AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Determinizm w systemach współbieżnych

Dr hab. inż. Andrei Karatkevich, prof. AGH
Katedra Informatyki Stosowanej

Seminarium Katedry Informatyki Stosowanej,
22 września 2021

Agenda

- Determinizm w systemie i modelu
- Współbieżność a determinizm
- Determinizm słaby i silny
- Zaproponowane podejście do konstruowania współbieżnych algorytmów sterowania
- Podsumowanie

Determinizm

- *System deterministyczny*: znając stan systemu w danym momencie i oddziaływania zewnętrzne w każdym przyszłym momencie, możemy bezbłędnie prognozować stan systemy w dowolnym czasie. Brak losowości.
- *Model deterministyczny*: model który dla danego stanu początkowego i danych wartości na wejściach przejawia jedyne możliwe zachowanie (nie może zareagować na rożne sposoby) (E.A. Lee)

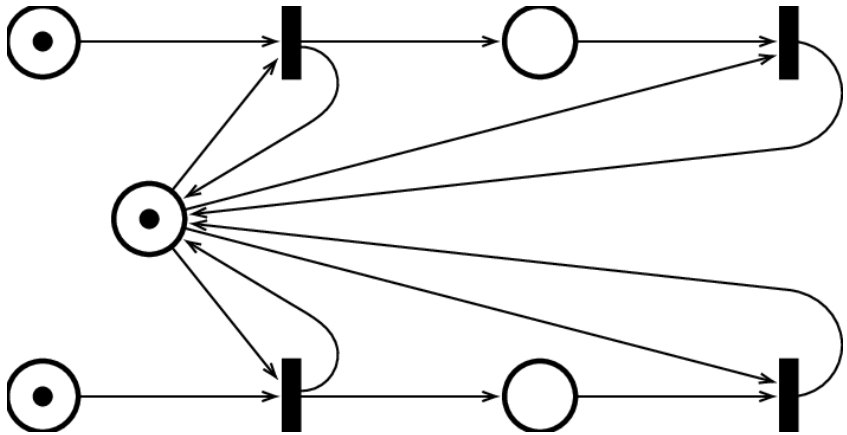
Lee, E.A.; Seshia, S.A. *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*, 2nd ed.; The MIT Press: Cambridge, MA, USA, 2016.

Lee, E.A. *Plato and the Nerd*. The Creative Partnership of Humans and Technology; MIT Press: Cambridge, MA, USA, 2017.

Determinizm

- Deterministyczny system, niedeterministyczny model:

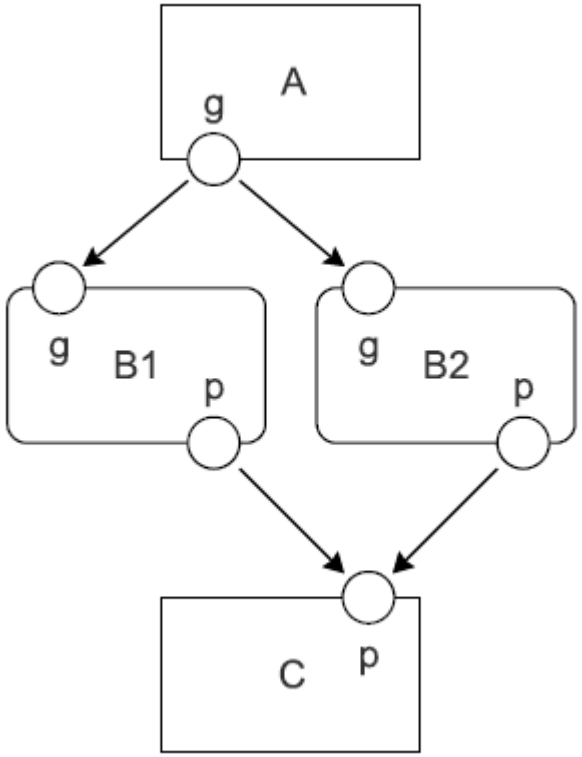
Sieć Petriego (nieinterpretowana), opisująca powiązania deterministycznych procesów



Determinizm

- Niedeterministyczny system, deterministyczny model:

Model wymusza kolejność zdarzeń, która w systemie może być inna



Determinizm

- Niedeterministyczny system, deterministyczny model: zdarzenia jednoczesne w modelu (czas logiczny) mogą nie być jednoczesne w systemie (czas fizyczny)

- Deterministyczny algorytm współbieżny wykonywany na jednym procesorze: kolejność wykonania „współbieżnych" fragmentów

- Sterowanie obiektem fizycznym: logicznie jednoczesne zdarzenia mogą nie być jednoczesne w obiekcie (zamknięcie drzwi autobusu i początek ruchu mogą być jednoczesne logicznie, ale nie fizycznie)

(Sirjani, M.; Lee, E.A.; Khamespanah, E. *Model Checking Software in Cyberphysical Systems*. In Proceedings of the IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, 13–17 July 2020; pp. 1017–1026.)

