, gdyż nie można dojść ponownie do stanu początkowego, z którego zaczęliśmy symulację. Dzieje się tak przez to, że nasz nieograniczony bufor co chwilę zwiększa liczbę przechowywanych tokenów.

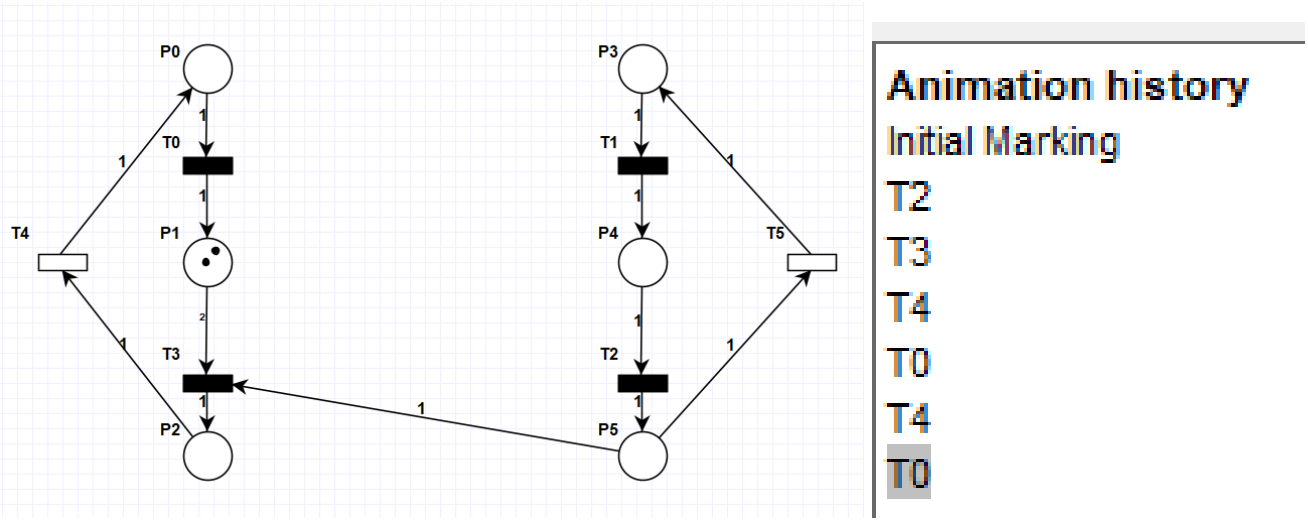
W niezmiennikach miejsc zauważyć możemy wartość 0 przy P7, które jest naszym nieograniczonym buforem. Dzieje się tak, ponieważ możemy uzyskać w nim dowolną wartość tokenów, co skutkuje tym, że nasza sieć nie jest ograniczona. W związku z tym, nie jesteśmy w stanie zagwarantować pełnego pokrycia miejsc.

7 Zadanie 6

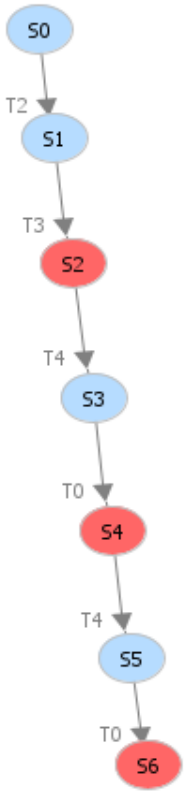
7.1 Treść zadania

Zasymulować prosty przykład ilustrujący zakleszczenie. Wygenerować graf osiągalności i zaobserwować znakowania, z których nie można wykonać przejść. Zaobserwować właściwości sieci w ”State Space Analysis”.

