Úvod do Petriho sítí

Petriho sítě

❖ Motivace:

- modely diskrétních systémů
- modely paralelních systémů
- modely distribuovaných systémů

❖ Využití:

$\boxed{\text{návrh} \times \text{syntéza} \times \text{analýza} \times \text{verifikace}}$

❖ Historie:

- C. A. Petri: Kommunikation mit automaten, 1962

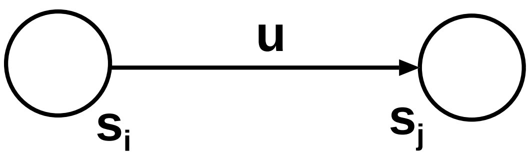
❖ Aplikace:

- hardware - paralelní architektury
- software - distribuované systémy, informační systémy, komunikační protokoly
- telekomunikace, strojírenství, administrativa

1. Základní koncepty Petriho sítí

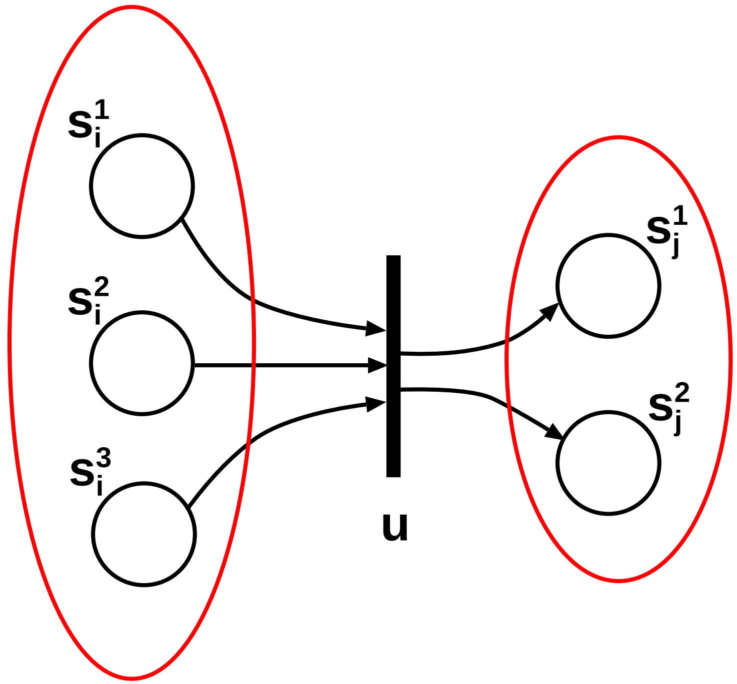
❖ Modelování událostí:

V konečném automatu:



$s_j \in \delta(s_i, u)$

V Petriho síti:



$\{s_i^1, s_i^2, s_i^3\} \approx s_i$

$\{s_j^1, s_j^2\} \approx s_j$

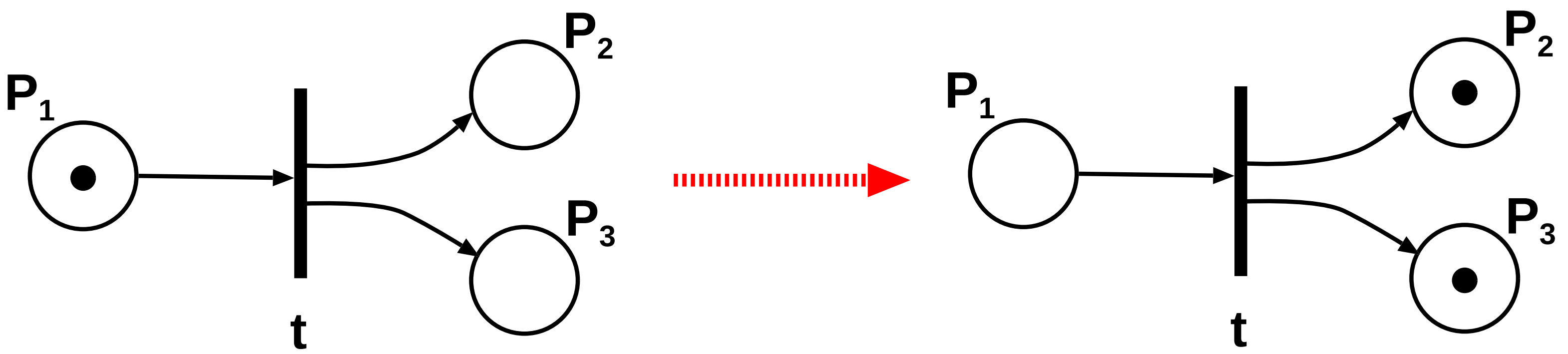
Složky Petriho sítě – *statická* reprezentace systému:

- **místa** (places)
- **přechody** (transitions)
- **hrany** (arcs)

Složky Petriho sítě – reprezentace *dynamiky (změn)* systému:

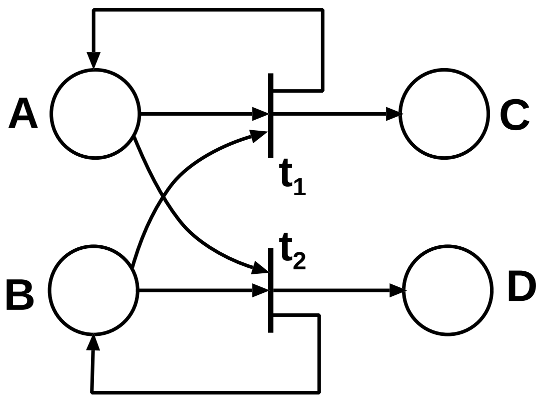
- **značky** (tokens)

Před provedením přechodu t: Po provedení přechodu t:



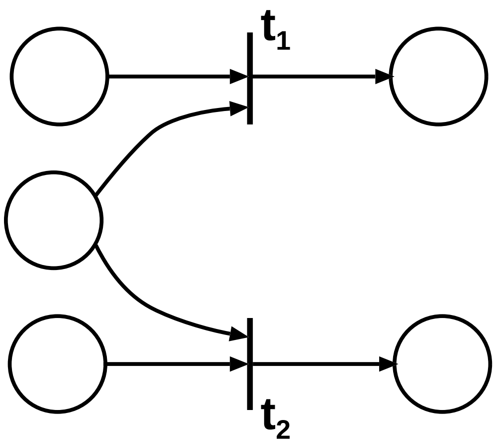
❖ Modelování podmíněnosti:

precondition: $A \wedge B$
postcondition: $(A \wedge \neg B \wedge C) \vee (\neg A \wedge B \wedge D)$



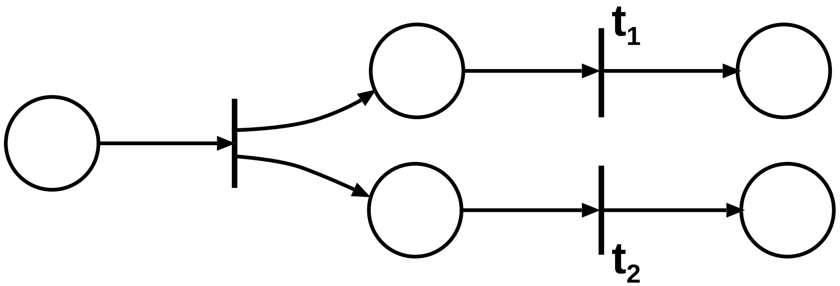
❖ Modelování vzájemné výlučnosti:

t_1 a t_2 jsou vzájemně vyloučeny
(konfliktní přechody)

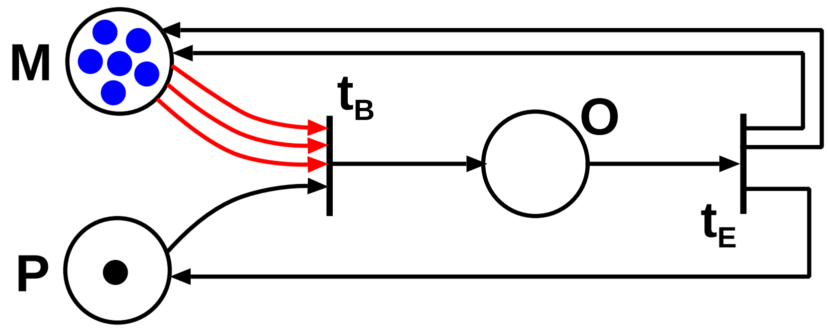


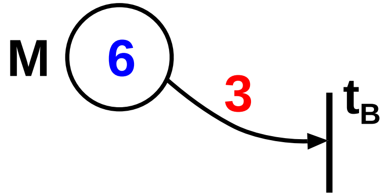
❖ Modelování paralelnosti (simultánnosti):

t_1 a t_2 jsou simultánní
(nezávislé přechody)



