Programowanie współbieżne i rozproszone

WYKŁAD 10

Poprawność programów sekwencyjnych

Poprawność programu sekwencyjnego wyrażana jest za pomocą zdania o postaci:

$$\{p\}S\{q\}$$

gdzie:

- S jest programem,
- p i q są asercjami nazywanymi odpowiednio: warunkiem wstępnym i warunkiem końcowym.

Poprawność programów sekwencyjnych

- Warunek wstępny określa warunki jakie spełniają dane wejściowe.
- Warunek końcowy określa warunki jakie powinny spełniać wyniki działania programu.

Poprawność programów sekwencyjnych

Częściowa poprawność programu:

Każde kończące się wykonanie programu *S* z danymi wejściowymi spełniającymi asercję *p* kończy się z danymi wyjściowymi spełniającymi asercję *q*.

Całkowita (pełna) poprawność programu:

Każde wykonanie programu *S* z danymi wejściowymi spełniającymi asercję *p* kończy się a dane wyjściowe spełniają asercję *q*.

Poprawność programów sekwencyjnych

- W przypadku częściowej poprawności nie bierze się pod uwagę, czy obliczenia programu *S* kończą się, czy nie.
- Kończenie się obliczeń programu dla wszystkich prawidłowych danych wejściowych, tj. spełniających asercję *p*, nazywane jest warunkiem stopu.

Poprawność programów sekwencyjnych

Dowód pełnej poprawności programu sekwencyjnego:

- Udowodnienie, że program jest częściowo poprawny.
- Udowodnienie, że program spełnia warunek stopu.

Poprawność programów współbieżnych

Poprawność programu współbieżnego jest trudniejsza do udowodnienia niż programu sekwencyjnego

Poprawność programów współbieżnych

W przypadku programu współbieżnego należy wykazać dodatkowe własności:

- Własności bezpieczeństwa.
- Własność żywotności.

Poprawność programów współbieżnych

Własności bezpieczeństwa obejmują warunki, które powinny być zawsze spełnione, tj. dla wszystkich możliwych realizacji procesów współbieżnych.

Własności bezpieczeństwa:

- wzajemne wykluczanie,
- brak blokady/zakleszczenia.

Poprawność programów współbieżnych

Blokada – sytuacja, w której jeden lub większa liczba procesów współbieżnych się nie kończy.

Zakleszczenie – rodzaj blokady dotyczącej zbioru procesów. Zakleszczenie pojawia się wtedy, gdy każdy z procesów należących do danego zbioru procesów jest wstrzymany w oczekiwaniu na zdarzenie, które może być spowodowane tylko przez jakiś inny proces z tego zbioru.

Poprawność programów współbieżnych

Własność żywotności obejmuje warunki, które powinny być ostatecznie spełnione, tj. jeśli powinno zajść pewne zdarzenie, to dla każdej możliwej realizacji procesów współbieżnych w pewnym momencie ono rzeczywiście zachodzi.

Własność żywotności:

- brak zagłodzenia.

Poprawność programów współbieżnych

Z własnością żywotności związana jest tzw. uczciwość.

Rodzaje uczciwości:

- <u>Uczciwość słaba</u>: jeśli proces nieprzerwanie zgłasza żądanie, to kiedyś będzie ono obsłużone.
- <u>Uczciwość mocna</u>: jeśli proces zgłasza żądanie nieskończenie wiele razy, to kiedyś będzie ono obsłużone.

Dowodzenie poprawności za pomocą diagramów stanów

Zbiór stanów osiągalnych – zbiór zawierający te i tylko te stany, które mogą wystąpić w dowolnym obliczeniu programu współbieżnego.

Diagram stanów konstruuje się stopniowo, począwszy od stanu początkowego, analizując wszystkie możliwe kolejne stany. Jeśli otrzymany w ten sposób stan już występuje w diagramie, to nie jest on powtarzany.

Dowodzenie poprawności za pomocą diagramów stanów

Proces *A*	**Proces *B***

pętla
{
a1: sekcja lokalna
a2: **semafor**.wait()
a3: sekcja krytyczna
a4: **semafor**.signal()
}
	pętla
	{
	b1: sekcja lokalna
	b2: **semafor**.wait()
	b3: sekcja krytyczna
	b4: **semafor**.signal()
	}

a1, a2, a3, a4 – stany procesu *A*
b1, b2, b3, b4 – stany procesu *B*

Specyfikacja własności poprawności w języku logiki

Przykład:

Proces *A*	Proces *B*

pętla
{
a1: sekcja lokalna
a2: **semafor**.wait()
a3: sekcja krytyczna
a4: **semafor**.signal()
}
	pętla
	{
	b1: sekcja lokalna
	b2: **semafor**.wait()
	b3: sekcja krytyczna
	b4: **semafor**.signal()
	}

Program spełnia własność wzajemnego wykluczania w tych stanach, w których prawdziwa jest formuła:

$$\neg\left(a3 \wedge a4\right)$$

Sieci Petriego

Sieci Petriego są graficznym i matematycznym narzędziem do modelowania różnego rodzaju zjawisk, w których znaczącą rolę odgrywają czynności wykonywane współbieżnie.

Sieci miejsc i tranzycji (PT-sieci)

$$\boldsymbol{N=(P,T,F,K,W,M_0)}$$

- *P* - niepusty, skończony zbiór miejsc.
- *T* - niepusty, skończony zbiór tranzycji.
- *F* - relacja opisująca zbiór łuków skierowanych sieci (od miejsc do tranzycji i od tranzycji do miejsc).
- *K* - funkcja określająca pojemność miejsc (dla każdego miejsca określa maksymalną liczbę znaczników, które miejsce może zawierać).