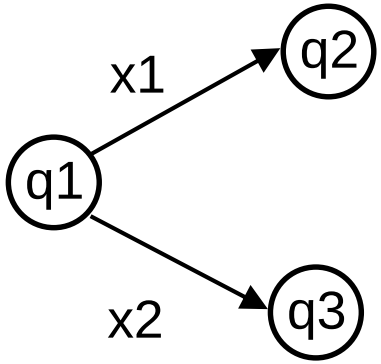
Determinizm

- *Algorytm deterministyczny*: algorytm, którego działanie jest całkowicie zdeterminowane przez warunki początkowe (dla takich samych danych wejściowych zawsze generuje taki sam wynik – ale taki może być i algorytm probabilistyczny!)
- *Deterministyczny automat skończony*: automat w którym dla każdego stanu i każdej wartości możliwe jest najwyżej jedyne przejście do innego stanu
- *„Określony" automat skończony* (*determinate state machine*): automat który, zaczynając od danego stanu początkowego, dla danej sekwencji wartości wejściowych generuje w jednoznaczny sposób sekwencję wyjściową

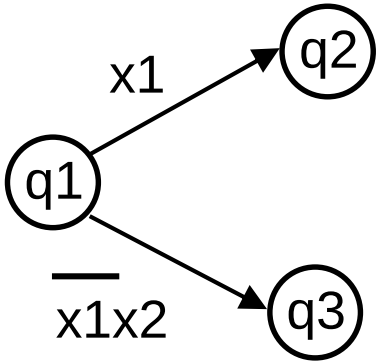
Deterministyczny automat skończony

- Funkcja przejść: $\delta : Q \times \Sigma \rightarrow Q$

 - gdzie Q – zbiór stanów, Σ – zbiór możliwych wartości wejściowych



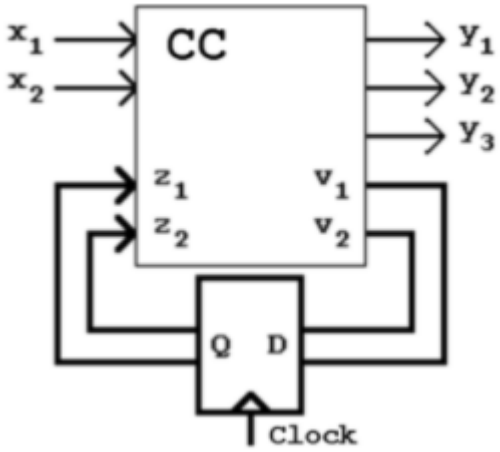
Niedeterministyczne przejścia!

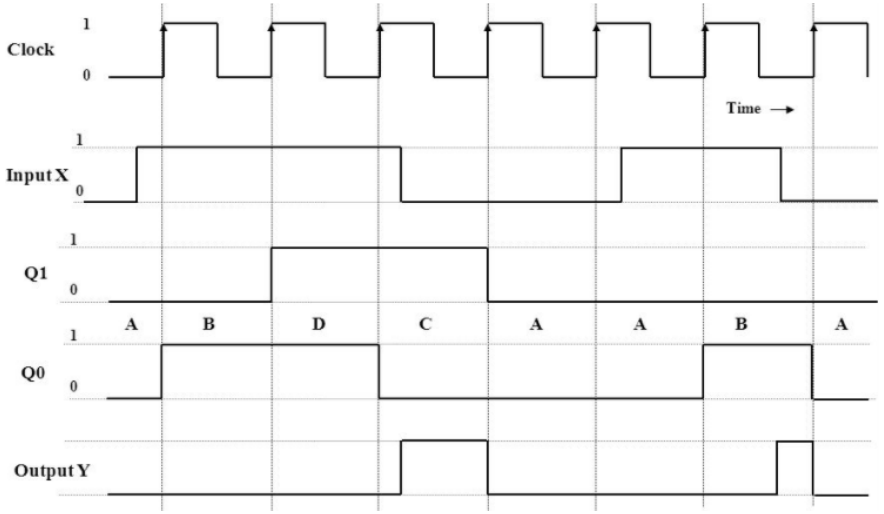


Deterministyczne przejścia

Czy naprawdę deterministyczny?