❖ Modelování požadavků na zdroje:

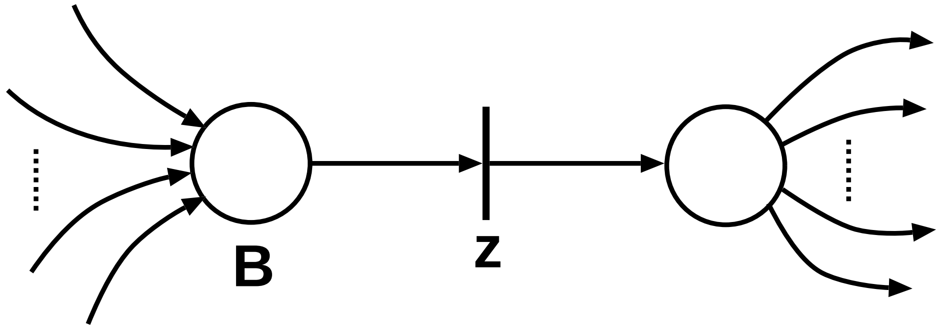




Interpretace míst a přechodů:

- M – počet volných paměťových bloků
- P – procesor je volný
- O – operace probíhá
- t_B – počátek operace
- t_E – konec operace

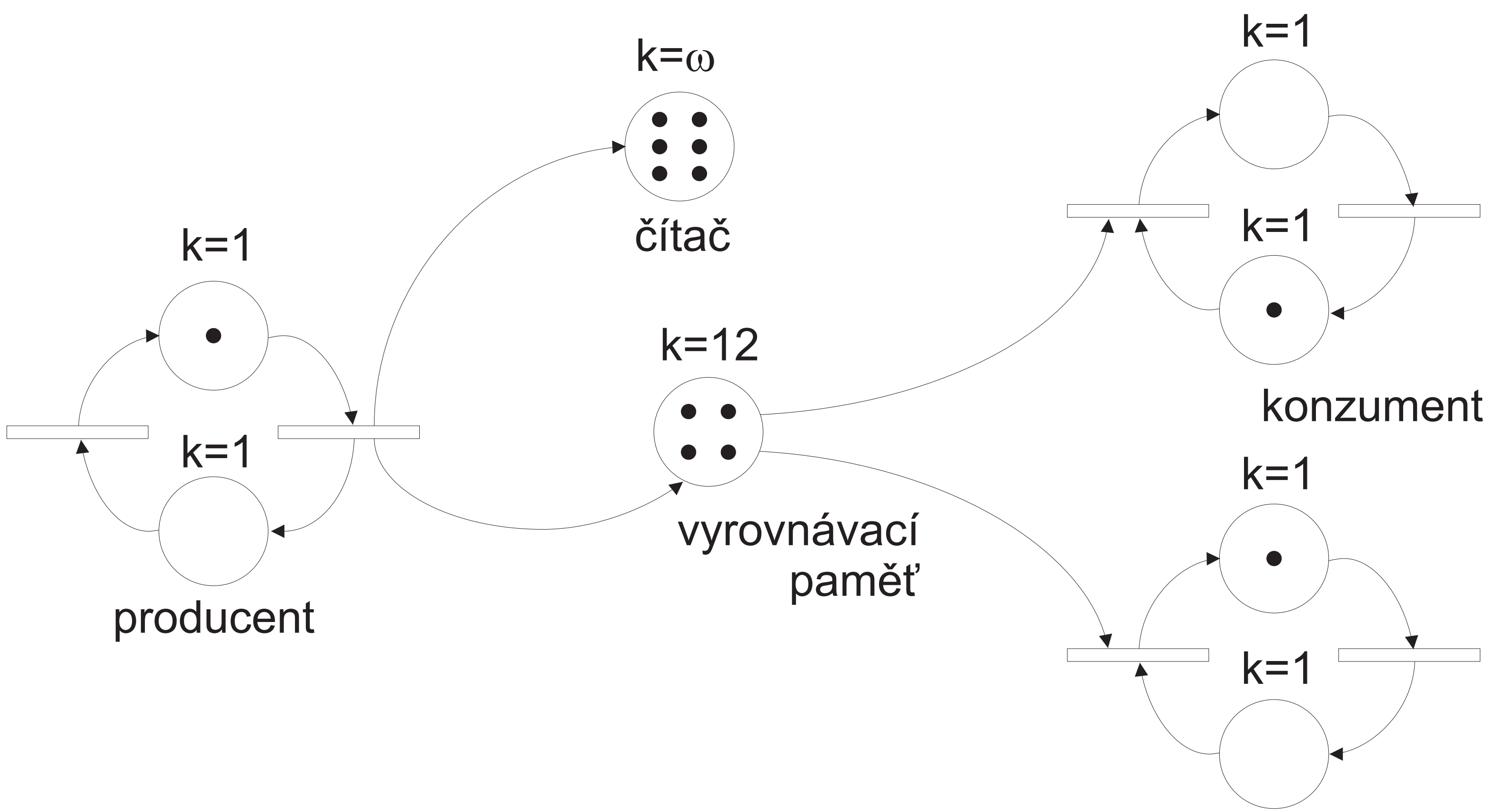
Poznámka: Problém vyrovnávacích pamětí (bufferů), front



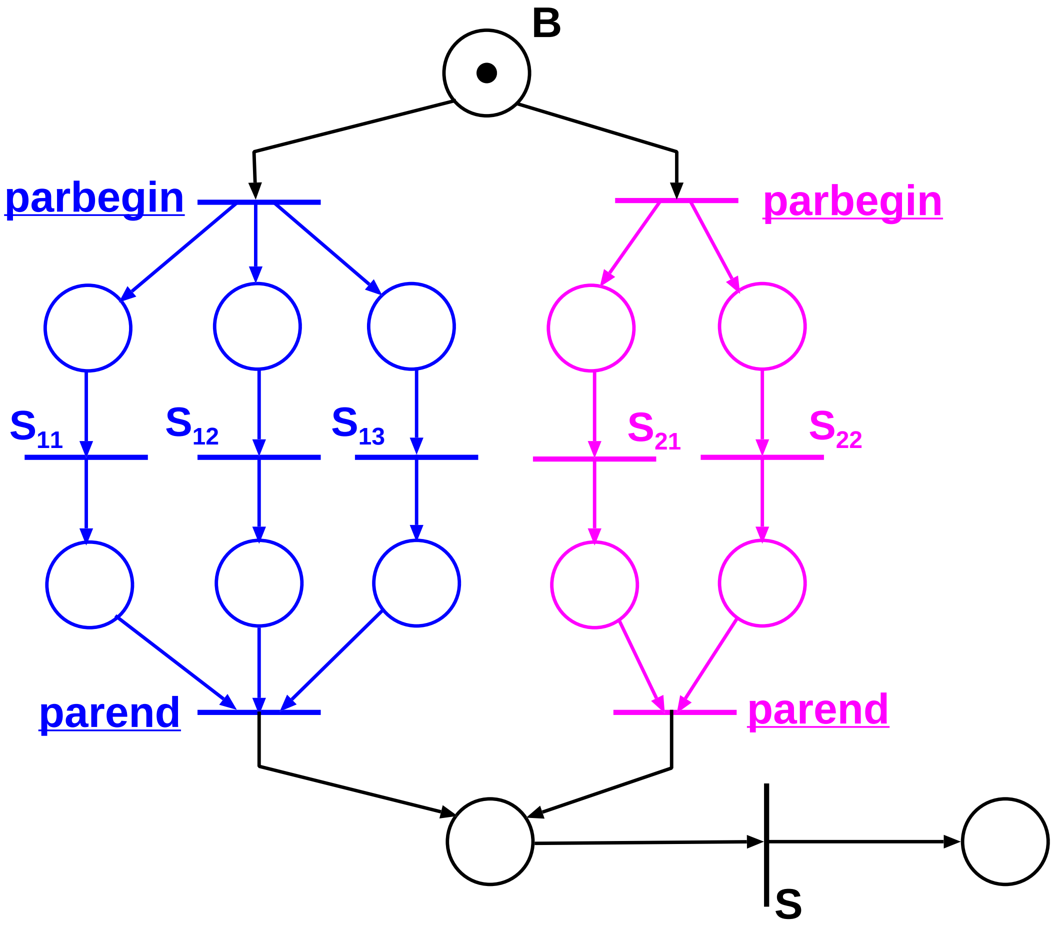
B: buffer, z zpracování položky

$\boxed{\text{Nemůže dojít k přetečení } B \text{ (bufferu, fronty)?}}$

❖ **Příklad 1**: producent-konzument



❖ **Příklad 2**: model úseku paralelního programu



<u>if</u> **B** <u>then</u>
  <u>parbegin</u>
    S_{11};
    S_{12};
    S_{13};
  <u>parend</u>
<u>else</u>
  <u>parbegin</u>
    S_{21};
    S_{22};
  <u>parend</u>;
S;

2. Základní matematické definice

- **Definice 1.** Trojici $N = (P, T, F)$ nazýváme sítí (net), jestliže:

1. P a T jsou disjunktní konečné množiny
2. $F \subseteq (P \times T) \cup (T \times P)$ je binární relace

P nazýváme množinou míst (places)

T nazýváme množinou přechodů (transitions)

F nazýváme tokovou relací (flow relation)

- Grafem sítě nazveme grafovou reprezentaci relace F.