Sieci miejsc i tranzycji (PT-sieci)

$$\boldsymbol{N=(P,T,F,K,W,M_0)}$$

c.d.:

- W - funkcja określająca wagę (krotność) łuków skierowanych.
- M_0 - funkcja określająca znakowanie początkowe miejsc (dla każdego miejsca określa liczbę znaczników, które miejsce zawiera na początku).

Sieci miejsc i tranzycji (PT-sieci)

$$N=(P,T,F,K,W,M_0)$$

$$P=\{p_{1,}p_{2,}\ldots,p_k\}$$

$$T=\{t_{1,}t_{2,}\ldots,t_q\}$$

$$F\subseteq(P\times T)\cup(T\times P)$$

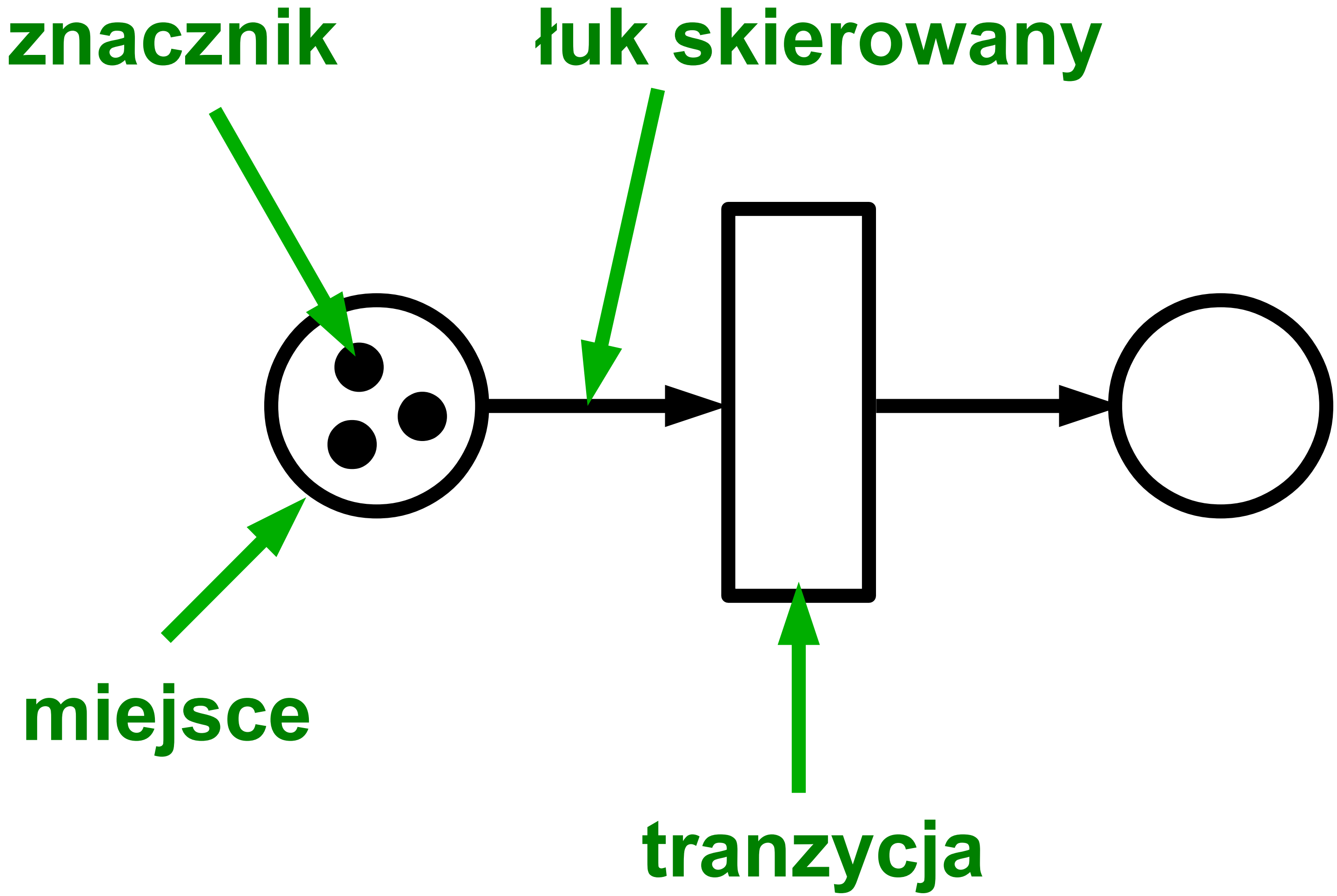
$$K:P\rightarrow N\cup\{\infty\}$$

$$W:F\rightarrow N$$

$$M_0:P\rightarrow N\cup\{0\}$$

$$\underset{p\in P}{\forall} M_0(p)\leqslant K(p)$$

Sieci miejsc i tranzycji (PT-sieci)



Sieci miejsc i tranzycji (PT-sieci)

- **Znakowanie sieci** – dowolne odwzorowanie

$$M : P \rightarrow N \cup \{0\}$$

takie, że spełniony jest warunek:

$$\underset{p \in P}{\forall} M(p) \leqslant K(p)$$

Sieci miejsc i tranzycji (PT-sieci)

- Każdemu miejscu $p \in P$ można przyporządkować parę liczb

$$\left(t^{-}(p), t^{+}(p)\right)$$

takich, że:

- pierwszy element pary jest liczbą znaczników odbieranych przez tranzycję *t* miejscu *p*,
- drugi element pary jest liczbą znaczników dodawanych przez tranzycję *t* miejscu *p*,

Sieci miejsc i tranzycji (PT-sieci)

$$t^{-}(p)=\begin{cases} W(p,t) & gdy\,(p,t)\in F \\ 0 & w\ przeciwnym\ razie \end{cases}$$

$$t^{+}(p)=\begin{cases} W(p,t) & gdy\,(t,p)\in F \\ 0 & w\ przeciwnym\ razie \end{cases}$$