- Jeśli dane „czekają” na pobranie, nie zmieniając się (e. g. obliczeniowy algorytm deterministyczny) – tak!
- Jeśli chodzi o system synchroniczny – tak!





A jeśli asynchroniczny?

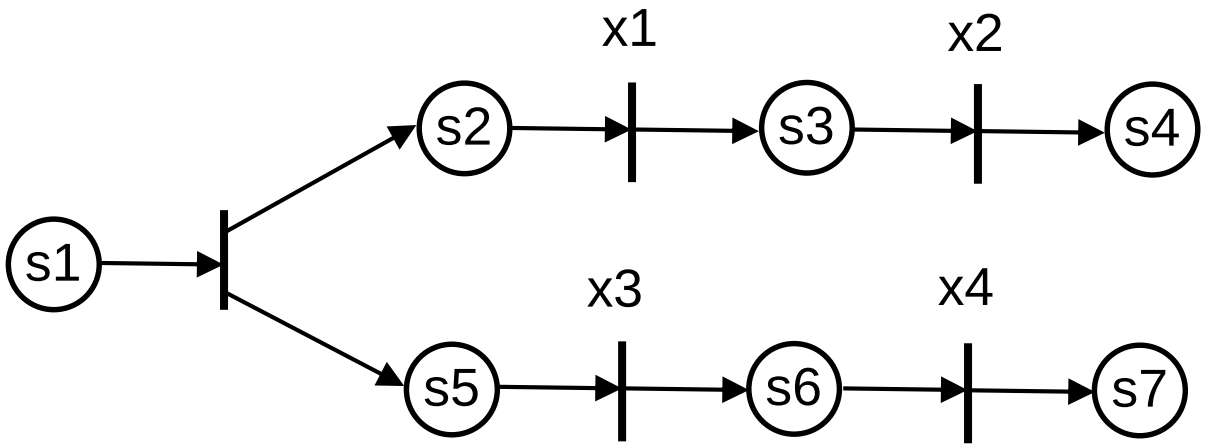
Czas ciągły, nie dyskretny:

- Jak długo potrwa zmiana stanu i wartości na wyjściach?
- Co będzie jeśli w trakcie przejścia zmienią się wartości na wejściach?
- Mogą pojawiać się skokowe zmiany wartości na wyjściach

Czy to jest determinizm (jedyne możliwe zachowanie)? Zależy, jak rozumiemy zachowanie.

Systemy współbieżne

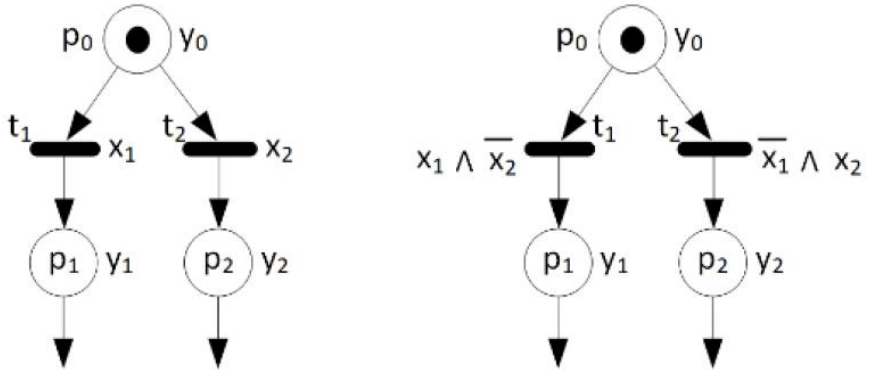
- Synchroniczne – determinizm jest łatwo osiągalny
- Asynchroniczne – jest z tym problem:



Jeśli x1x2x3x4 = 1,
s4 czy s7 zostanie osiągnięte jako pierwsze?