- Graf sítě je bipartitní orientovaný graf s množinou uzlů $P \cup T$ vrcholů.

❖ **Definice 2.** Nechť $N = (P, T, F)$ je síť.

1. Pro všechny prvky $x \in (P \cup T)$
    - ${}^\bullet x = \{y \mid yFx\}$ se nazývá vstupní množinou (preset) prvku x
    - $x^\bullet = \{y \mid xFy\}$ se nazývá výstupní množinou (postset) prvku x

    Podobně pro množinu prvků: Nechť $X \subseteq (P \cup T)$, pak

    $${}^\bullet X = \bigcup_{x \in X} {}^\bullet x \quad \text{a} \quad X^\bullet = \bigcup_{x \in X} x^\bullet$$

    Zřejmě platí: $\boxed{\forall x, y \in (P \cup T) : x \in {}^\bullet y \ \Leftrightarrow \ y \in x^\bullet}$

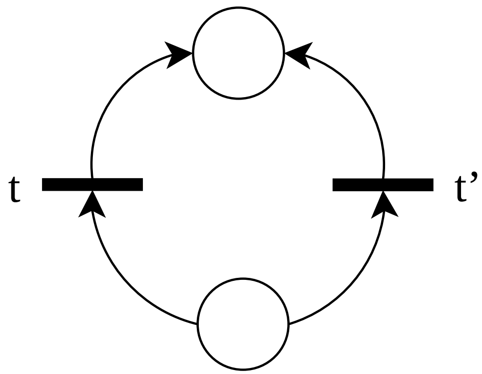
2. Uspořádaná dvojice $< p, t > \in P \times T$ se nazývá vlastní cyklus (self-loop), jestliže $pFt \wedge tFp$. Neobsahuje-li síť vlastní cyklus, pak se nazývá čistou sítí (pure net).

3. Prvek $x \in (P \cup T)$ se nazývá izolovaný, jestliže ${}^\bullet x \cup x^\bullet = \emptyset$.

❖ **Definice 3.** Nechť $N = (P, T, F)$ je síť. N se nazývá jednoduchou sítí (simple net), jestliže

$$\boxed{\forall x, y \in (P \cup T) \colon ({}^\bullet x = {}^\bullet y \ \wedge \ x^\bullet = y^\bullet) \Rightarrow x = y}$$

Příklad nejednoduché sítě:



❖ **Definice 4.** Nechť $N_1 = (P_1, T_1, F_1)$ a $N_2 = (P_2, T_2, F_2)$ jsou sítě. Existuje-li bijekce

$\beta : (P_1 \cup T_1) \leftrightarrow (P_2 \cup T_2)$ taková, že

1. $x \in P_1 \Leftrightarrow \beta(x) \in P_2$
2. $(x, y) \in F_1 \Leftrightarrow (\beta(x), \beta(y)) \in F_2$

pak N_1 a N_2 nazýváme izomorfní.

3. P/T Petriho sítě

❖ **Definice 5:** Šestici $N = (P, T, F, W, K, M_0)$ nazýváme *P/T Petriho sítí*

(Place/Transition Petri Net), jestliže:

1. (P, T, F) je konečná síť
2. $W : F \to \mathbb{N} \setminus \{0\}$ je ohodnocení hran grafu určující kladnou *váhu* každé hrany sítě
3. $K : P \to \mathbb{N} \cup \{\omega\}$ je zobrazení určující *kapacitu* každého místa
4. $M_0 : P \to \mathbb{N} \cup \{\omega\}$ je *počáteční značení* míst Petriho sítě takové, že $\forall p \in P : M_0(p) \leq K(p)$

Poznámka:

- $\mathbb{N}$ je množina $\mathbb{N} = \{0, 1, 2, ...\}$
- ω značí *supremum* množiny $\mathbb{N}$ s vlastnostmi:
  1. $\forall n \in \mathbb{N} : n < \omega$
  2. $\forall m \in \mathbb{N} \cup \{\omega\} : m + \omega = \omega + m = \omega - m = \omega$
- Petriho sítí budeme dále rozumět P/T Petriho síť

Definice 6: (*Evoluční pravidla Petriho sítí*)

Nechť $N = (P, T, F, W, K, M_0)$ je Petriho síť.

1. Zobrazení $M: P \to \mathbb{N} \cup \{\omega\}$ se nazývá *značení* (marking) Petriho sítě N, jestliže $\forall p \in P: M(p) \leq K(p)$
2. Nechť M je značení Petriho sítě N. Přechod $t \in T$ je *proveditelný* (enabled) *při značení* M (stručněji M*-proveditelný*), jestliže

$$\boxed{\begin{array}{l} \forall p \in {}^{\bullet}t: M(p) \geq W(p,t) \\ \forall p \in t^{\bullet}: M(p) \leq K(p) - W(t,p) \end{array}}$$

❖ Definice 6. (pokračování)

3. Je-li $t \in T$ M-proveditelný, pak jeho *provedením* získáme *následné značení* M' ke značení M, které je definováno takto:

$$\boxed{\forall p \in P\colon M'(p) = \begin{cases} M(p) - W(p,t) & \text{je-li } p \in {}^\bullet t \setminus t^\bullet \\ M(p) + W(t,p) & \text{je-li } p \in t^\bullet \setminus {}^\bullet t \\ M(p) - W(p,t) + W(t,p) & \text{je-li } p \in {}^\bullet t \cap t^\bullet \\ M(p) & \text{jinak} \end{cases}}$$

Provedení přechodu t (transition firing) ze značení M do značení M' zapisujeme symbolicky:

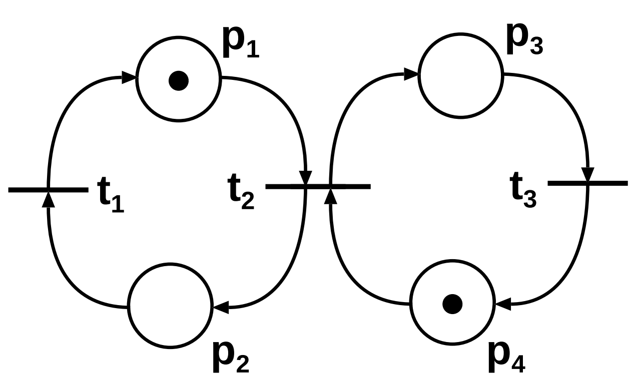
$$\boxed{M[t\rangle M'}$$

❖ Definice 6. (pokračování)

4. Označme $[M\rangle$ nejmenší množinu různých značení Petriho sítě N, pro kterou platí:
   (a) $M \in [M\rangle$
   (b) Je-li $M_1 \in [M\rangle$ a pro nějaké $t \in T$ platí $M_1[t\rangle M_2$, pak $M_2 \in [M\rangle$.

   Množina $[M\rangle$ se nazývá *množinou dosažitelných značení* (reachability set) *ze značení* M.
   Množina $[M_0\rangle$ se nazývá *množinou dosažitelných značení sítě* N.