Sieci miejsc i tranzycji (PT-sieci)

$$\boldsymbol{t}^{-} = [t^{-}(p_1), t^{-}(p_2), \ldots, t^{-}(p_k)]$$

$$\boldsymbol{t}^{+} = [t^{+}(p_1), t^{+}(p_2), \ldots, t^{+}(p_k)]$$

$$\boldsymbol{M} = [M(p_1), M(p_2), \ldots, M(p_k)]$$

$$\boldsymbol{K} = [K(p_1), K(p_2), \ldots, K(p_k)]$$

Sieci miejsc i tranzycji (PT-sieci)

- **Słaba reguła odpalania tranzycji**:
  - Tranzycja *t* może być odpalona przy znakowaniu ***M*** wtedy i tylko wtedy, gdy:

$$\boldsymbol{t}^{-} \leqslant \boldsymbol{M}$$

- **Średnia reguła odpalania tranzycji**:
  - Tranzycja *t* może być odpalona przy znakowaniu ***M*** wtedy i tylko wtedy, gdy:

$$\left(\boldsymbol{t}^{-} \leqslant \boldsymbol{M}\right) \wedge \left(\boldsymbol{M} - \boldsymbol{t}^{-} + \boldsymbol{t}^{+} \leqslant \boldsymbol{K}\right)$$

Sieci miejsc i tranzycji (PT-sieci)

- **Mocna reguła odpalania tranzycji**:
  - Tranzycja *t* może być odpalona przy znakowaniu ***M*** wtedy i tylko wtedy, gdy:

$$\left(\boldsymbol{t}^{-} \leqslant \boldsymbol{M}\right) \wedge \left(\boldsymbol{M} + \boldsymbol{t}^{+} \leqslant \boldsymbol{K}\right)$$

Sieci miejsc i tranzycji (PT-sieci)

- **Odpalanie tranzycji**:
  - Odpalenie tranzycji *t* przy znakowaniu ***M*** skutkuje nowym znakowaniem ***M'*** takim, że:

$$\boldsymbol{M}' = \boldsymbol{M} - \boldsymbol{t}^{-} + \boldsymbol{t}^{+}$$

Sieci miejsc i tranzycji (PT-sieci)

- **Znakowanie osiągalne**:
  - Znakowanie sieci, które można otrzymać ze znakowania ***M*** poprzez odpalenie skończonej sekwencji tranzycji nazywamy znakowaniem osiągalnym.
  - Zbiór wszystkich znakowań osiągalnych ze znakowania ***M*** oznaczamy przez:

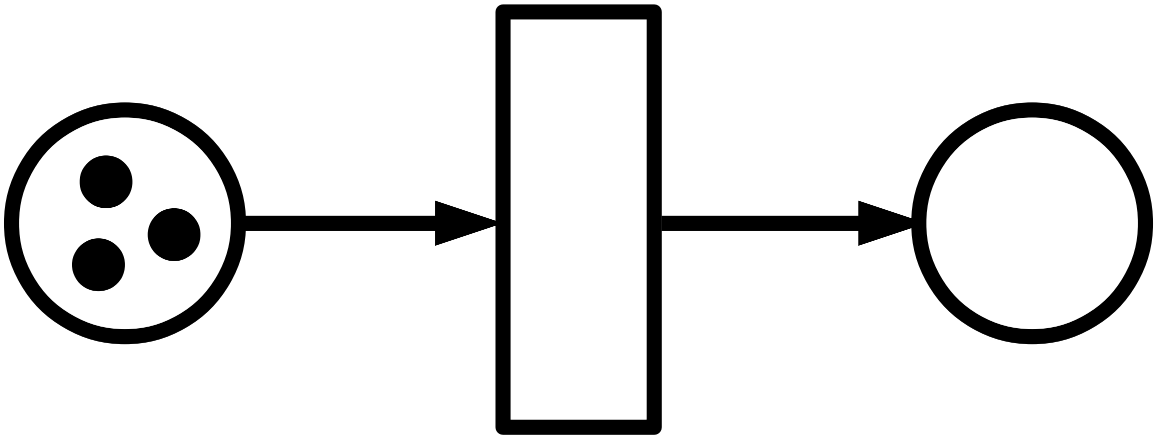
$$[\boldsymbol{M}>$$

Sieci miejsc i tranzycji (PT-sieci)

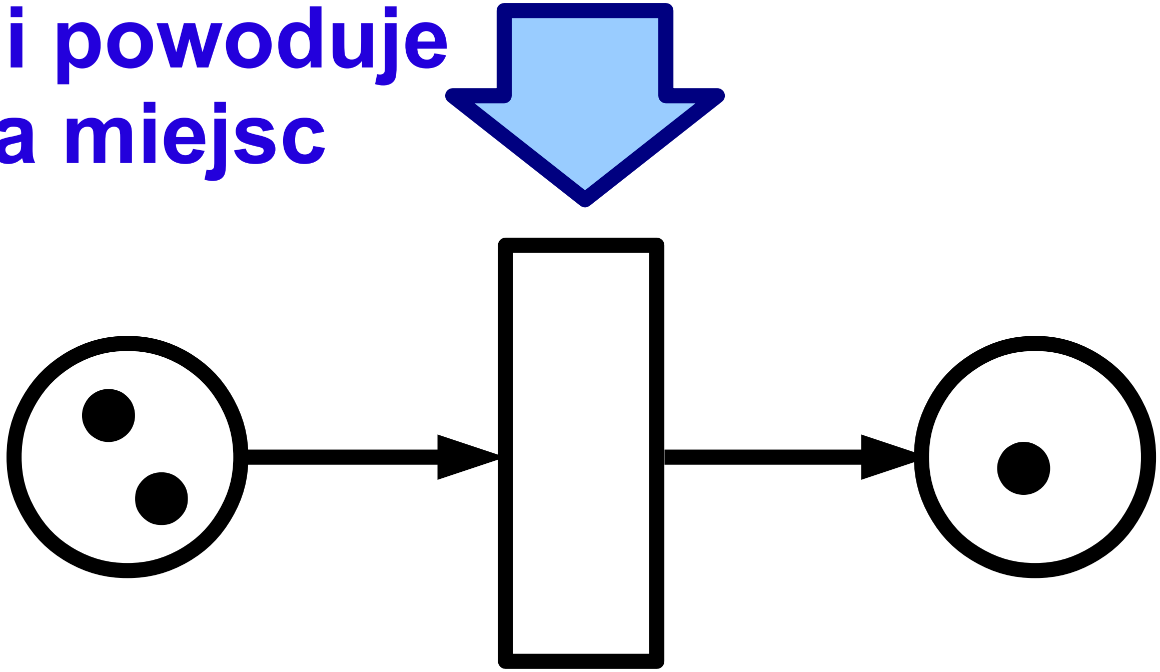
- **Żywotność sieci**:
  - Tranzycję *t* nazywamy żywą wtedy i tylko wtedy, gdy dla każdego znakowania ***M*** osiągalnego ze znakowania początkowego $\boldsymbol{M_0}$, istnieje znakowanie ***M'*** osiągalne ze znakowania ***M***, przy którym tranzycja *t* jest gotowa do odpalenia.
  - Zbiór tranzycji $T' \subseteq T$ nazywamy żywym, gdy każda tranzycja ze zbioru T' jest żywa.
  - Sieć Petriego nazywamy <u>żywą</u> wtedy i tylko wtedy, gdy zbiór *T* wszystkich jej tranzycji jest żywy.