Dylemat: współbieżność a determinizm

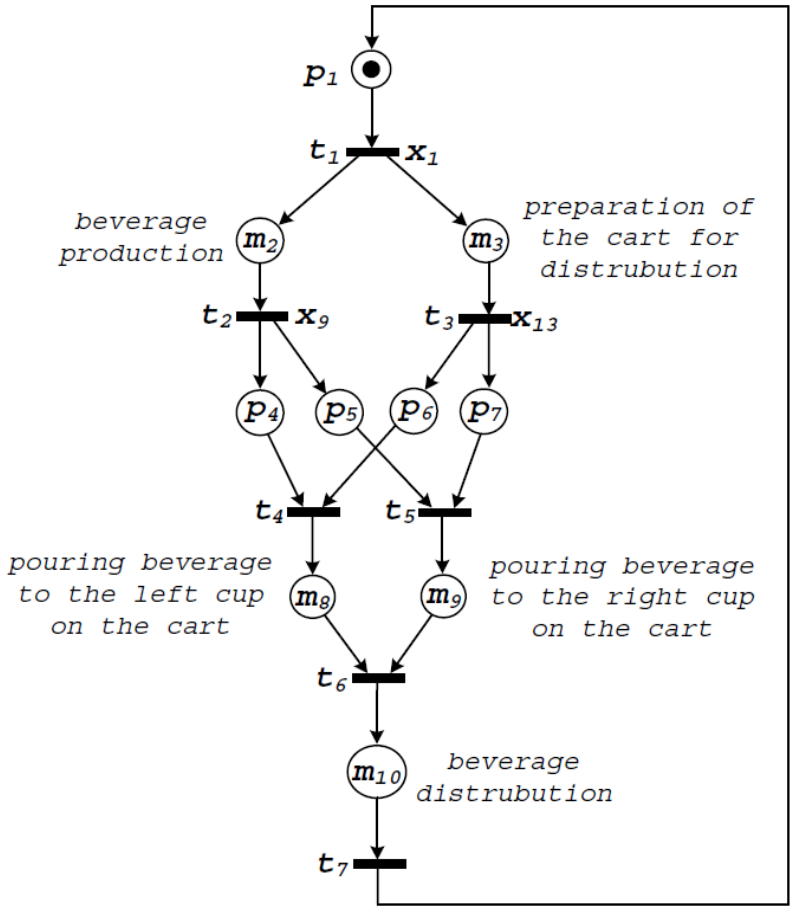
- Za dużo współbieżności – system przestaje być deterministyczny
- Za dużo determinizmu (e.g. priorytety wszędzie) – system przestaje być współbieżny (sprowadza się do automatu skończonego)