❖ **Příklad 3**: Uvažujme následující Petriho síť:



$[M_0\rangle = \{M_0, M_1, M_2, M_3\}$, kde

$$M_0 = (1, 0, 0, 1)$$
$$M_1 = (0, 1, 1, 0)$$
$$M_2 = (1, 0, 1, 0)$$
$$M_3 = (0, 1, 0, 1)$$

4. Stavový prostor a přechodová funkce Petriho sítě

❖ Množina $[M_0\rangle$ reprezentuje *stavový prostor Petriho sítě*. Mohou nastat dva případy:

$$[M_0\rangle \quad \begin{cases} \text{je konečná množina} \\ \text{je spočetná nekonečná množina} \end{cases}$$

❖ **Definice 7.** Nechť $N = (P, T, F, W, K, M_0)$ je Petriho síť a $[M_0\rangle$ její množina dosažitelných značení. *Přechodovou funkcí Petriho sítě* N nazveme funkci δ:

$$\boxed{\begin{array}{l} \delta \colon [M_0\rangle \times T \to [M_0\rangle, \text{ pro kterou} \\ \forall t \in T\colon \ \ \forall M, M' \in [M_0\rangle\colon \ \ \delta(M, t) = M' \overset{def.}{\Longleftrightarrow} M[t\rangle M' \end{array}}$$

❖ Přechodová funkce δ může být zobecněna na posloupnost přechodů:

$$\delta : [M_0\rangle \times T^* \rightarrow [M_0\rangle$$

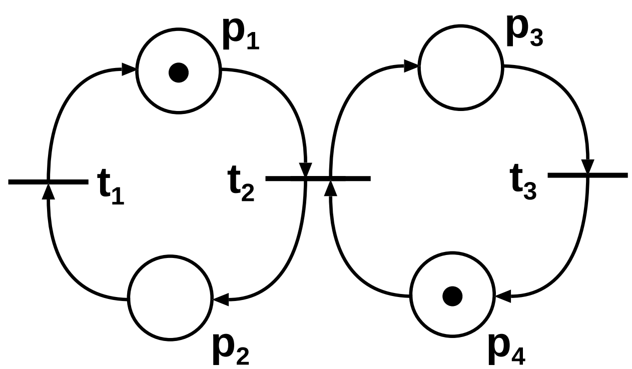
takto:

$$\boxed{\begin{array}{l} \delta(M, t\tau) = \delta(\delta(M, t), \tau),\ \tau \in T^* \\ \delta(M, \varepsilon) = M, \text{ kde } \varepsilon \text{ je prázdný symbol} \end{array}}$$

❖ Řetězec $\tau \in T^+$ nazveme *výpočetní posloupností* Petriho sítě, je-li $\delta(M_0, \tau)$ definována (+ případné další podmínky).

❖ *Jazyk Petriho sítě* = množina výpočetních posloupností Petriho sítě.

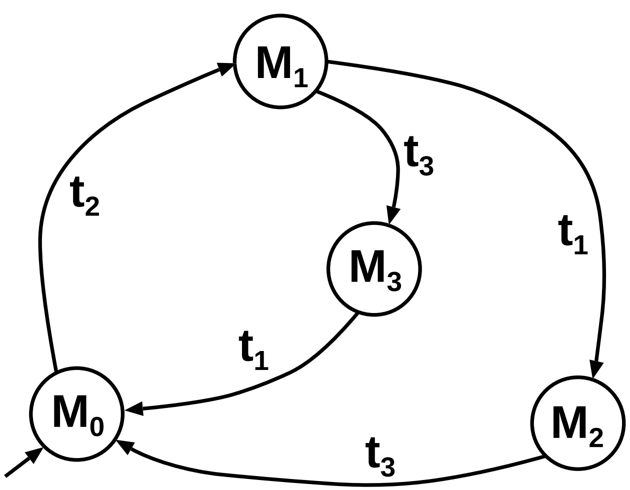
❖ **Příklad 4**: Uvažme Petriho síť z příkladu 1 a její množinu dosažitelných značení:



$[M_0\rangle = \{M_0, M_1, M_2, M_3\}$, kde

$$M_0 = (1,0,0,1)$$
$$M_1 = (0,1,1,0)$$
$$M_2 = (1,0,1,0)$$
$$M_3 = (0,1,0,1)$$

Odpovídající přechodová funkce specifikovaná grafem vypadá takto:



Množina výpočetních posloupností dané Petriho sítě pak může být charakterizována regulárním výrazem:

$$(t_2(t_3t_1 + t_1t_3))^*$$

Každý neprázdný prefix řetězce specifikovaného tímto výrazem tvoří výpočetní posloupnost.

5. Analýza P/T Petriho sítí

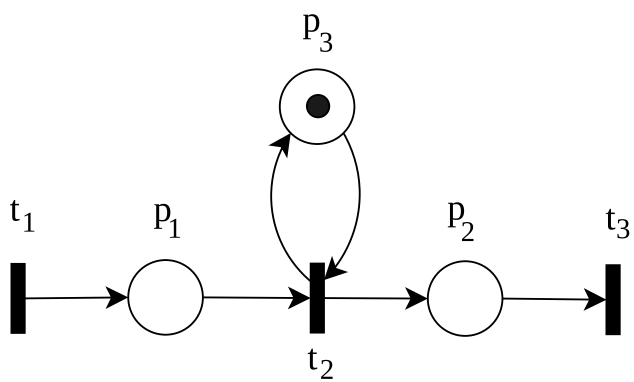
❖ Základní problémy analýzy

- bezpečnost (safeness)
- omezenost (boundness)
- konzervativnost (conservation)
- živost (liveness)

❖ **Definice 8**: Místo $p \in P$ Petriho sítě $N = (P, T, F, W, K, M_0)$ s počátečním značení

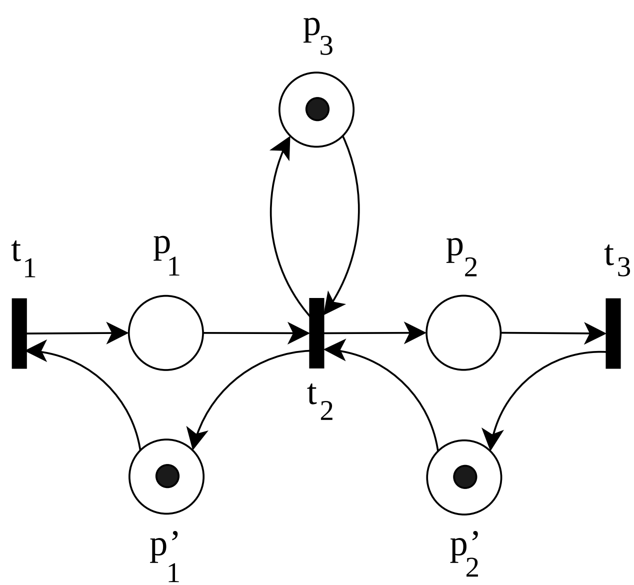
M_0 je *bezpečné* (safe), jestliže pro všechna značení $M \in [M_0\rangle$ je $M(p) \leq 1$. Petriho síť je *bezpečná*, je-li každé její místo bezpečné.

❖ **Příklad 5**:



síť, která není bezpečná

Neobsahuje-li graf Petriho sítě násobné hrany, může být transformován na bezpečnou síť následujícím postupem.



odpovídající bezpečná síť

Postup:

1. K místu p, které má bý bezpečné přidej komplementární místo p'.
2. Modifikuj incidující přechody podle algoritmu komplementace sítě.

❖ **Definice 9**: Místo $p \in P$ Petriho sítě $N = (P, T, F, W, K, M_0)$ se nazývá *k-bezpečné*,

jestliže pro všechna značení $M \in [M_0\rangle$ je $M(p) \leq k$. Je-li místo p' k-bezpečné pro nějaké k, nazývá se *omezené* (bounded). Petriho síť, jejíž všechna místa jsou omezená se nazývá *omezená Petriho síť*.

Omezenost sítě $\Rightarrow$ konečný stavový prostor sítě $\Rightarrow$ ekvivalenci sítě s konečnými automaty

❖ **Definice 10**: Petriho síť $N = (P, T, F, W, K, M_0)$ je *striktně konzervativní*, jestliže platí:

$$\boxed{\forall M \in [M_0\rangle : \sum_{p \in P} M(p) = \sum_{p \in P} M_0(p)}$$

Konzervativnost vzhledem k váhovému vektoru $\underline{w} = (w_1, \ldots, w_n), w_i \geq 0$