Sieci miejsc i tranzycji (PT-sieci)

Przykład 1:

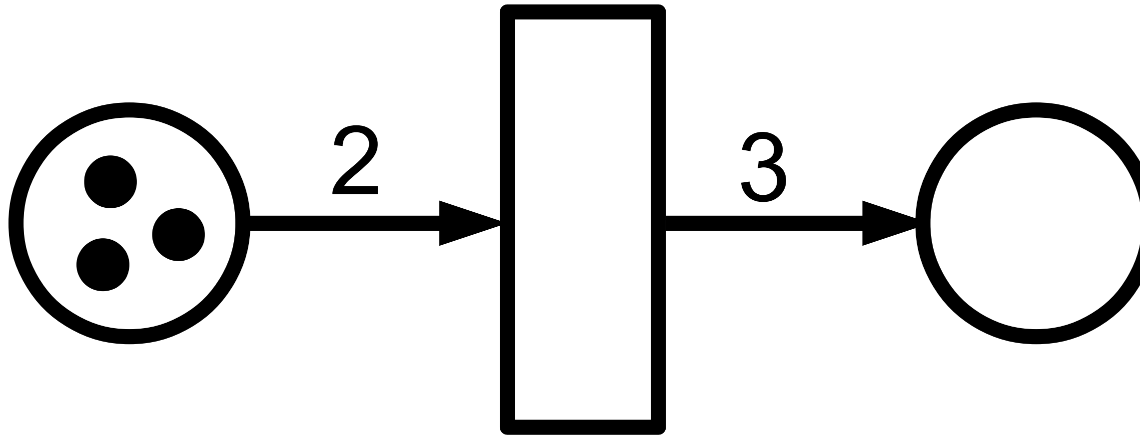


Odpalenie tranzycji powoduje zmianę znakowania miejsc

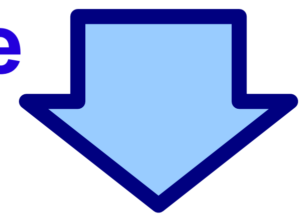


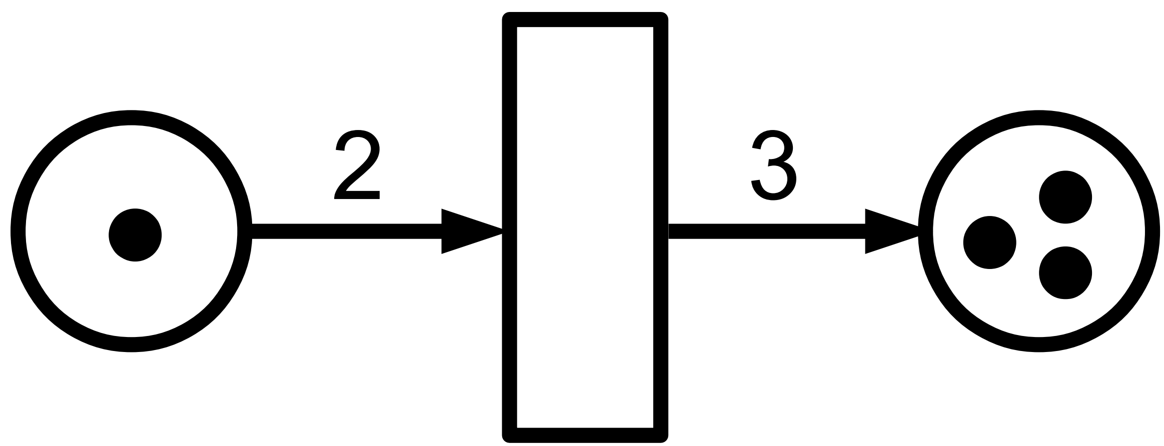
Sieci miejsc i tranzycji (PT-sieci)

Przykład 2:

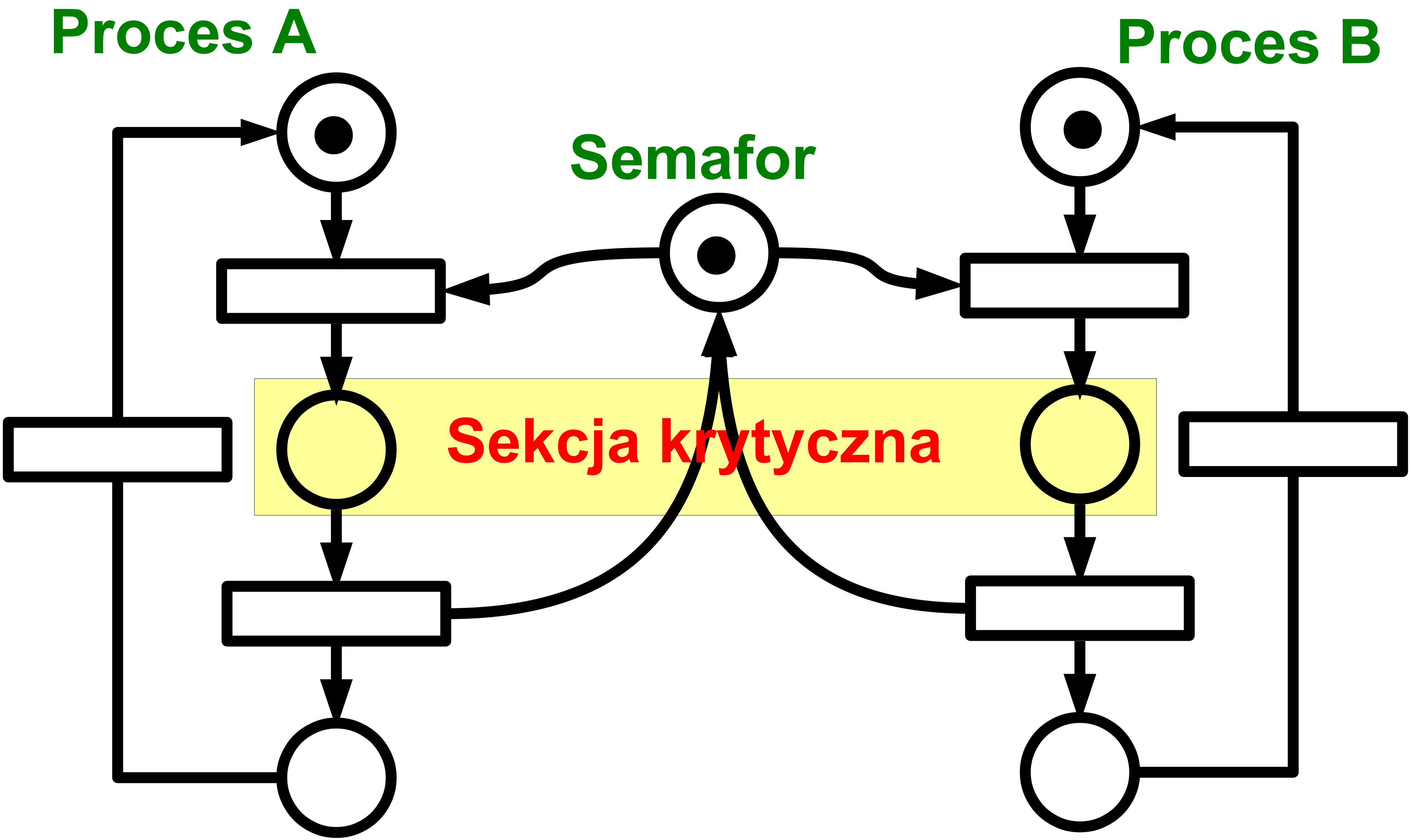


Odpalenie tranzycji powoduje zmianę znakowania miejsc



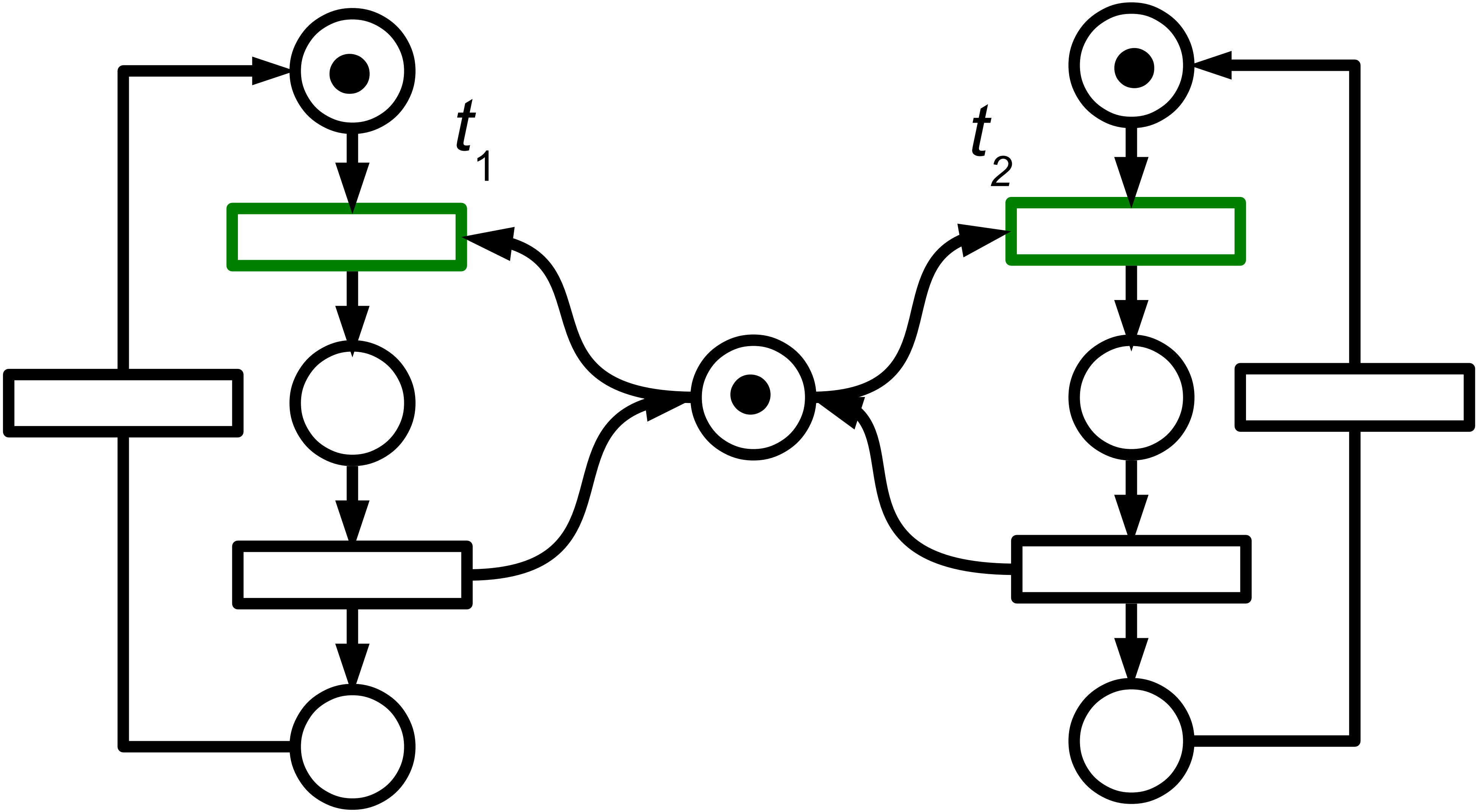