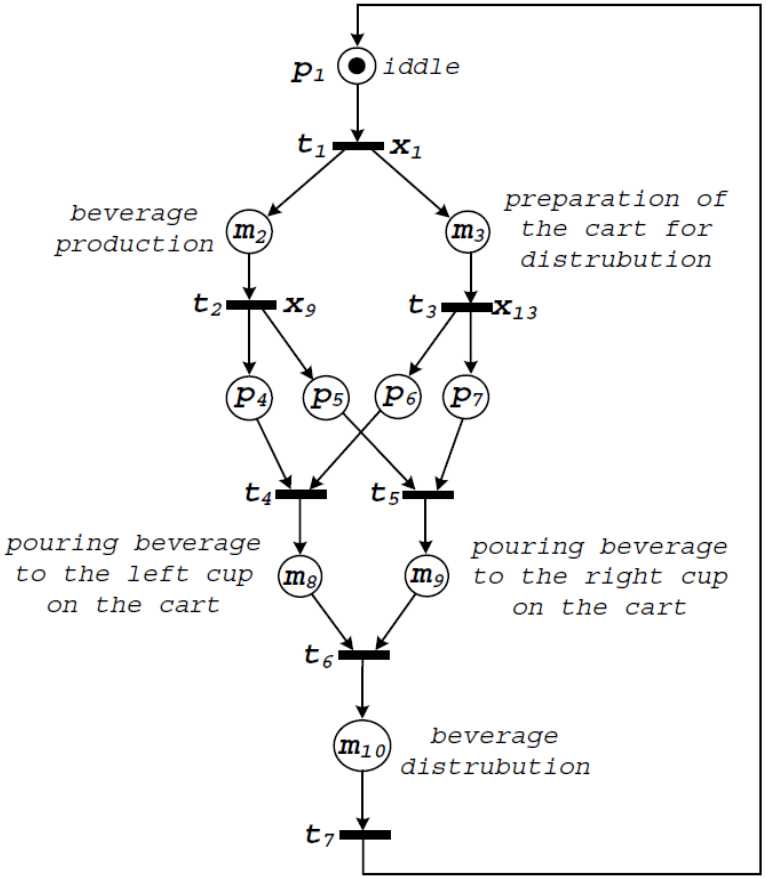
Szukamy złotego środka

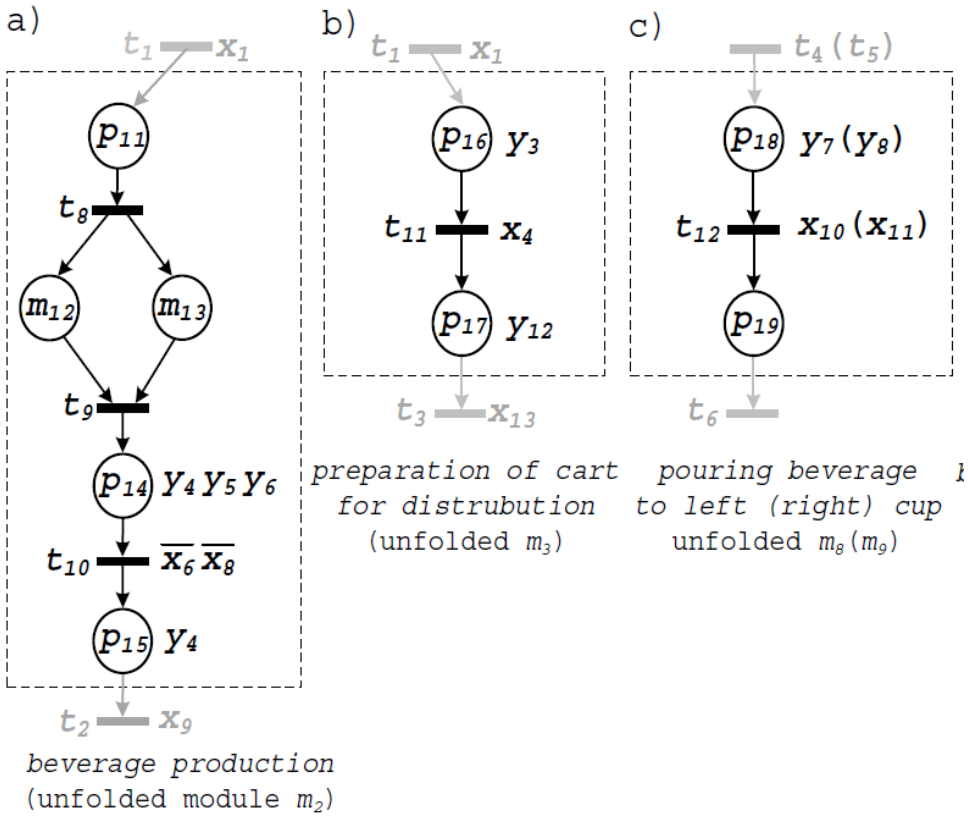
Model, którego używamy

- Interpretowana sieć Petriego sterowania (control interpreted Petri net, CIPN)
- Dozory (warunki) są przypisane do tranzycji
- Akcje są skojarzone z miejscami
- Możliwe są makromiejsca – skojarzone z sieciami na niższym poziomie hierarchii



Hierarchiczna CIPN





Niektóre potrzebne pojęcia

- Moduł: sieć CIPN z miejscem wejściowym p_{in} i miejscem wyjściowym p_{out} ($\bullet p_{in} = \emptyset$, $p_{out}\bullet = \emptyset$)
- Hierarchiczna CIPN: CIPN z wydzielonym podzbiorem miejsc (nazywanych makromiejscami) takich, że z każdym z nich jest skojarzony moduł; sieć zachowuje się tak, jak gdyby moduł podstawić zamiast makromiejsca

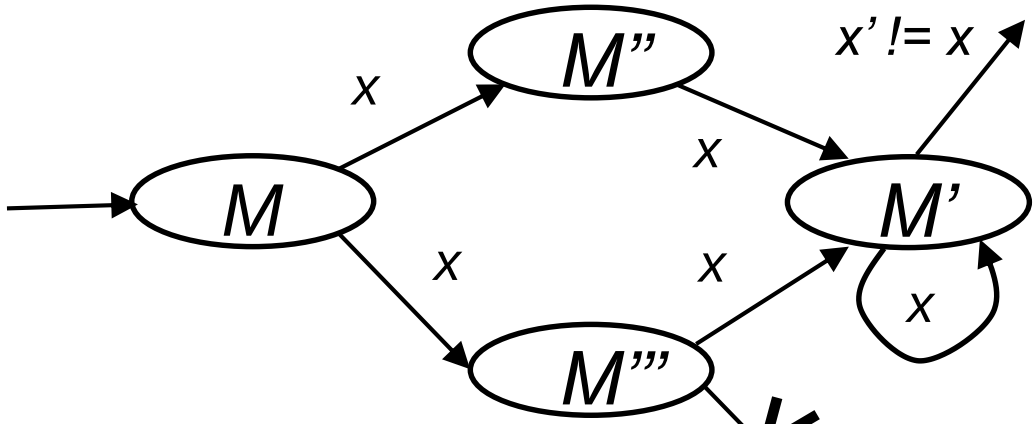
- Stabilne znakowanie (względem *x*): znakowanie *M*, które się nie zmienia, kiedy sygnały wejściowe mają wartości *x*
- Tu *x* rozumiemy jako wektor binarny wartości sygnałów wejściowych