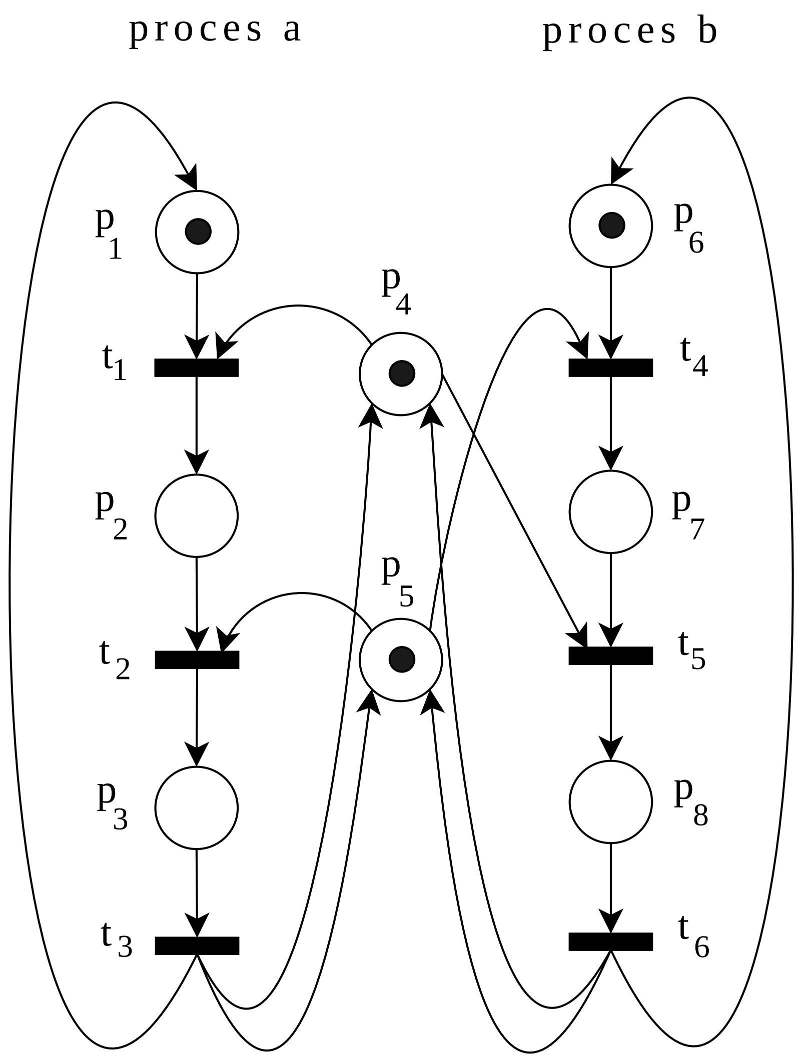
$$\boxed{\forall M \in [M_0\rangle : \sum_{i=1}^{n} w_i.M(p_i) = \sum_{i=1}^{n} w_i.M_0(p_i)}$$

❖ **Definice 11**: Nechť $N = (P, T, F, W, K, M_0)$ je Petriho síť a $t \in T$.

1. t se nazývá *živý přechod*, jestliže pro každé značení $M \in [M_0\rangle$ existuje značení $M' \in [M\rangle$ takové, že t je proveditelný při značení M'.
2. Síť N se nazývá *živou*, je-li každý její přechod živý.

Aplikace: živost x deadlock

❖ **Příklad 6**:



Proveditelné posloupnosti přechodů:

$t_1 t_2 t_3 t_4 t_5 t_6 \ldots$

$t_4 t_5 t_6 t_1 t_2 t_3 \ldots$

Uvažujme však posloupnost přechodů, která začíná $t_1 t_4 \ldots$

- **Definice 12**: Značení M Petriho sítě $N = (P, T, F, W, K, M_0)$ je *živé*, jestliže
  - pro všechna $t \in T$ existuje $M' \in [M\rangle$ takové, že přechod t je proveditelný při značení M'.
- **Věta 1**: Petriho síť je *živá*, právě když všechna značení z $[M_0\rangle$ jsou živá.

- **Definice 13**: (Problém dosažitelnosti - Reachability problem)
  - Je dána Petriho síť N s počátečním značením M_0 a značení M. Je $M \in [M_0\rangle$?
- **Definice 14**: (Problém pokrytí - Coverability problem)
  - Je dána Petriho síť N s počátečním značením M_0 a značení M. Existuje $M' \in [M_0\rangle$ takové, že $M' \geq M$?
- **Další problémy analýzy**:
  - posloupnosti přechodů (firing sequences)
  - ekvivalence sítí
  - inkluse sítí

Techniky analýzy Petriho sítí:

❖ **Strom dosažitelných značení (The Reachability Tree)**:

Strom dosažitelných značení je konečnou reprezentací množiny dosažitelných značení $[M_0\rangle$. Strom dosažitelných značení je kořenový orientovaný strom, jehož kořenem je počáteční značení M_0 a vrcholy tvoří vektory z $(\mathbb{N} \cup \{\omega\})^n, n = |P|$. Kde ω značí supremum množiny $\mathbb{N}$ s vlastnostmi:

1. $\forall n \in \mathbb{N} \colon n < \omega$
2. $\forall m \in \mathbb{N} \cup \{\omega\} \colon m + \omega = \omega + m = \omega - m = \omega$

❖ **Invarianty P/T Petriho sítí**:

6. Barvené Petriho sítě

- Kurt Jensen, Aarhus Uviversity, Dánsko, 1981.
- Monografie: K. Jensen: *Coloured Petri Nets*. Monographs in Theoretical Computer Science, Springer-Verlag, 1992-1997. Tří díly: základní koncepty, analýza a průmyslové případové studie.
- Řada úvodních článků, příkladů, ... dostupná na `http://www.daimi.au.dk/CPnets/`.
- Existují i alternativní koncepty CPN, všechny ale více méně v podobném duchu. Někdy se též hovoří o tzv. High-Level Petri Nets.

❖ CPN jsou motivovány snahou odstranit některé nevýhody klasických (P/T) Petriho sítí:

- Petriho sítě, poskytující primitiva pro popis synchronizace paralelních procesů, jsou rozšířeny o explicitní popis datových typů a datových manipulací.

❖ Nástroje: Design/CPN, CPN Tools (oba Aarhus University), dále např. ExSpect, ...
(viz `http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/db.html`).

❖ CPN byly aplikovány v řadě průmyslových případových studií:

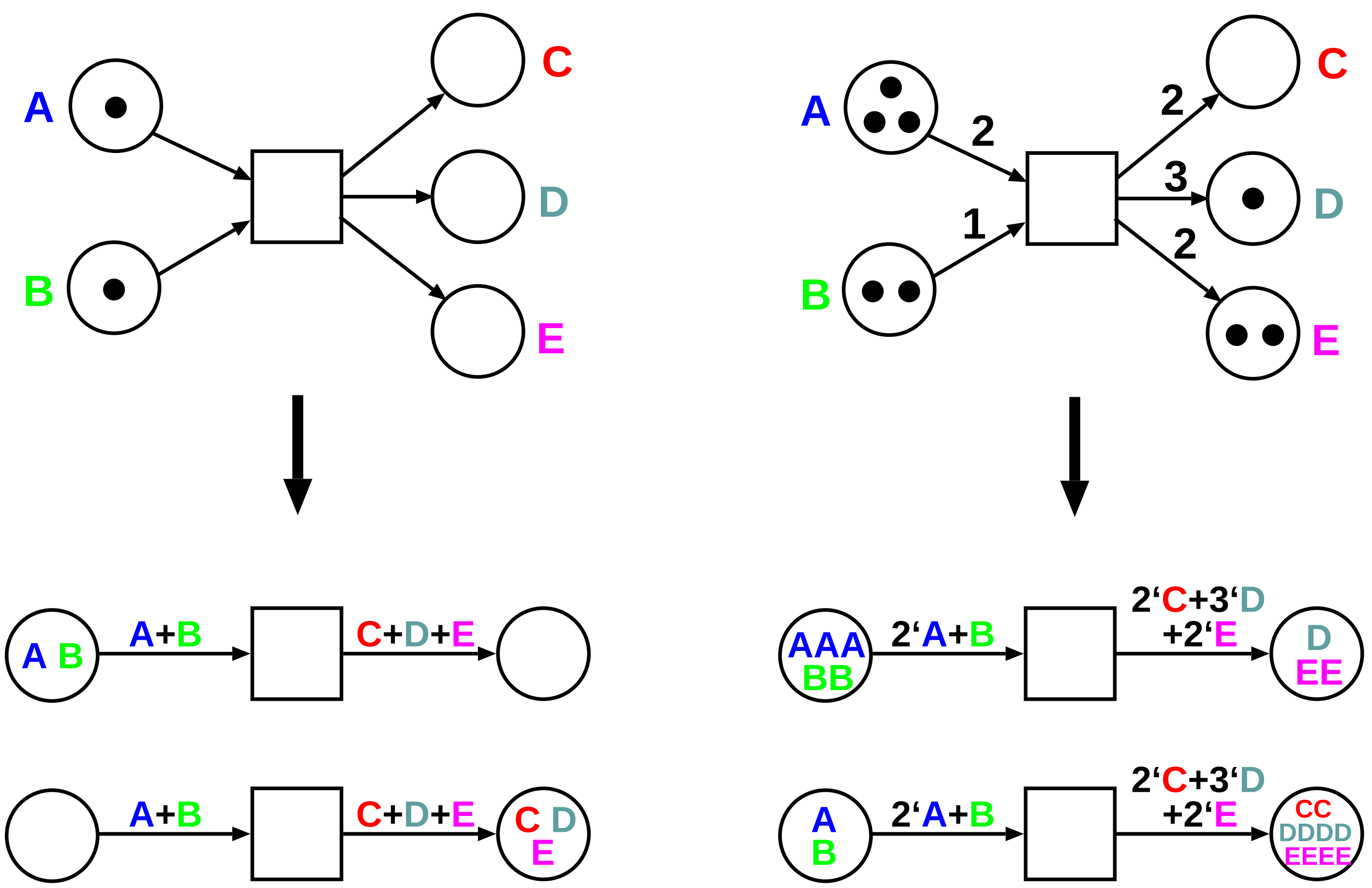
- komunikační protokoly a sítě,
- software (části SW Nokia, bankovní transakce, distribuované algoritmy, ...),
- hardware,
- řídící systémy,
- vojenské systémy,
- ...