7.2 Zmodyfikowany przykład z zadania 5 ilustrujący zakleszczenie

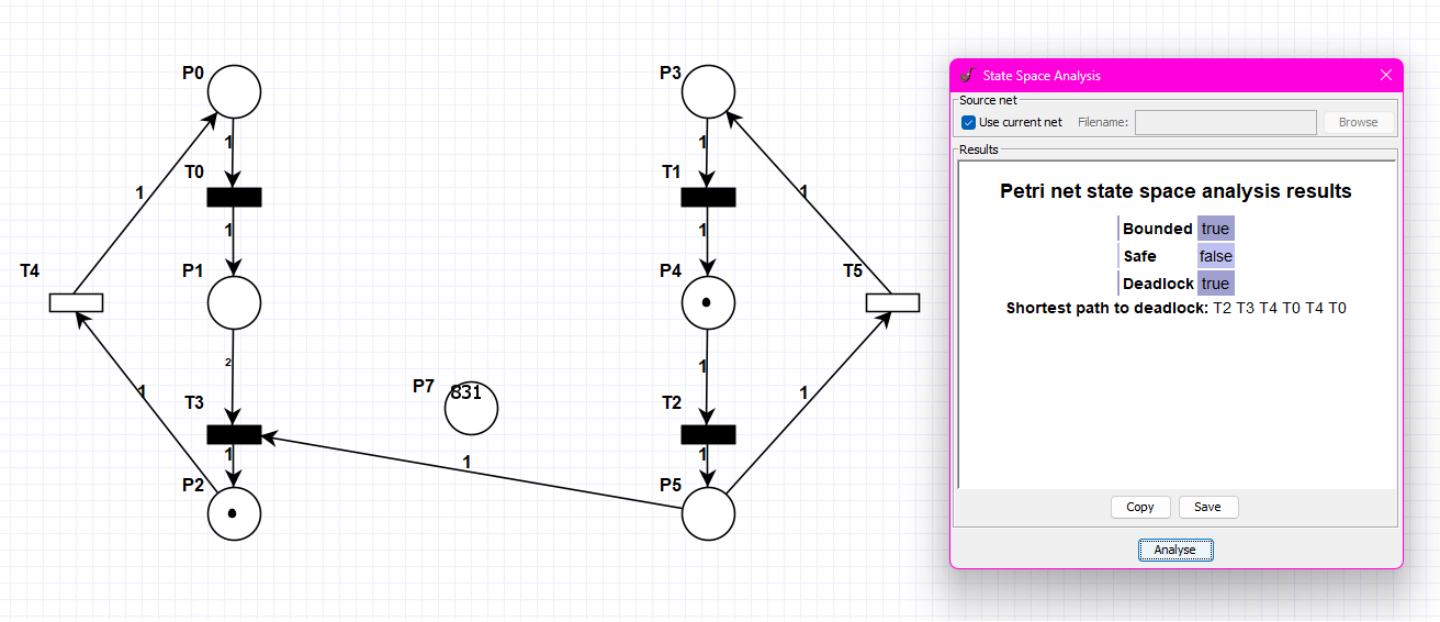


7.3 Graf osiągalności



Z grafu osiągalności możemy zauważyć znakowanie S6, z którego nie jesteśmy w stanie wykonać przejścia. Dzieje się tak, ponieważ nasza symulacja utknęła (zakleszczyła się). Oznacza to, że w momencie gdy symulacja osiągnie stan S6, to nie będzie w stanie wykonać następnego przejścia, a co za tym idzie, pozostanie w tym stanie na zawsze (deadlock).

7.4 Właściwości sieci w State Space Analysis



Z właściwości sieci w "State Space Analysis"możemy zauważyć, że nasza sieć jest ograniczona (bounded). To się zgadza, ponieważ liczba tokenów nie ulega zmianie w trakcie symulacji i jest stała. Sieć nie jest bezpieczna, ponieważ w którymś z miejsc, jesteśmy w stanie uzyskać więcej niż jeden token (co widać nawet na zrzucie ekranu z punktu 7.2 w miejscu P1, gdzie są 2 tokeny). Na koniec zaobserwujmy, że osiągnęliśmy to co chcieliśmy - zakleszczenie (deadlock), wraz z podaną najkrótszą ścieżką do tego stanu - T2, T3, T4, T0, T4, T0.