Determinizm słaby i silny

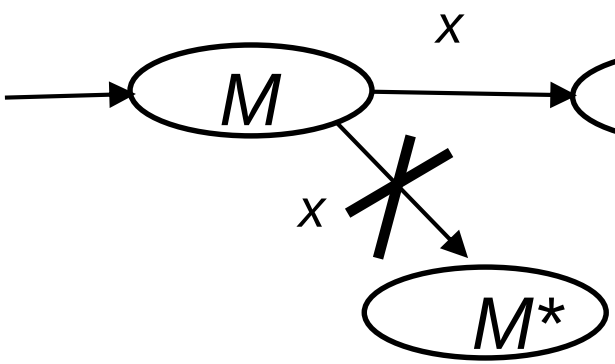
- *Słaby determinizm*: sieć CIPN jest słabo deterministyczna, jeśli dla danego znakowania *M* i danych wartości *x* sygnałów wejściowych sieć przechodzi do stabilnego znakowania *M'* w sposób jednoznaczny.
- *Silny determinizm*: sieć CIPN jest silnie deterministyczna, jeśli ona jest słabo deterministyczna oraz dla każdego osiągalnego znakowania *M* i danych wartościach *x* na wejściach tylko jedno następne znakowanie jest możliwe.

Determinizm słaby i silny

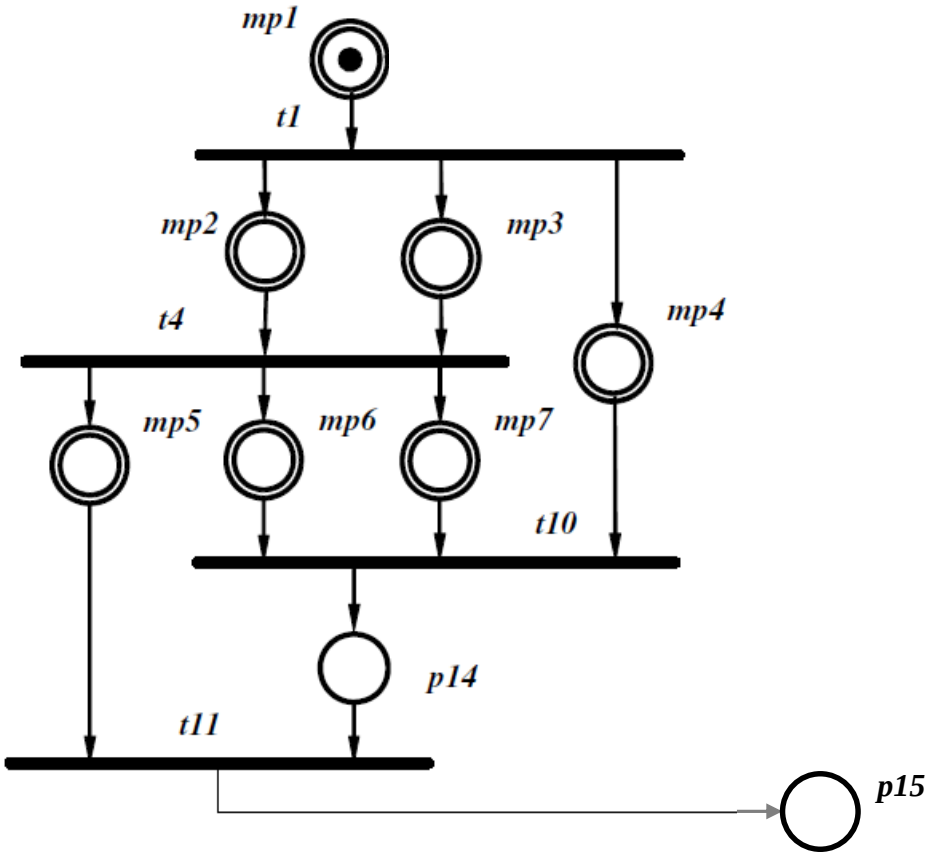
Słaby determinizm:



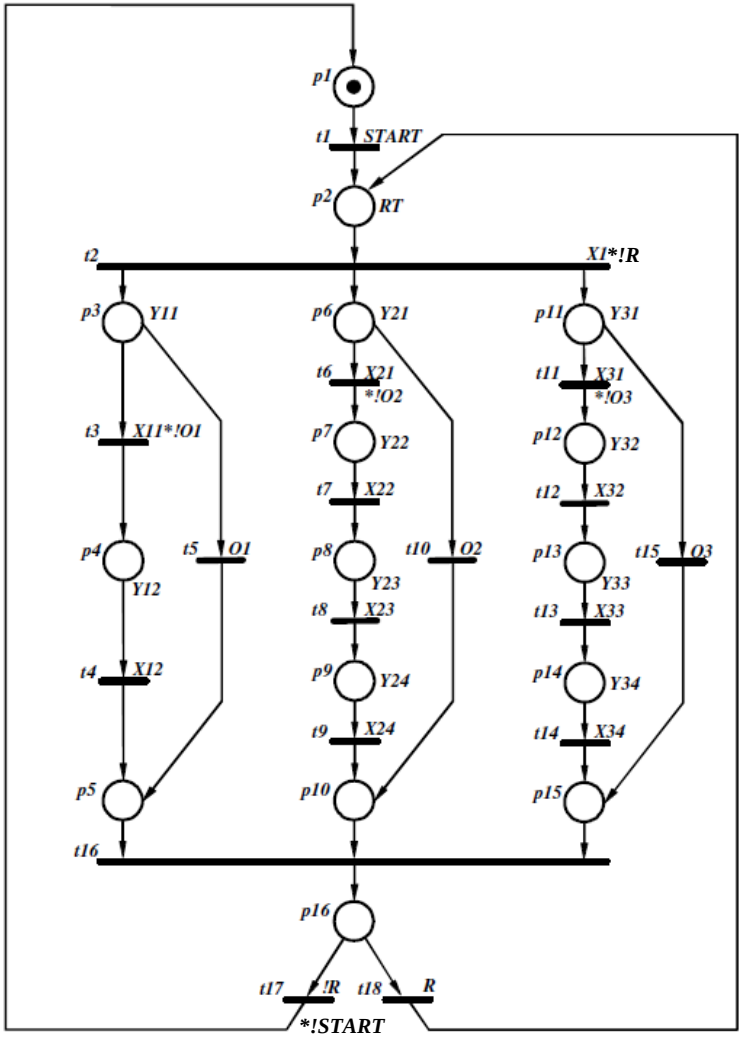
Silny determinizm:



Silny determinizm (przykład bez dozorów)



Słaby determinizm (przykład)



Słaby determinizm

Twierdzenie 1. Sieć CIPN jest słabo deterministyczna, jeśli (1) dla każdych dwóch tranzycji mających wspólne miejsce wejściowe ich dozory nie mogą być jednocześnie spełnione oraz (2) dla każdego znakowania *M* i stabilnych wartości wejściowych *x* sieć przechodzi do stabilnego znakowania (cykliczna zmiana znakowań nie jest możliwa bez zmiany wartości wejściowych).

Silny determinizm

Twierdzenie 2. Sieć CIPN jest silnie deterministyczna, jeśli i tylko jeśli ona jest słabo deterministyczna oraz dla każdych dwóch tranzycji t_1 i t_2 takich że ${}^\bullet t_1 \neq t_1{}^\bullet$, ${}^\bullet t_2 \neq t_2{}^\bullet$, $t_1 \neq t_2$, dla których istnieje osiągalne znakowanie M, takie że t_1 i t_2 są aktywne w M, ich dozory nie mogą być jednocześnie spełnione.

Determinizm a hierarchia

Twierdzenie 3. Niech $N = (P, T, F, M_0, X, Y)$ – słabo deterministyczna CIPN. Wybierzmy podzbiór miejsc $P_m \subseteq P$ i zamieńmy je na makromiejsca, kojarząc z każdym z nich słabo deterministyczny moduł. Uzyskana w wyniku tego hierarchiczna sieć *HN* jest słabo deterministyczna.

- Łatwo (przez indukcję) generalizuje się na wielopoziomowe hierarchiczne sieci.
- Nie działa z silnym determinizmem – jeśli moduły są silnie deterministyczne, *HN* może być słabo deterministyczna!

W poszukiwaniu kompromisu

- Pełny silny determinizm dla współbieżnego systemu asynchronicznego:
  - Jest ciężki (czasami niemożliwy) do osiągnięcia
  - Wypacza zalety współbieżności i asynchroniczności
- Z drugiej strony, determinizm jest wskazany, ponieważ:
  - Pozwala mieć pewność, jak (i czy poprawnie) zachowa się system;
  - Znacznie ułatwia analizę i weryfikację