❖ Podobně jako u P/T Petriho sítí existují různá rozšíření CPN o fyzický čas.

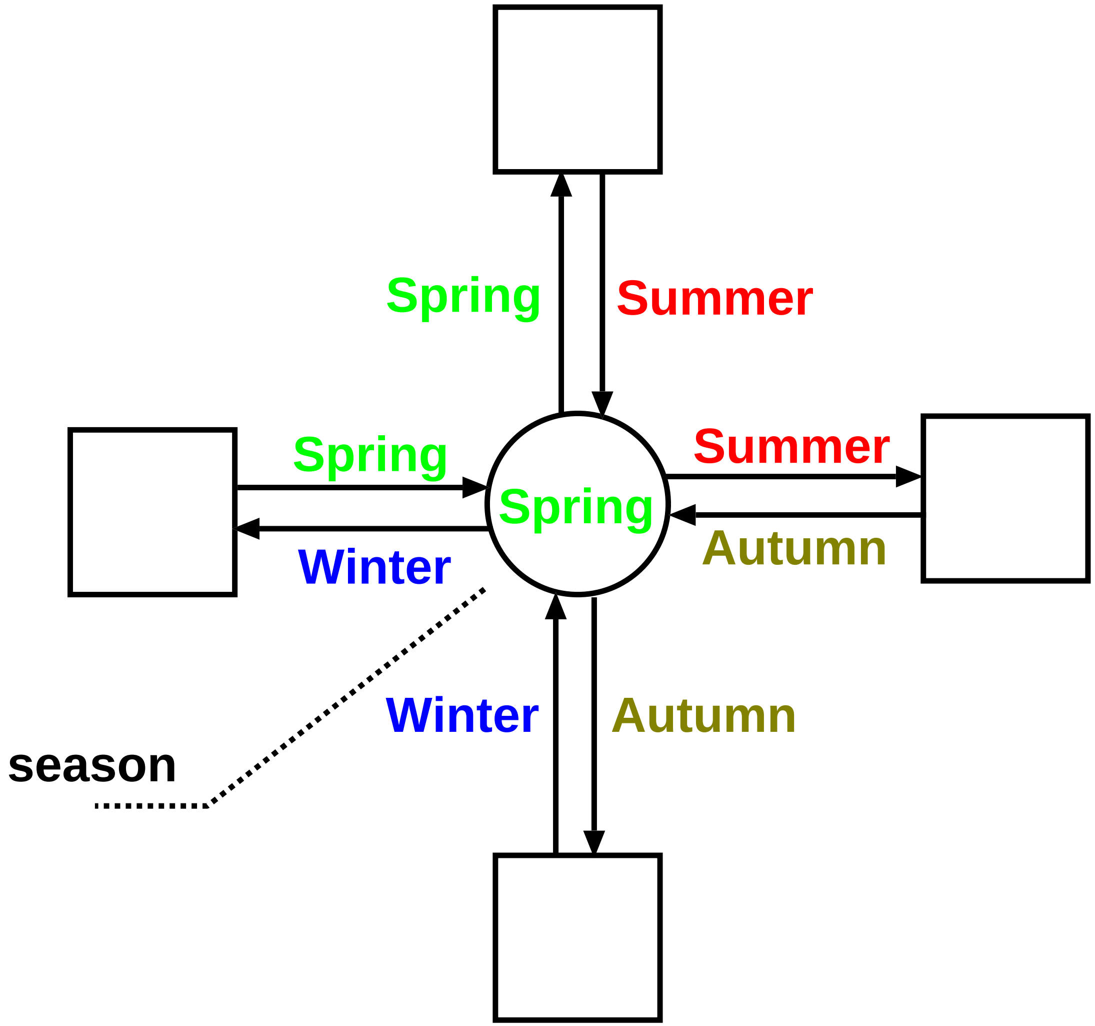
❖ CPN jsou základem pro další rozšíření: hierarchické CPN či různé objektově-orientované Petriho sítě (PNtalk, Renew, ...).

Petriho sítě s individuálními značkami

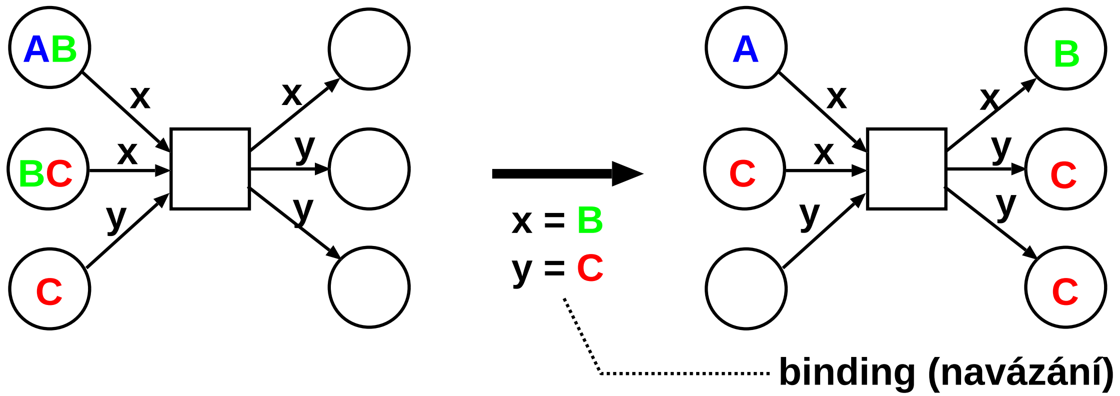
❖ Individual Token Nets with Constant Arrow Labels:

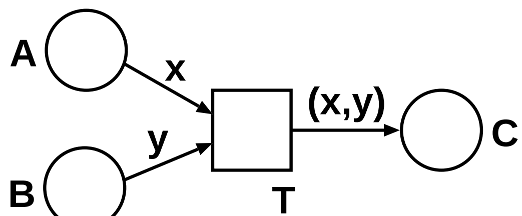


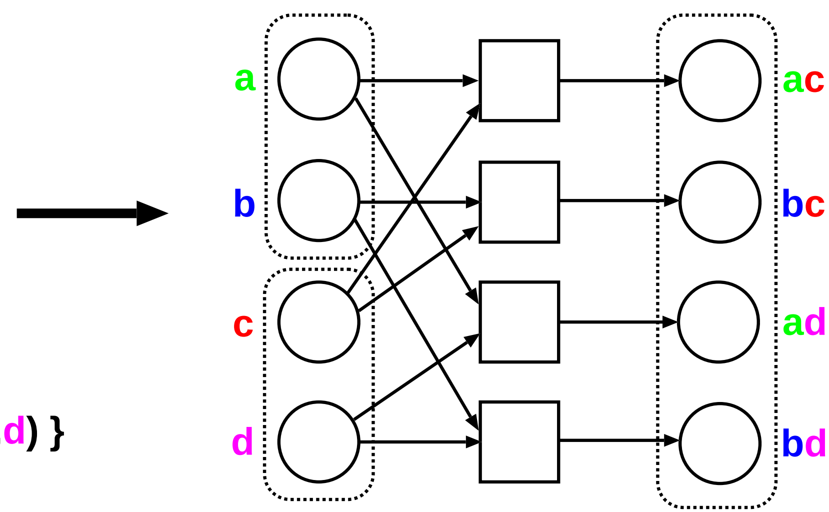
❖ Další jednoduchý příklad – změna ročních období:



❖ Individual Token Nets with Variable Arrow Labels:







TYPEOF(A) = { a, b }
TYPEOF(B) = { c, d }
TYPEOF(C) = { a, b } x {c, d }
= { (a,c), (a,d), (b,c), (b,d) }

colour sets

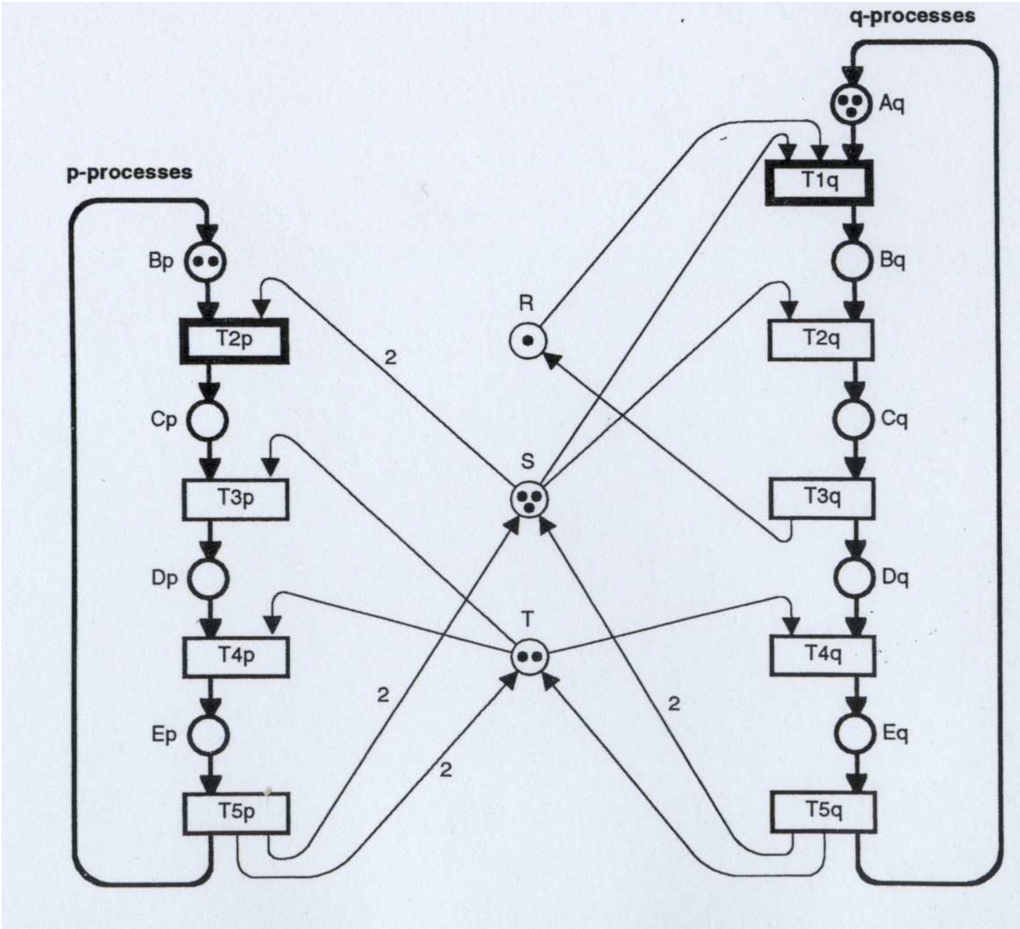
colours

Neformální zavedení CPN

❖ Uvažujme příklad popisu systému přidělování prostředků (zdrojů). Systém je tvořen:

- 2 třídami procesů – procesy `p`, resp. `q`,
- 3 typy zdrojů – `R`, `S`, `T`,
- stavy procesů – `Bp`, `Cp`, ..., `Ep`, `Aq`, `Bq`, ..., `Eq`,
- počátečním stavem.

Vlastní činnost systému lze popsat P/T Petriho sítí takto:



❖ V CPN můžeme “sloučit” popis chování podobných procesů $\mathtt{p}$ a $\mathtt{q}$. Budeme registrovat, který průchod “alokačním cyklem” daný proces provádí.

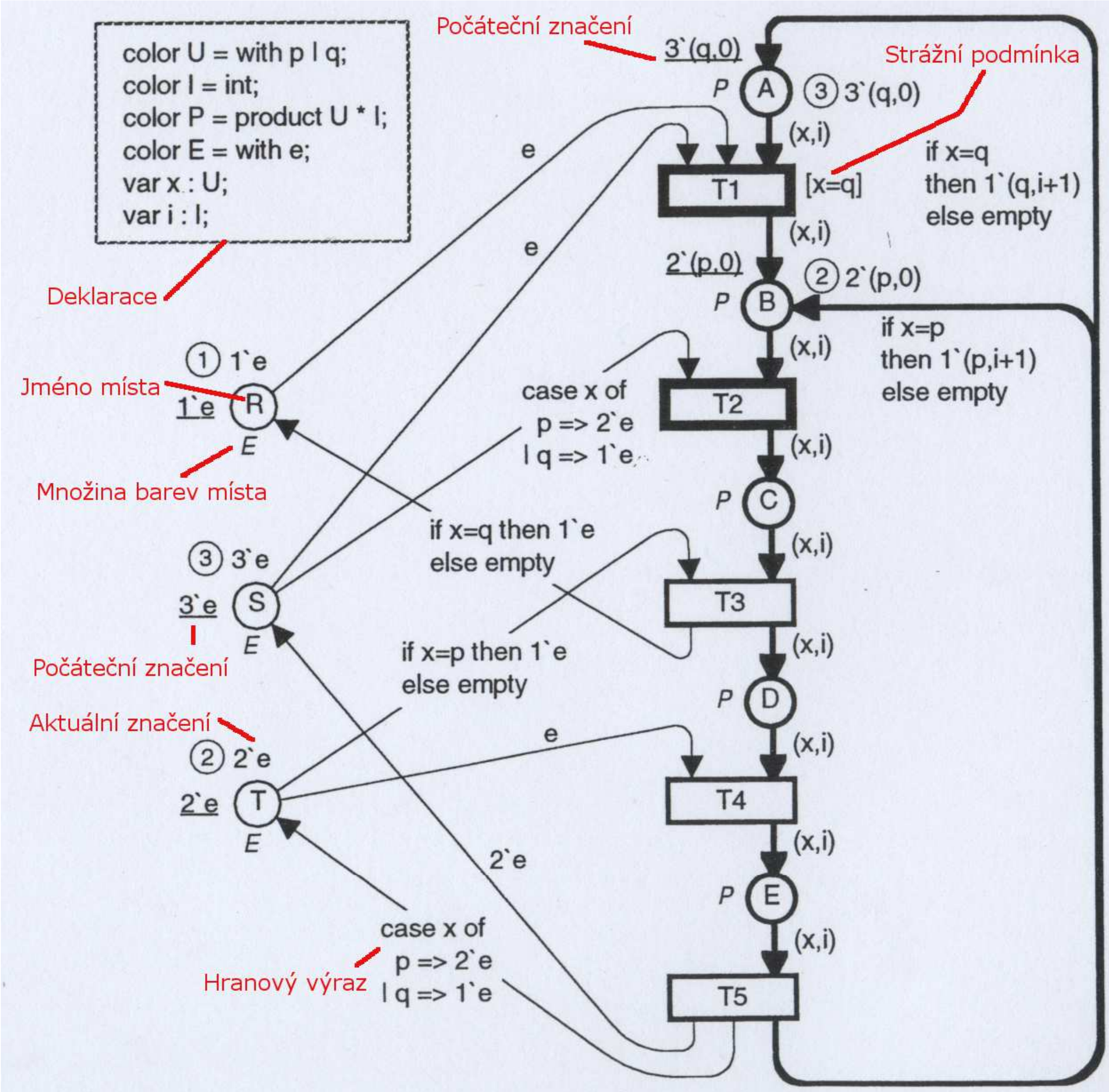
❖ Model ve tvaru CPN zahrnuje dvě složky:

1. grafickou část – graf Petriho sítě a
2. popisy – inskripci.

❖ Inskripce, vyjádřená inskripčním jazykem, obsahuje:

- deklaraci množin barev (coloured sets), tj. datových typů,
- specifikaci množin barev míst,
- popis hran,
- strážní podmínky přechodů,
- počáteční značení,
- (jména míst a přechodů).

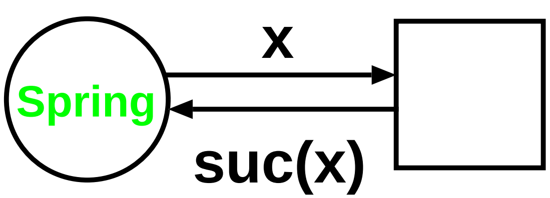
❖ Náš systém sdílení zdrojů pak můžeme modelovat např. tak, jak je ukázáno na následujícím slajdu...



color U = with p | q;
color I = int;
color P = product U * I;
color E = with e;
var x : U;
var i : I;
Deklarace
Počáteční značení
3`(q,0)
Strážní podmínka
A
3 3`(q,0)
(x,i)
T1
[x=q]
if x=q
then 1`(q,i+1)
else empty
(x,i)
2`(p,0)
2 2`(p,0)
B
if x=p
then 1`(p,i+1)
else empty
T2
case x of
p => 2`e
| q => 1`e
C
T3
D
T4
T5
E
P
e
1 1`e
Jméno místa
1`e
R
Množina barev místa
if x=q then 1`e
else empty
3 3`e
3`e
S
Počáteční značení
if x=p then 1`e
else empty
Aktuální značení
2 2`e
2`e
T
2`e
case x of
p => 2`e
| q => 1`e
Hranový výraz

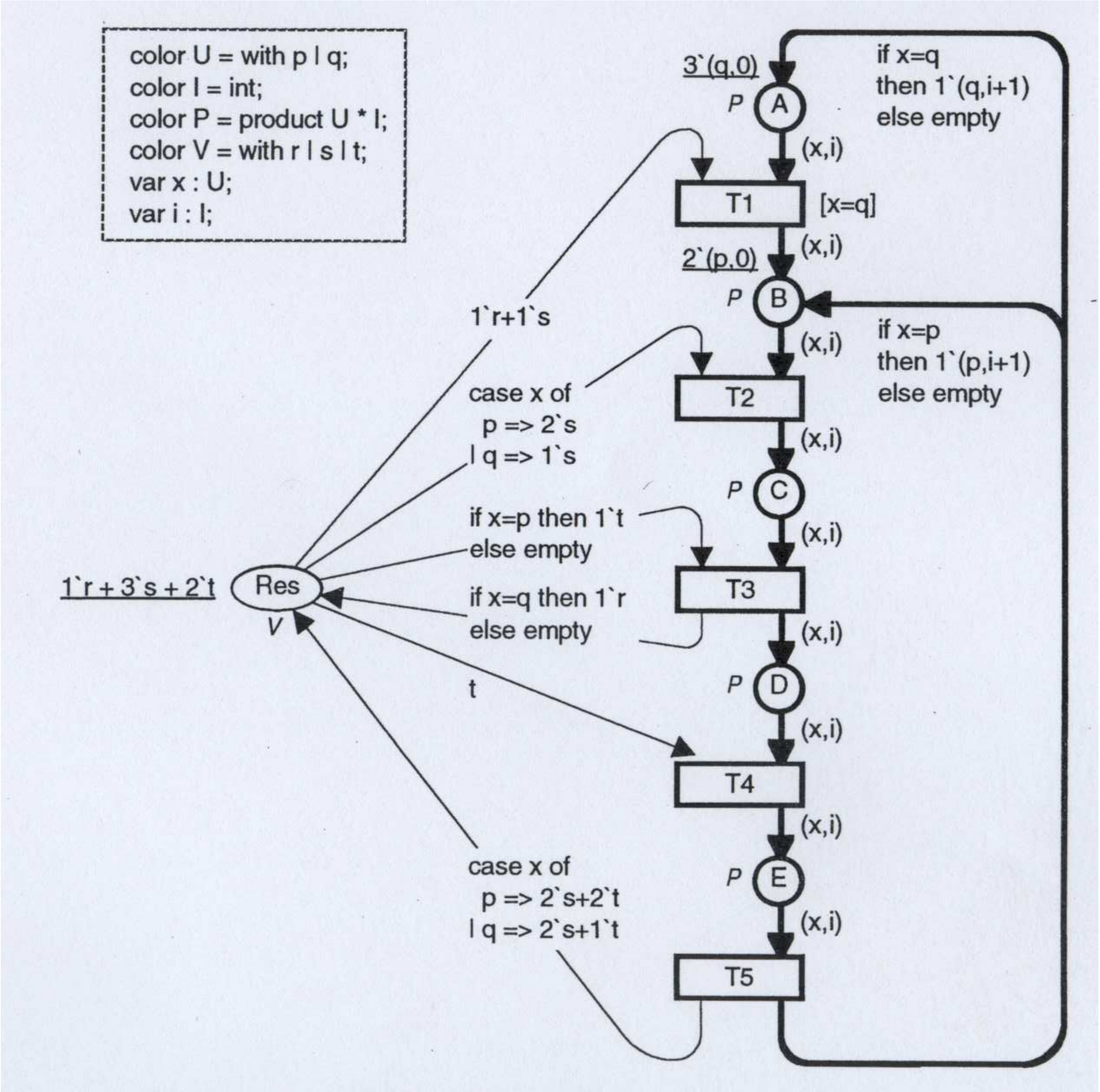
❖ Každý hranový výraz se vyhodnotí na multimnožinu značek:

- konstruktor multimnožiny: $n_1`c_1 + n_2`c_2 + ...n_m`c_m$,
- $n_1, n_2, ..., n_m$ jsou konstanty, proměnné nebo funkce, které se vyhodnotí na kladná přirozená čísla,
- $c_1, c_2, ..., c_m$ jsou konstanty, proměnné nebo funkce, které se vyhodnotí na barvy,
- příklady:
  - `if x=C then 3‘D else 4‘E+5‘F`
  - `2‘(x+y)+3‘1`
  - varianta jednoduchého popisu změn ročních období:



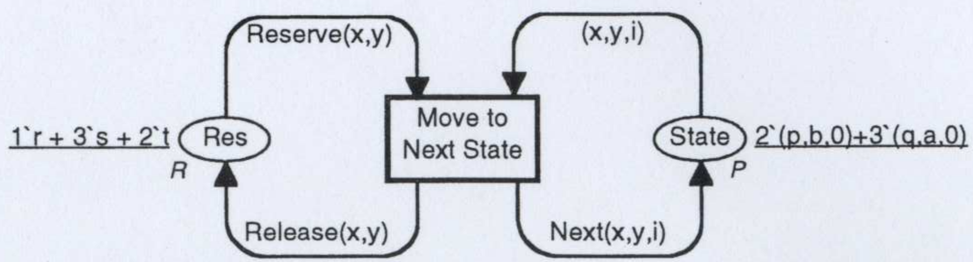
suc(Spring)=Summer
suc(Summer)=Autumn
suc(Autumn)=Winter
suc(Winter)=Spring

❖ Po zavedení jiného systému barev a hranových výrazů můžeme náš systém sdílení zdrojů modelovat např. také tak, jak je ukázáno na následujícím slajdu...


color U = with p | q;
color I = int;
color P = product U * I;
color V = with r | s | t;
var x : U;
var i : I;
3`(q,0)
P
A
if x=q
then 1`(q,i+1)
else empty
(x,i)
T1
[x=q]
(x,i)
2`(p,0)
P
B
1`r+1`s
(x,i)
if x=p
then 1`(p,i+1)
else empty
case x of
p => 2`s
| q => 1`s
T2
(x,i)
P
C
(x,i)
if x=p then 1`t
else empty
1`r + 3`s + 2`t
Res
V
T3
if x=q then 1`r
else empty
(x,i)
P
D
t
(x,i)
T4
(x,i)
case x of
p => 2`s+2`t
| q => 2`s+1`t
P
E
(x,i)
T5

❖ A konečně po zavedení ještě jiného systému barev a hranových výrazů můžeme náš systém sdílení zdrojů modelovat také takto:

```
color U = with p | q;
color S = with a | b | c | d | e;
color I = int;
color P = product U *S * I;
color R = with r | s | t;
fun Succ(y) = case y of a=>b |  b=>c |  c=>d |  d=>e |  e=>a;
fun Next(x,y,i) = (x, if (x,y) = (p,e) then b else Succ(y), if y=e then i+1 else i);
fun Reserve(x,y) = case (x,y) of (p,b)=>2`s | (p,c)=>1`t | (p,d)=>1`t
                                 | (q,a)=>1`r+1`s | (q,b)=>1`s | (q,d)=>1`t | _=>empty;
fun Release(x,y) = case (x,y) of (p,e)=>2`s+2`t | (q,c)=>1`r | (q,e)=>2`s+1`t | _=>empty;
var x : U;
var y : S;
var i : I;
```



❖ Výše uvedený příklad demonstruje mj. skutečnost, že při použití CPN máme volbu, které rysy systému popsat Petriho sítí a které výpočtem v použitém inskripčním jazyce.