Sieci miejsc i tranzycji (PT-sieci)



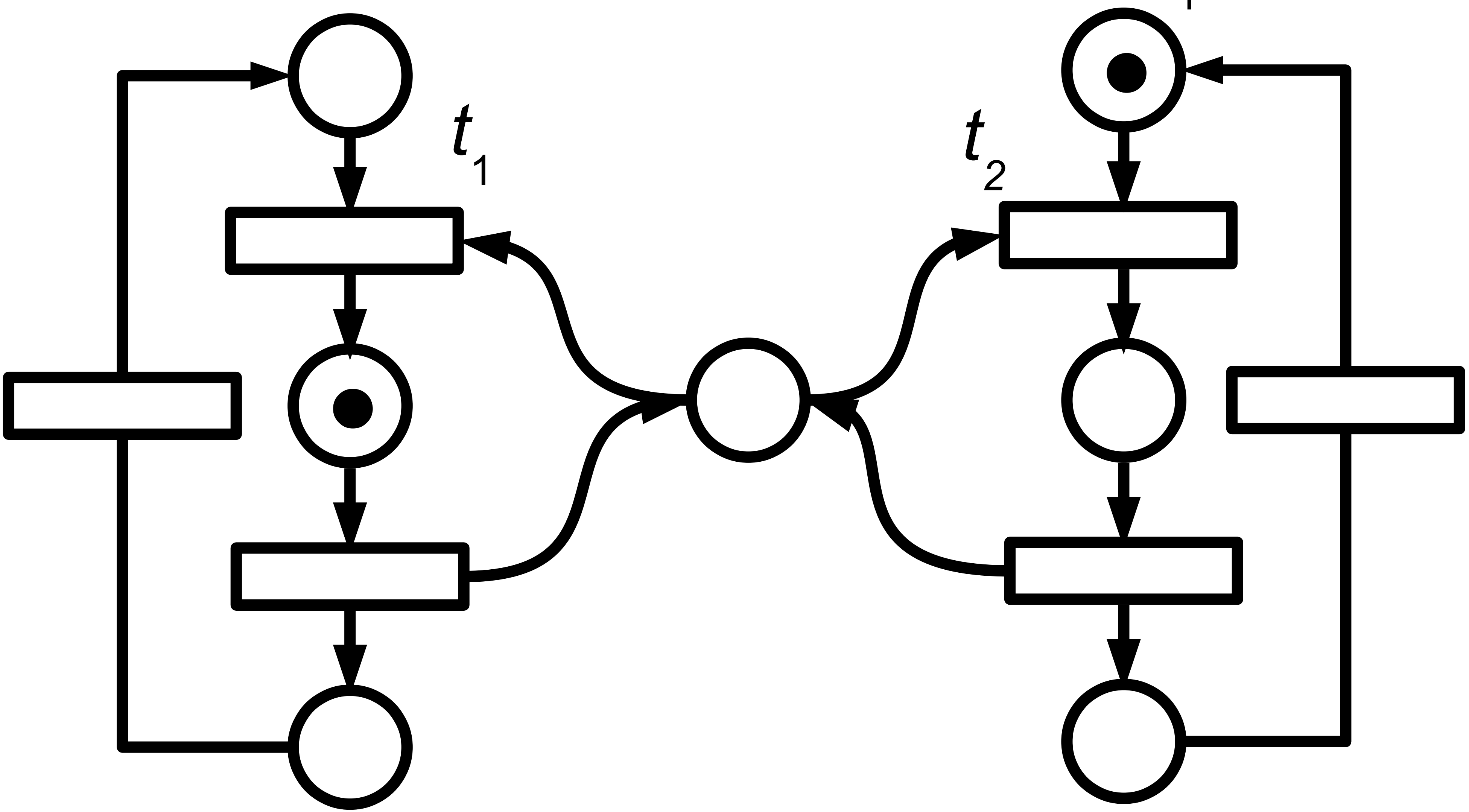
Sieci miejsc i tranzycji (PT-sieci)

Tranzycje gotowe do odpalenia:



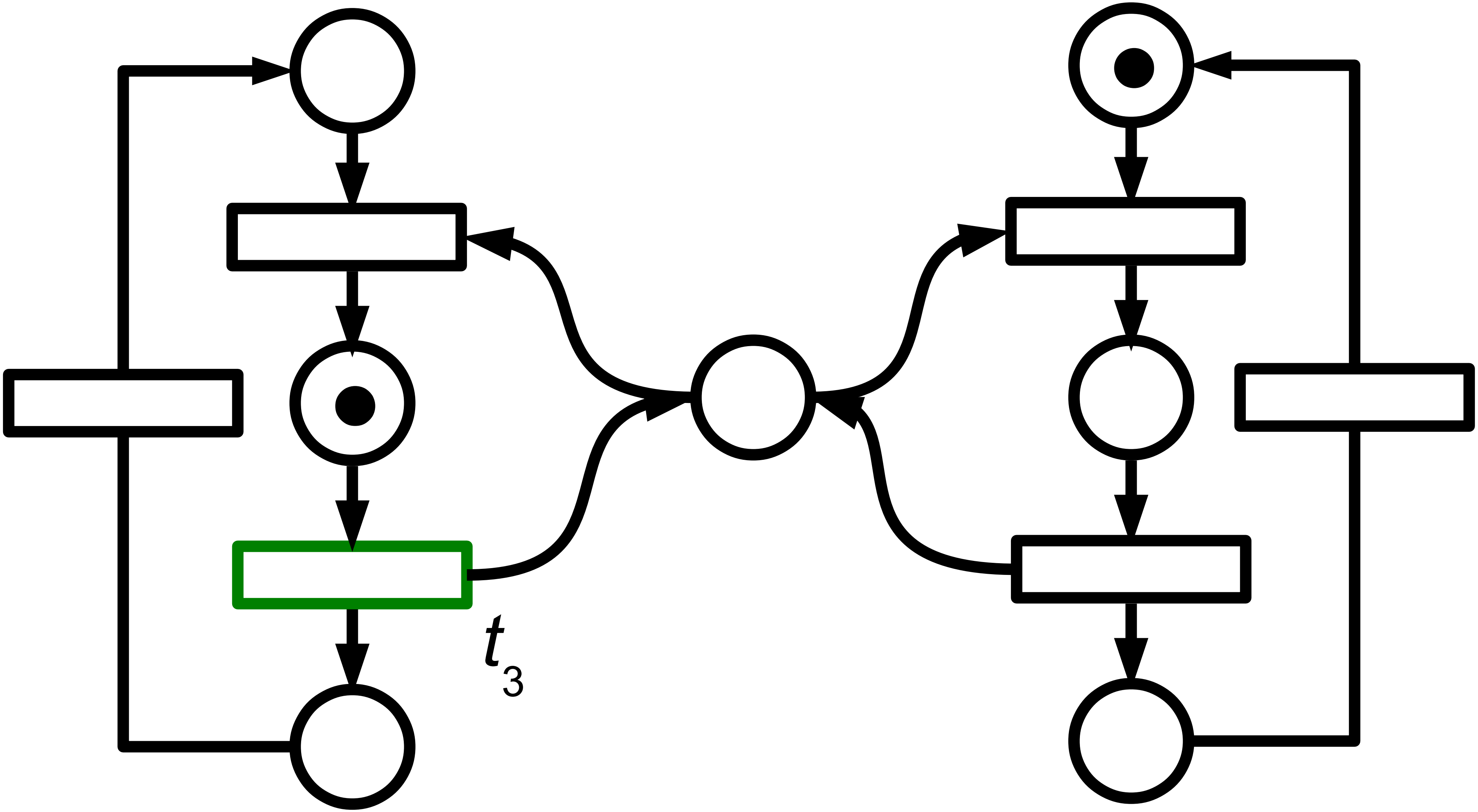
Sieci miejsc i tranzycji (PT-sieci)

Znakowanie po odpaleniu tranzycji t_1:



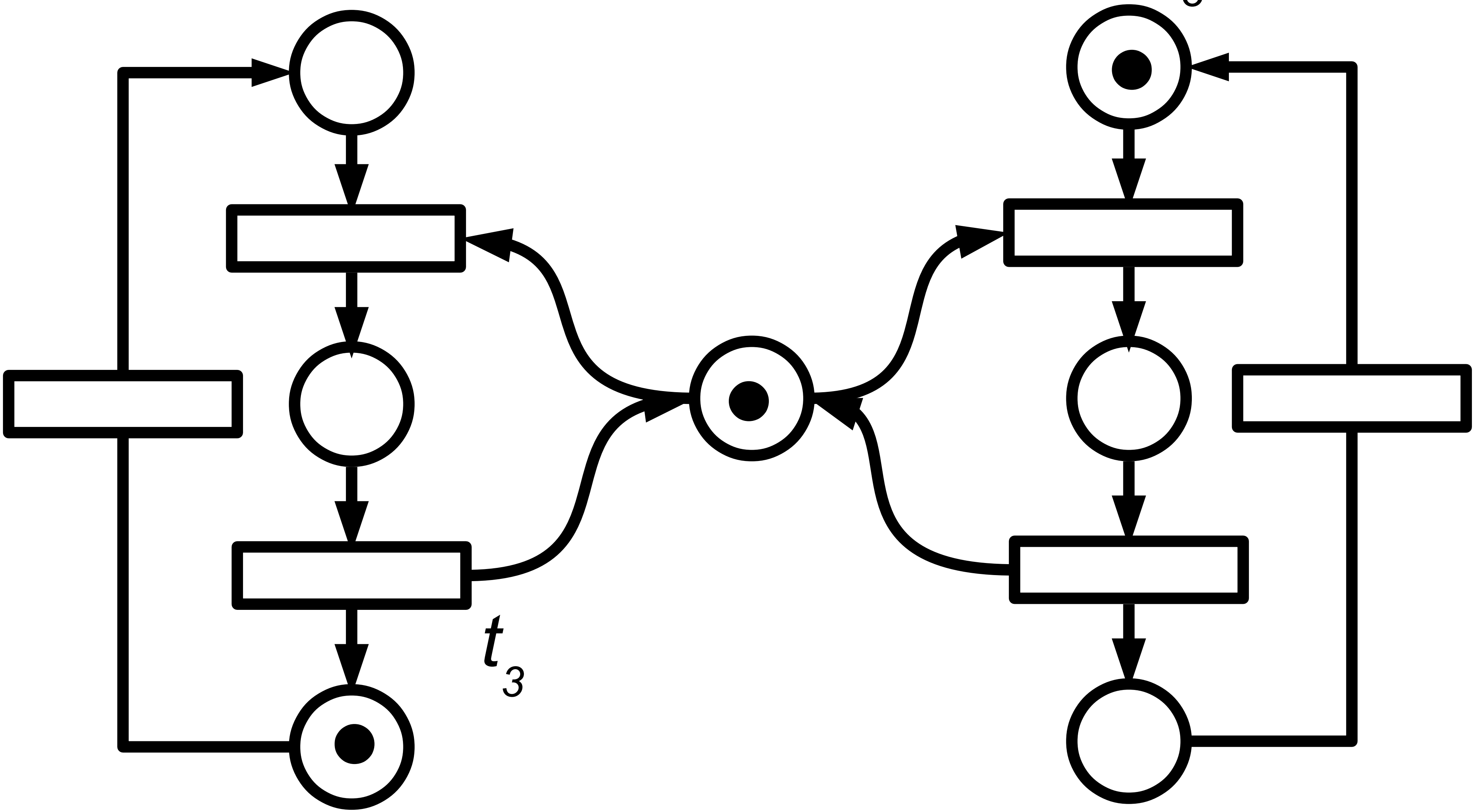
Sieci miejsc i tranzycji (PT-sieci)