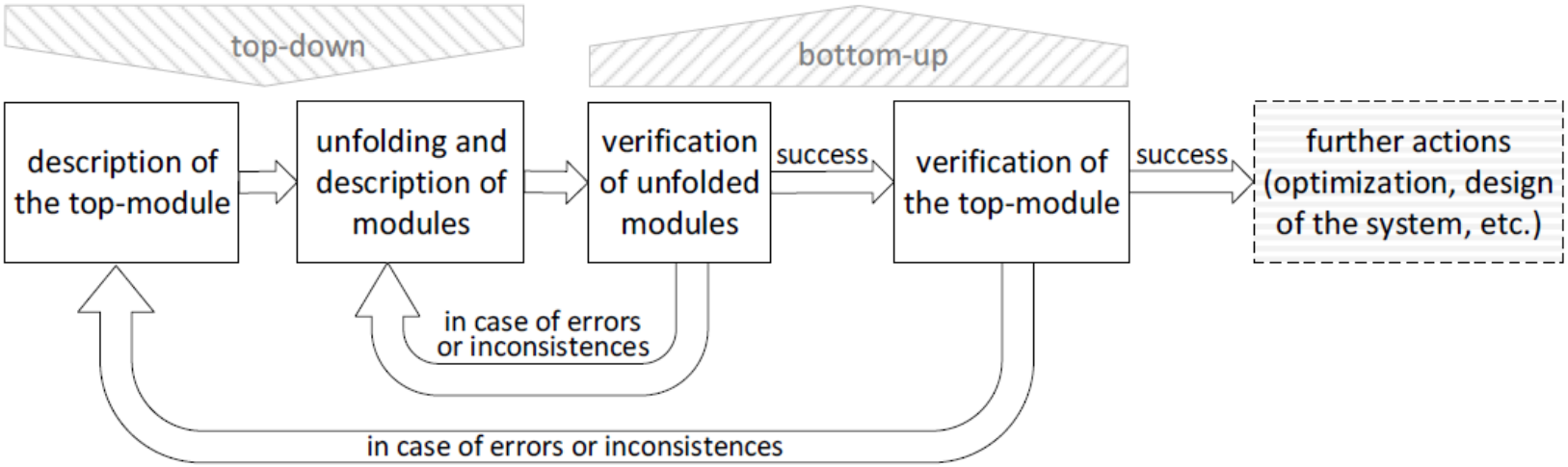
Pomysł na kompromis

- Moduły – silnie deterministyczne
- System jako taki – słabo deterministyczny
- Ograniczać bezpośrednią komunikację między współbieżnymi modułami
- W miarę możliwości brak silnego determinizmu „chować” we współdziałaniu między siecią a jej makromiejscami

Zaproponowane podejście do konstruowania algorytmów sterowania współbieżnego

- Specyfikacja „top-down”:
  - Określenie górnego poziomu sieci (makromiejsca jako „czarne skrzynki”);
  - Sekwencyjne rozwinięcie makromiejsc w moduły;
  - Moduły silnie deterministyczne
- Weryfikacja „bottom-up”:
  - Weryfikacja modułów
  - Weryfikacja górnego poziomu sieci

Zaproponowane podejście do konstruowania



Główne zalety podejścia:

- Ułatwione (bez eksplozji stanów) analiza i weryfikacja systemu na każdym poziomie
- Ułatwione lokalizacja i naprawianie błędów

Analiza i weryfikacja modułów

- Analiza żywotności, bezpieczeństwa i nadmiarowości, wykrywanie zakleszczeń
  - Bezpośrednia analiza grafu osiągalności
  - Redukcja sieci
  - Stubborn sets (zakleszczenia)
  - Narzędzia: PIPE (a tool for creating and analysing of Petri nets), CPN Tools
- Analiza specyficznych dla danego systemu wymagań, wyrażonych w logice temporalnej (LTL, CTL)
  - Model checking
  - Ważne narzędzie: nuXmv model checker

Podsumowanie

- Proponujemy rozróżnienie rodzajów determinizmu
- Zaproponowane podejście do konstruowania algorytmów sterujących ułatwia zrozumienie ich działania i zmniejsza prawdopodobieństwo błędów
- Zaproponowane podejście ułatwia weryfikację systemu, lokalizację i naprawianie błędów
- Fokus na determinizmie zapewnia przewidywalne zachowanie systemu
- Kompromis (silny determinizm dla modułów, słaby dla całego systemu) pozwala wykorzystywać zalety zarówno determinizmu jak i współbieżności

Publikacje

- Wiśniewski R., Grobelna I., Karatkevich A. *Determinism in Cyber-Physical Systems Specified by Interpreted Petri Nets// Sensors*, Volume 20, 2020, 5565
- Grobelna I., Karatkevich A. *Challenges in Application of Petri Nets in Manufacturing Systems// Electronics* (MDPI) - accepted
- Szpyrka M., Karatkevich A., Baniewicz J. *Alvis approach to modelling and verification of real-time systems running on single-processor environment* - under construction…

Dziękuję za uwagę!