Tranzycje gotowe do odpalenia:



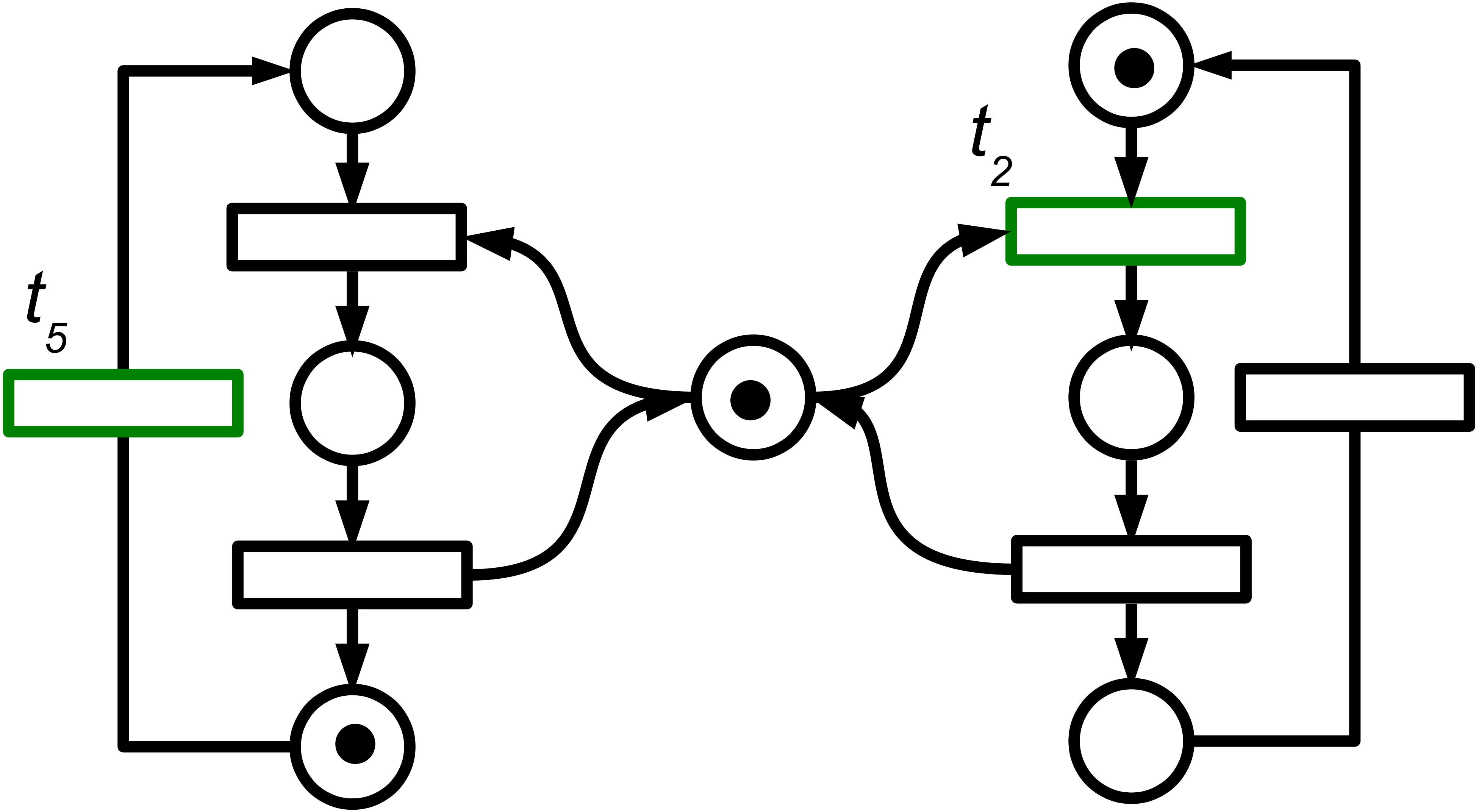
Sieci miejsc i tranzycji (PT-sieci)

Znakowanie po odpaleniu tranzycji t_3:



Sieci miejsc i tranzycji (PT-sieci)

Tranzycje gotowe do odpalenia:



Sieci miejsc i tranzycji (PT-sieci)

Znakowanie po odpaleniu tranzycji t_2: