Barvené Petriho sítě

Úvod do CPN

- Barvené Petriho sítě (Coloured Petri Nets – CPNs):
  - 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:

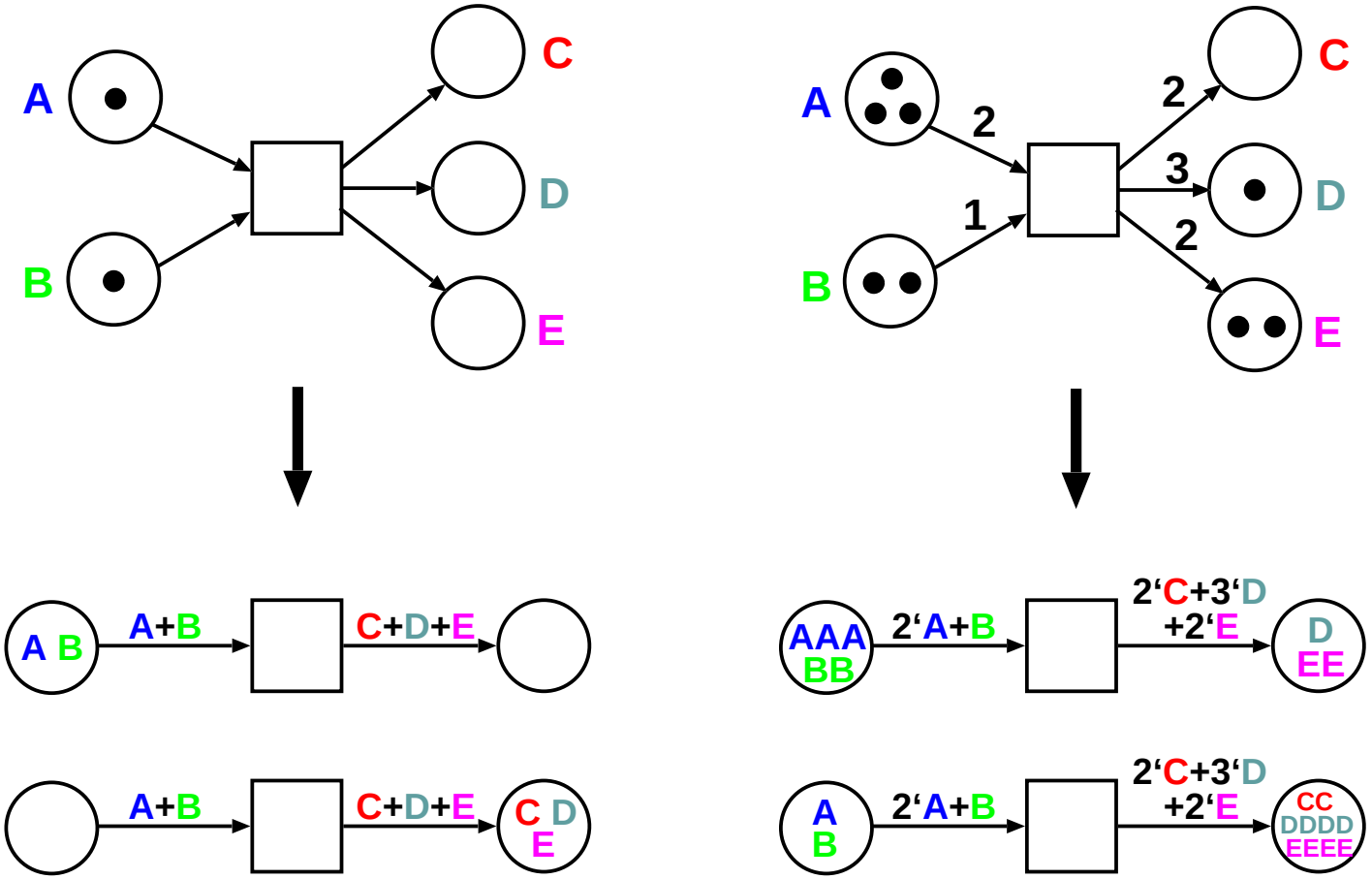
- komunikační protokoly a sítě,
- software (části SW Nokia, bankovní transakce, distribuované algoritmy, ...),
- hardware,
- řídící systémy,
- vojenské systémy,
- ...

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

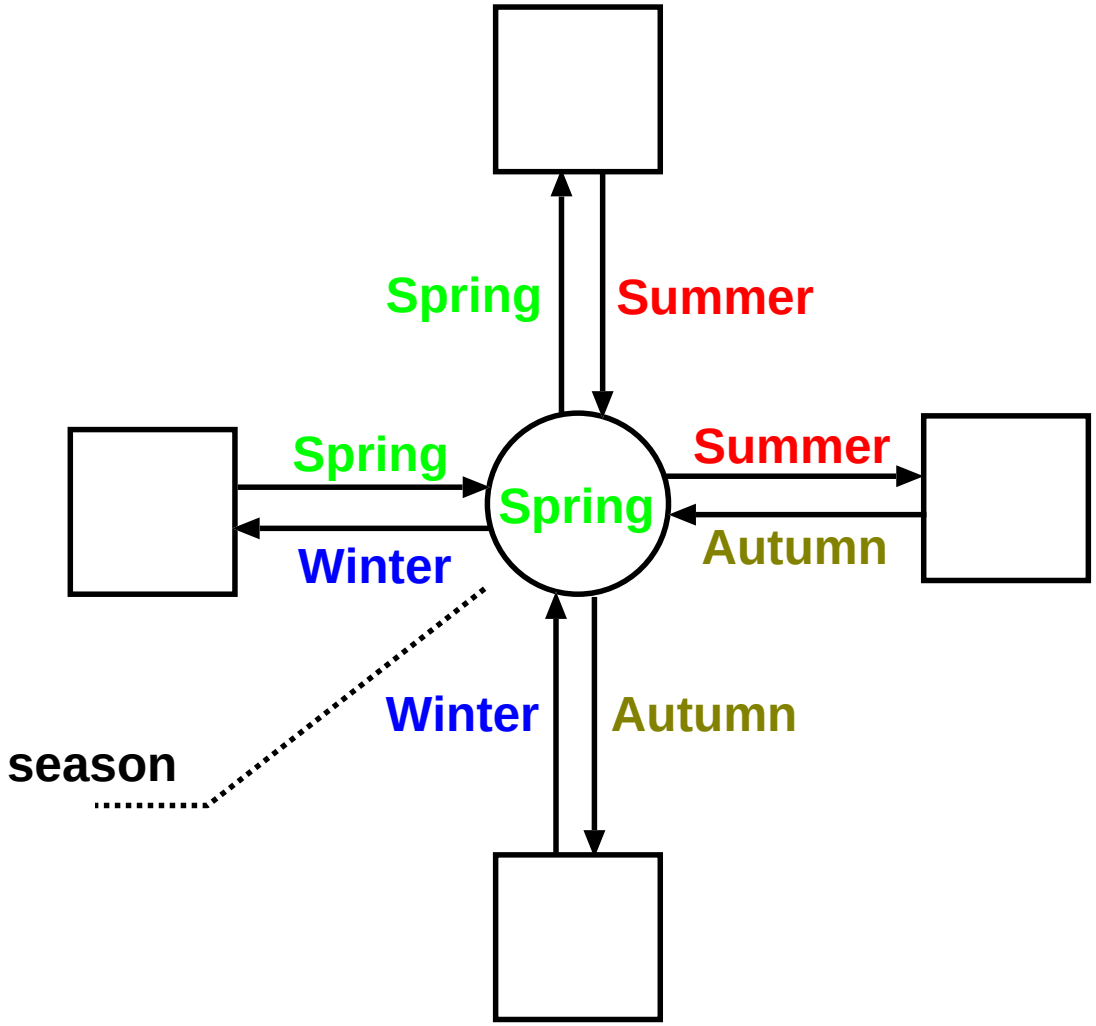
❖ CPN jsou základem pro další rozšíření: hierarchické CPN či různé objektově-orientované Petriho sítě (PNtalk, Renew, ...).

Petriho sítě s individuálními značkami

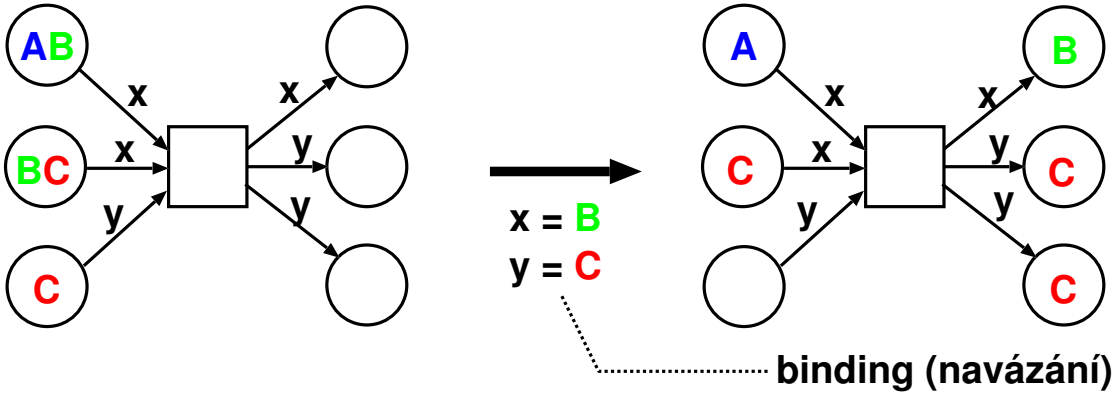
- Individual Token Nets with Constant Arrow Labels:

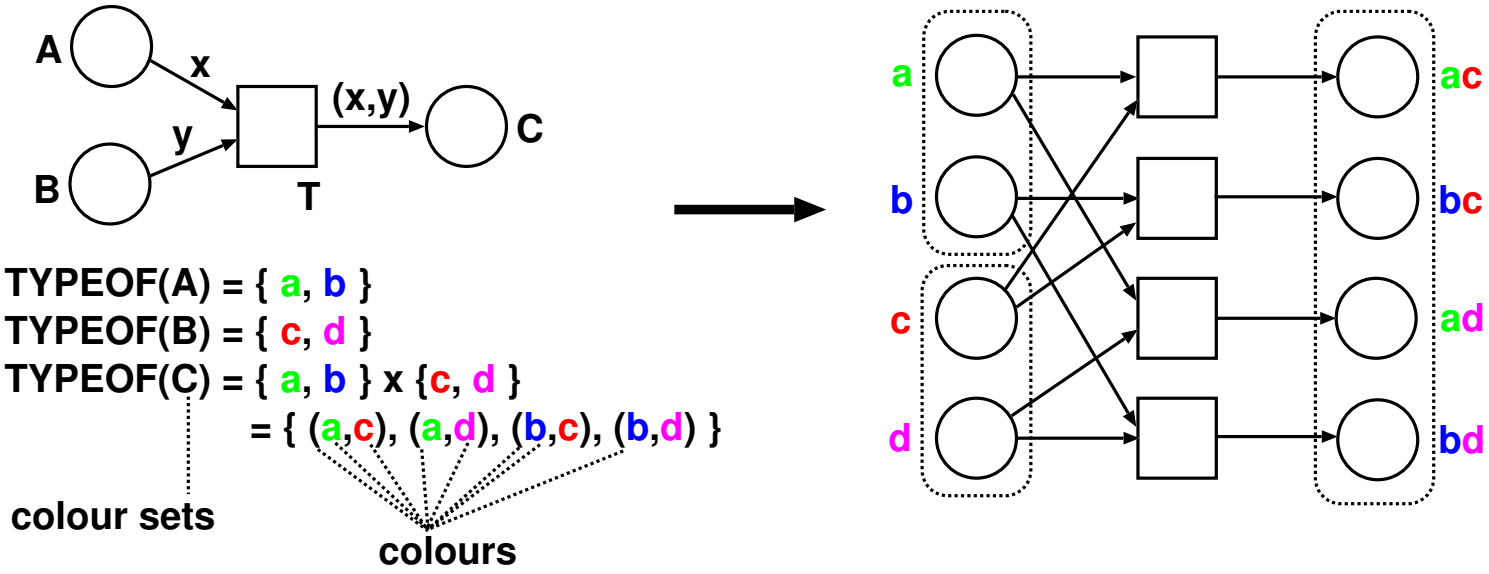


❖ Další jednoduchý příklad – změna ročních období:



❖ Individual Token Nets with Variable Arrow Labels:



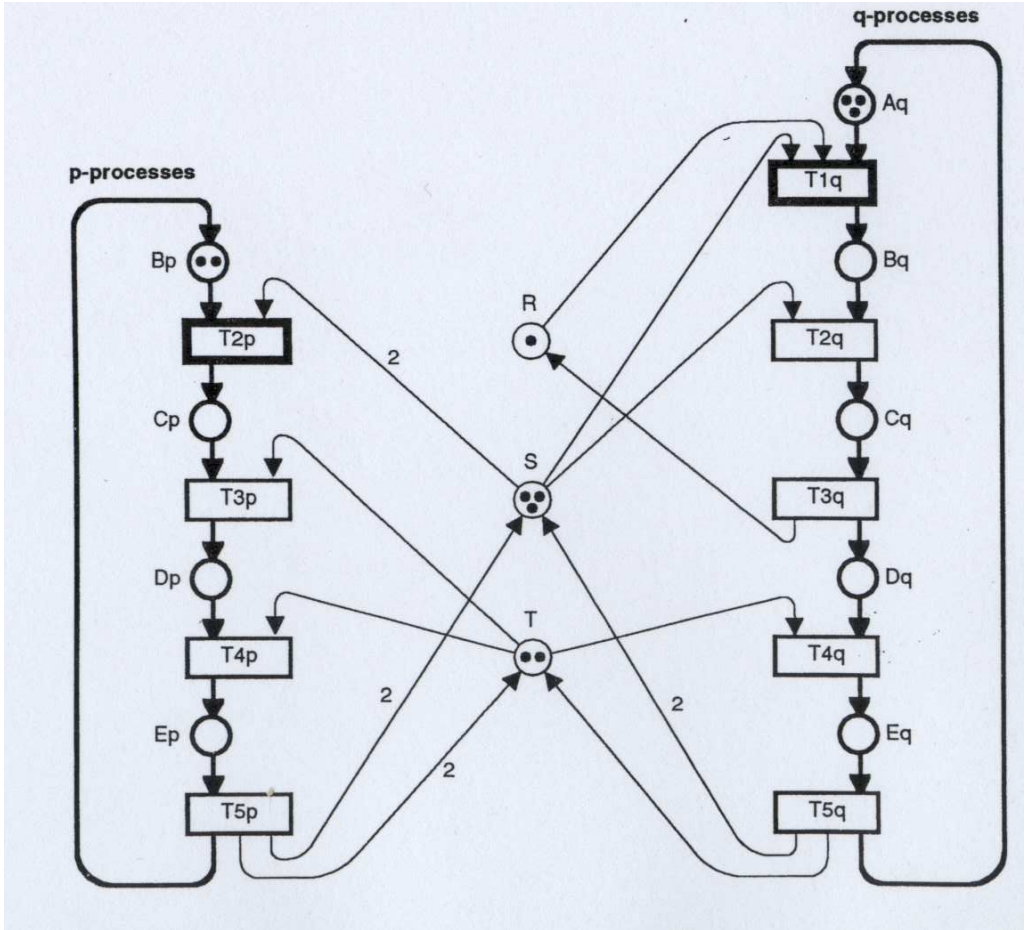


Neformální zavedení CPN

❖ 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 `p` a `q`. Budeme registrovat, který průchod “alokačním cyklem” daný proces provádí.

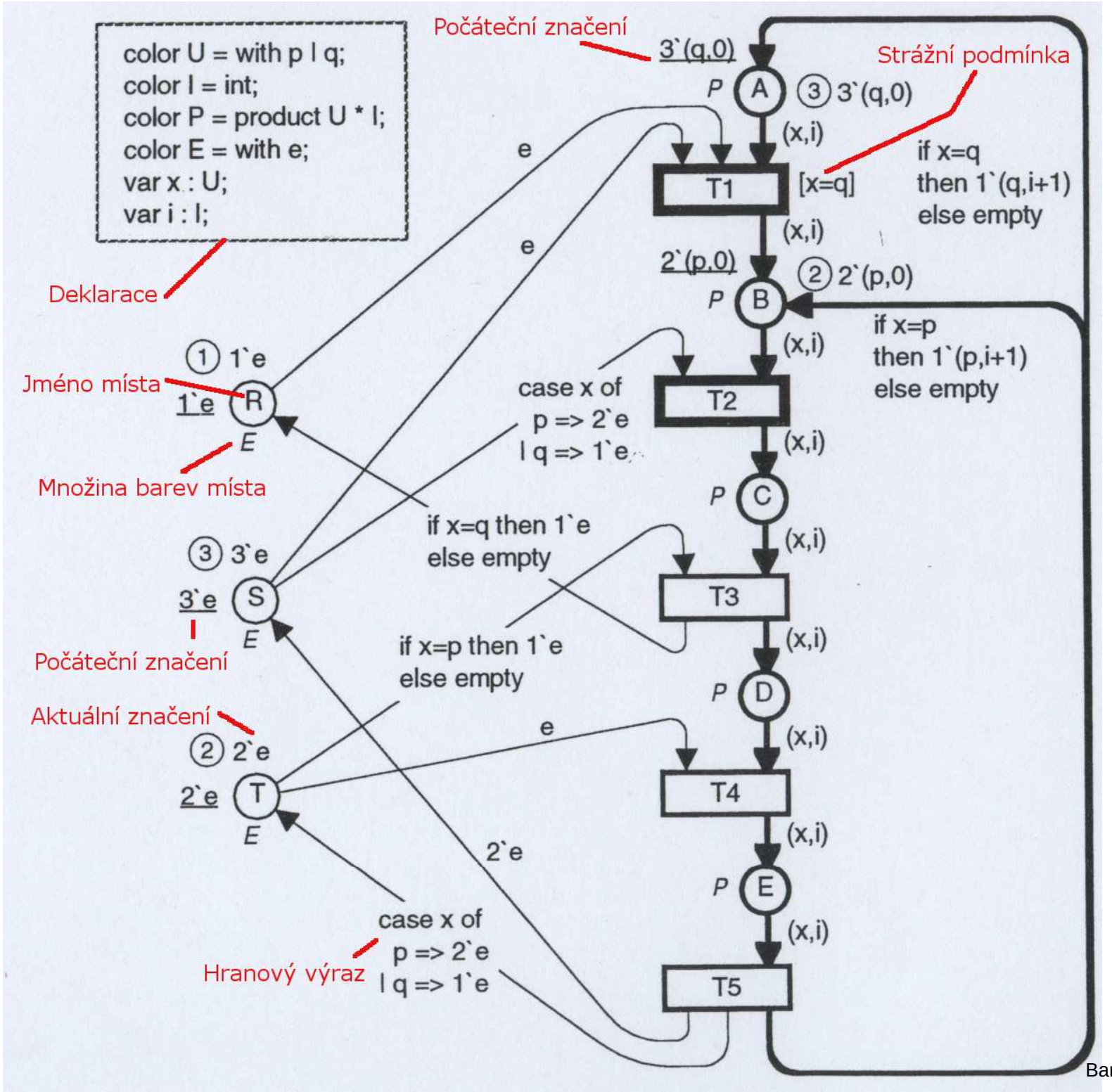
❖ Model ve tvaru CPN zahrnuje dvě složky:

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

❖ Inskripce, vyjádřená inskripčním jazykem, obsahuje:

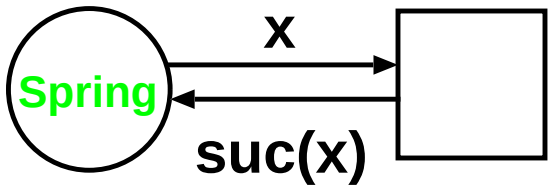
- deklaraci množin barev (coloured sets), tj. datových typů,
- specifikaci množin barev míst,
- popis hran,
- strážní podmínky přechodů,
- počáteční značení,
- (jména míst a přechodů).

❖ 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
P
A
3
3`(q,0)
(x,i)
e
T1
[x=q]
if x=q
then 1`(q,i+1)
else empty
(x,i)
e
2`(p,0)
2
2`(p,0)
P
B
if x=p
then 1`(p,i+1)
else empty
(x,i)
1
1`e
Jméno místa
1`e
R
E
Množina barev místa
case x of
p => 2`e
| q => 1`e
T2
(x,i)
P
C
(x,i)
if x=q then 1`e
else empty
3
3`e
3`e
S
E
T3
(x,i)
Počáteční značení
if x=p then 1`e
else empty
P
D
(x,i)
Aktuální značení
e
2
2`e
2`e
T
E
T4
(x,i)
2`e
P
E
(x,i)
case x of
p => 2`e
| q => 1`e
Hranový výraz
T5

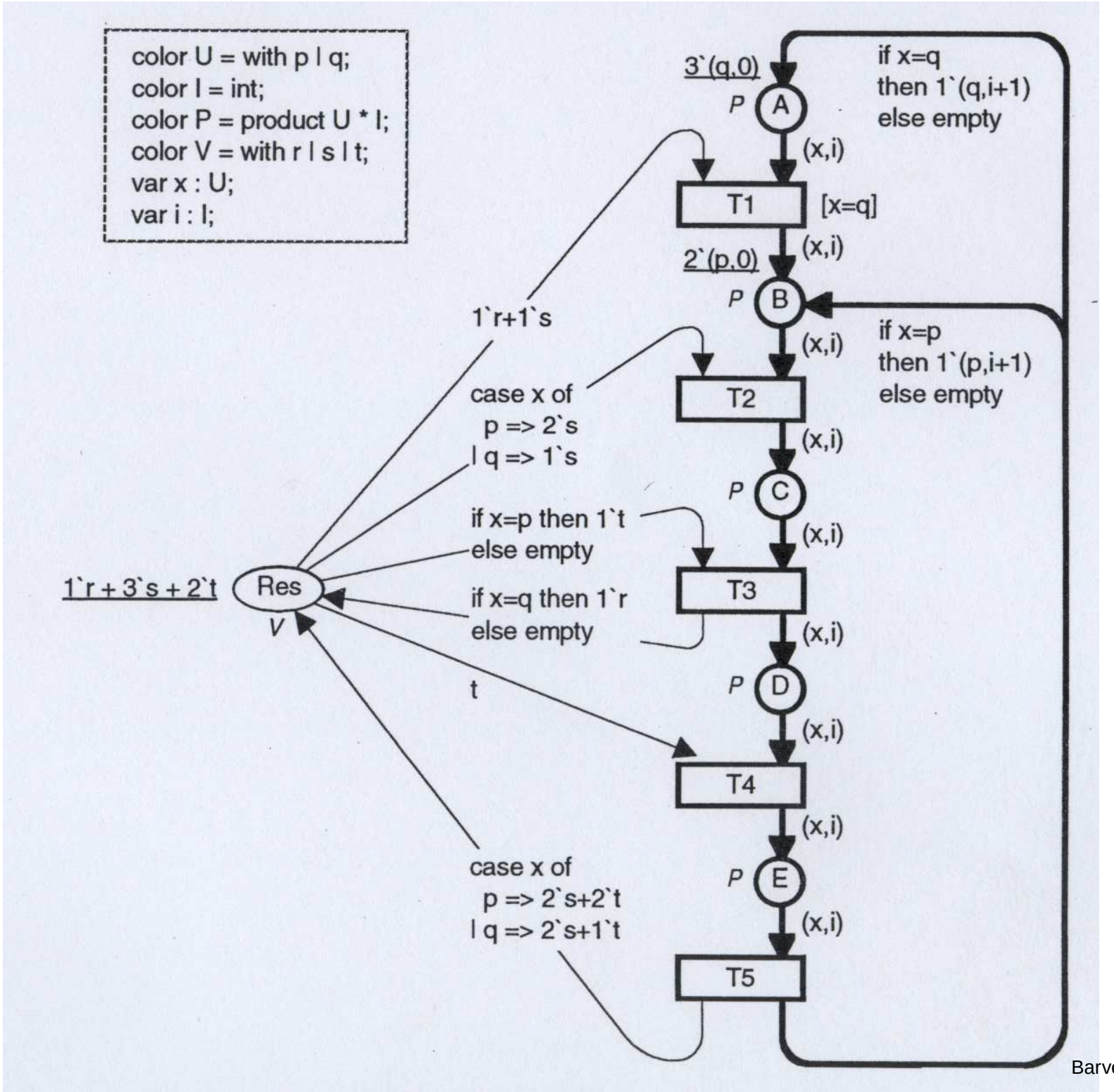
❖ Každý hranový výraz se vyhodnotí na multimnožinu značek:

- 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:



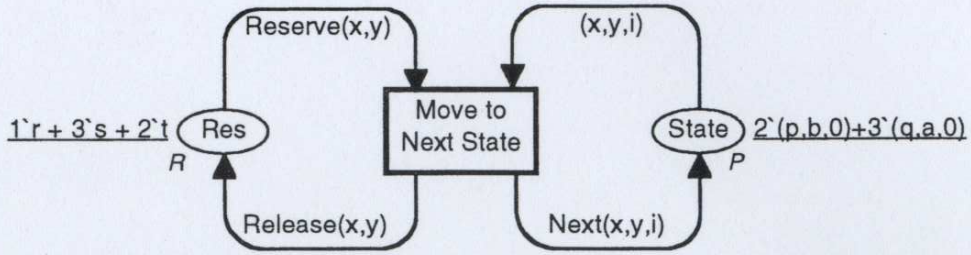
suc(Spring)=Summer
suc(Summer)=Autumn
suc(Autumn)=Winter
suc(Winter)=Spring

❖ 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
if x=p then 1`t
else empty
(x,i)
1`r + 3`s + 2`t
Res
V
if x=q then 1`r
else empty
T3
(x,i)
t
P
D
(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:

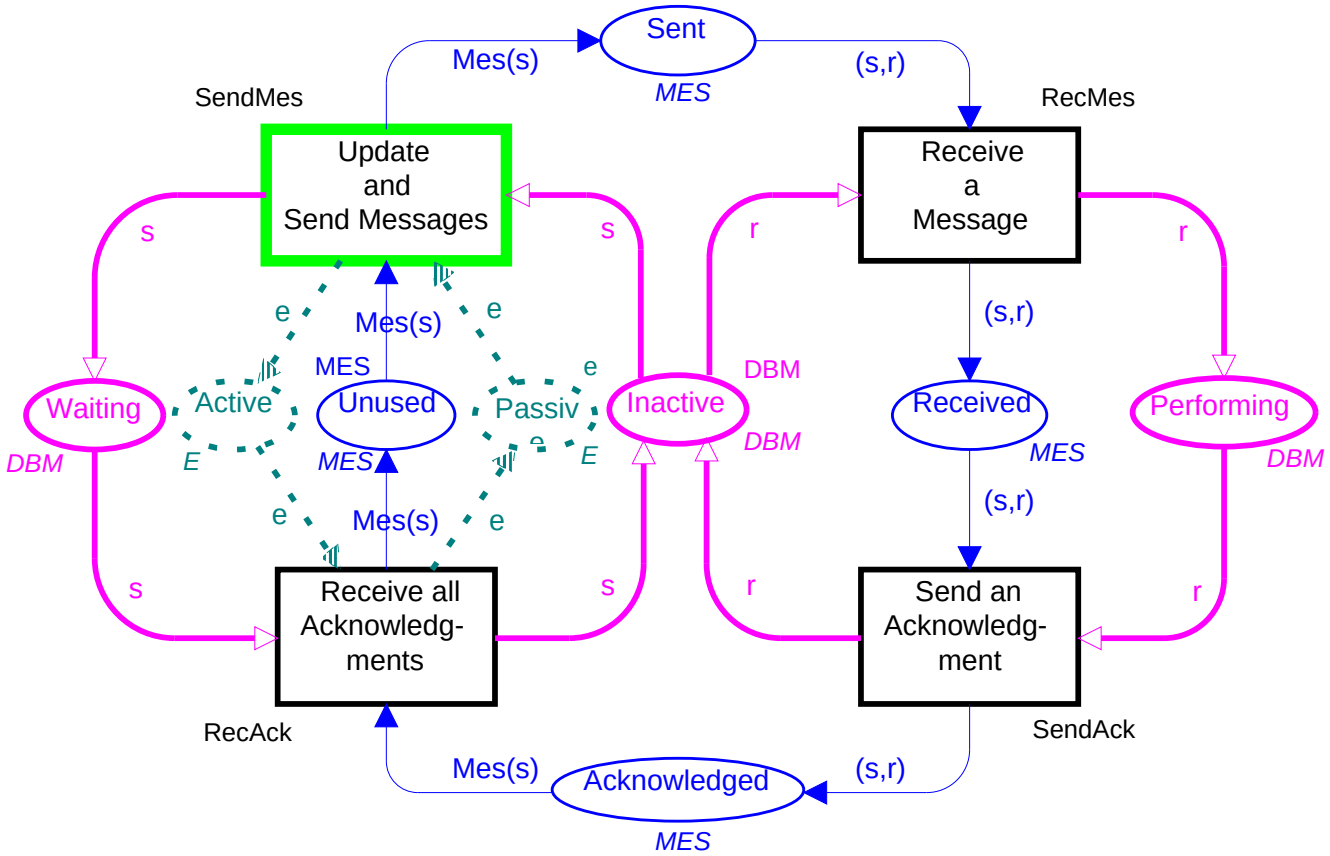
```
color U = with p | q;
color S = with a | b | c | d | e;
color I = int;
color P = product U *S * I;
color R = with r | s | t;
fun Succ(y) = case y of a=>b |  b=>c |  c=>d |  d=>e |  e=>a;
fun Next(x,y,i) = (x, if (x,y) = (p,e) then b else Succ(y), if y=e then i+1 else i);
fun Reserve(x,y) = case (x,y)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

❖ Jiný příklad: databáze je distribuována do n míst (sites); každé místo obsahuje kopii všech dat, o kterou se stará systém správy databáze (DBM):

- $DBM = \{d_1, ..., d_n\}$,
- zprávy zasílané mezi DBM: $MES = \{(s, r) \mid s, r \in DBM \wedge s \neq r\}$, kde s – sender, r – receiver,
- $Mes(s) = \sum_{r \in DBM \setminus \{s\}} 1`(s, r)$.




```
val n = 4;
color DBM = index d with 1..n  declare ms;
color PR = product DBM * DBM  declare mult;
fun diff(x,y) = (x<>y);
color MES = subset PR by diff  declare ms;
color E = with e;
fun Mes(s) = mult'PR(1`s,DBM--1`s);
var s, r : DBM;
```

Formální definice multimnožiny

- Multimnožina m nad množinou S je funkce $m : S \to \mathbb{N}$:
  - m(s) značí počet výskytů prvku s v multimnožině m.
  - Multimnožinu m obvykle reprezentujeme formální sumou $\sum_{s \in S} m(s)`s$.

- Symbolem S_{MS} značíme množinu všech multimnožin nad S.

- Jestliže $m(s) \neq 0$, pak říkáme, že s patří do m a píšeme $s \in m$.

- Pro multimnožiny je definována:
  - operace sjednocení $m_1 + m_2 = \sum_{s \in S}(m_1(s) + m_2(s))`s$,
  - skalární multiplikace,
  - predikáty $=, \neq, \leq, \geq$ (např. $m_1 \leq m_2 \Leftrightarrow \forall s \in S : m_1(s) \leq m_2(s)$),
  - kardinalita $|m| = \sum_{s \in S} m(s)$ a
  - je-li $m_1 \leq m_2$, pak také rozdíl $m_2 - m_1$.

Formální definice syntaxe CPN

❖ Některé pomocné funkce a pojmy:

- Vycházíme z dané konečné množiny Σ konečných množin barev (Σ je tedy množina typů značek).
- Typ proměnné v budeme značit $Type(v)$.
- Je-li V množina proměnných, pak $Type(V) = \{Type(v) \mid v \in V\}$.
- Typ výrazu $expr$ budeme značit $Type(expr)$.
- Množinu proměnných výrazu $expr$ budeme značit $Var(expr)$.
- Uzavřeným výrazem rozumíme výraz $expr$ bez proměnných, tj. $Var(expr) = \emptyset$.
- Označme $EXPR$ množinu všech výrazů přípustných ve zvoleném inskripčním jazyce.
- Označme $CEXPR \subseteq EXPR$ množinu všech uzavřených výrazů přípustných ve zvoleném inskripčním jazyce.
- Konečně označme $\mathbb{B} = \{\texttt{true}, \texttt{false}\}$.

❖ Nehierarchická barvená Petriho síť CPN je n-tice $CPN = (\Sigma, P, T, A, N, C, G, E, I)$, kde:

1. Σ je konečná množina konečných, neprázdných typů nazývaných množinami barev.
2. P je konečná množina míst.
3. T je konečná množina přechodů taková, že $P \cap T = \emptyset$.
4. A je konečná množina hran taková, že $A \cap P = A \cap T = \emptyset$.
5. N je uzlová funkce (node function) $N : A \rightarrow P \times T \cup T \times P$.
6. C je funkce barev (colour function) $C : P \rightarrow \Sigma$.
7. G je funkce strážních podmínek (guard function) $G : T \rightarrow EXPR$ taková, že $\forall t \in T : Type(G(t)) = \mathbb{B} \wedge Type(Var(G(t))) \subseteq \Sigma$.
8. E je funkce hranových výrazů (arc expression function) $E : A \rightarrow EXPR$ taková, že $\forall a \in A : Type(E(a)) = C(p(a))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma$, přičemž $p(a)$ je místo v $N(a)$.
9. I je inicializační funkce $I : P \rightarrow CEXPR$ taková, že $\forall p \in P : Type(I(p)) = C(p)_{MS}$.

❖ Naši první CPN verzi systému alokace sdílených zdrojů můžeme dle definice zapsat takto:

(i) Σ = {U, I, P, E}.

(ii) P = {A, B, C, D, E, R, S, T}.

(iii) T = {T1, T2, T3, T4, T5}.

(iv) A = {AtoT1, T1toB, BtoT2, T2toC, CtoT3, T3toD, DtoT4, T4toE, EtoT5, T5toA, T5toB, RtoT1, StoT1, StoT2, TtoT3, TtoT4, T3toR, T5toS, T5toT}.

(v) N(a) = (SOURCE,DEST) if a is in the form SOURCEtoDEST.

(vi) C(p) =

P	if p∈{A, B, C, D, E}
E	otherwise.

(vii) G(t) =

x=q	if t=T1
true	otherwise.

(viii) E(a) =

e	if a∈{RtoT1, StoT1, TtoT4}
2`e	if a=T5toS
case x of p=>2`e | q=>1`e	if a∈{StoT2, T5toT}
if x=q then 1`e else empty	if a=T3toR
if x=p then 1`e else empty	if a=TtoT3
if x=q then 1`(q,i+1) else empty	if a=T5toA
if x=p then 1`(p,i+1) else empty	if a=T5toB
(x,i)	otherwise.

(ix) I(p) =

3`(q,0)	if p=A
2`(p,0)	if p=B
1`e	if p=R
3`e	if p=S
2`e	if p=T
Ø	otherwise.

Formální definice sémantiky CPN

- Navázání množiny proměnných V přiřazuje každému $v \in V$ prvek $b(v) \in Type(v)$.

- Hodnotu získanou vyhodnocením $expr$ při navázání b budeme značit $expr\langle b \rangle$.

- Nechť $Var(t)$ je množina všech proměnných vyskytujících se ve výrazech na hranách přilehlých k přechodu t, respektive ve stráži t.

- Navázání přechodu t je funkce b definovaná na $Var(t)$ tak, že:

  1. $\forall v \in Var(t) : b(v) \in Type(v)$.
  2. $G(t)\langle b \rangle$.

$B(t)$ pak značí množinu všech navázání přechodu t.

❖ Element značení (token element) je dvojice (p, c), kde $p \in P$ a $c \in C(p)$.

- Množinu všech elementů značení značíme TE.
- Značení je pak multimnožina nad TE.
- Množinu všech značení značíme $\mathbb{M}$.
- Počáteční značení M_0 je definováno vyhodnocením inicializační funkce: $\forall (p, c) \in TE : M_0((p, c)) = (I(p))(c)$.

❖ Element navázání (binding element) je dvojice (t, b), kde $t \in T$ a $b \in B(t)$.

1. Množinu všech elementů navázání značíme BE.
2. Krokem pak rozumíme neprázdnou a konečnou multimnožinu nad BE.
3. Množinu všech kroků značíme $\mathbb{Y}$.

❖ Krok $Y \in \mathbb{Y}$ je proveditelný (enabled) ve značení $M \in \mathbb{M}$, jestliže platí následující:

$$\forall p \in P : \sum_{(t,b)\in Y} E(p,t)\langle b\rangle \leq M(p)$$

❖ Je-li Y proveditelný, pak:

- Říkáme, že každý $(t,b) \in Y$ je proveditelný, a také, že t je proveditelný (pro navázání b).
- Pro $(t_1,b_1),(t_2,b_2) \in Y$, $(t_1,b_1) \neq (t_2,b_2)$, říkáme, že $(t_1,b_1),(t_2,b_2)$ jsou současně proveditelné (concurrently enabled).
- Je-li $Y((t,b)) \geq 2$, říkáme, že (t,b) je současně proveditelný sám se sebou.
- Podobně je-li $|Y(t)| \geq 2$, říkáme, že t je současně proveditelný sám se sebou.

❖ Je-li $Y \in \mathbb{Y}$ proveditelný v $M_1 \in \mathbb{M}$, může být proveden a změnit tak značení na $M_2 \in \mathbb{M}$ takové, že:

$$\forall p \in P : M_2(p) = (M_1(p) - \sum_{(t,b)\in Y} E(p,t)\langle b\rangle) + \sum_{(t,b)\in Y} E(t,p)\langle b\rangle)$$

Hierarchické barvené Petriho sítě

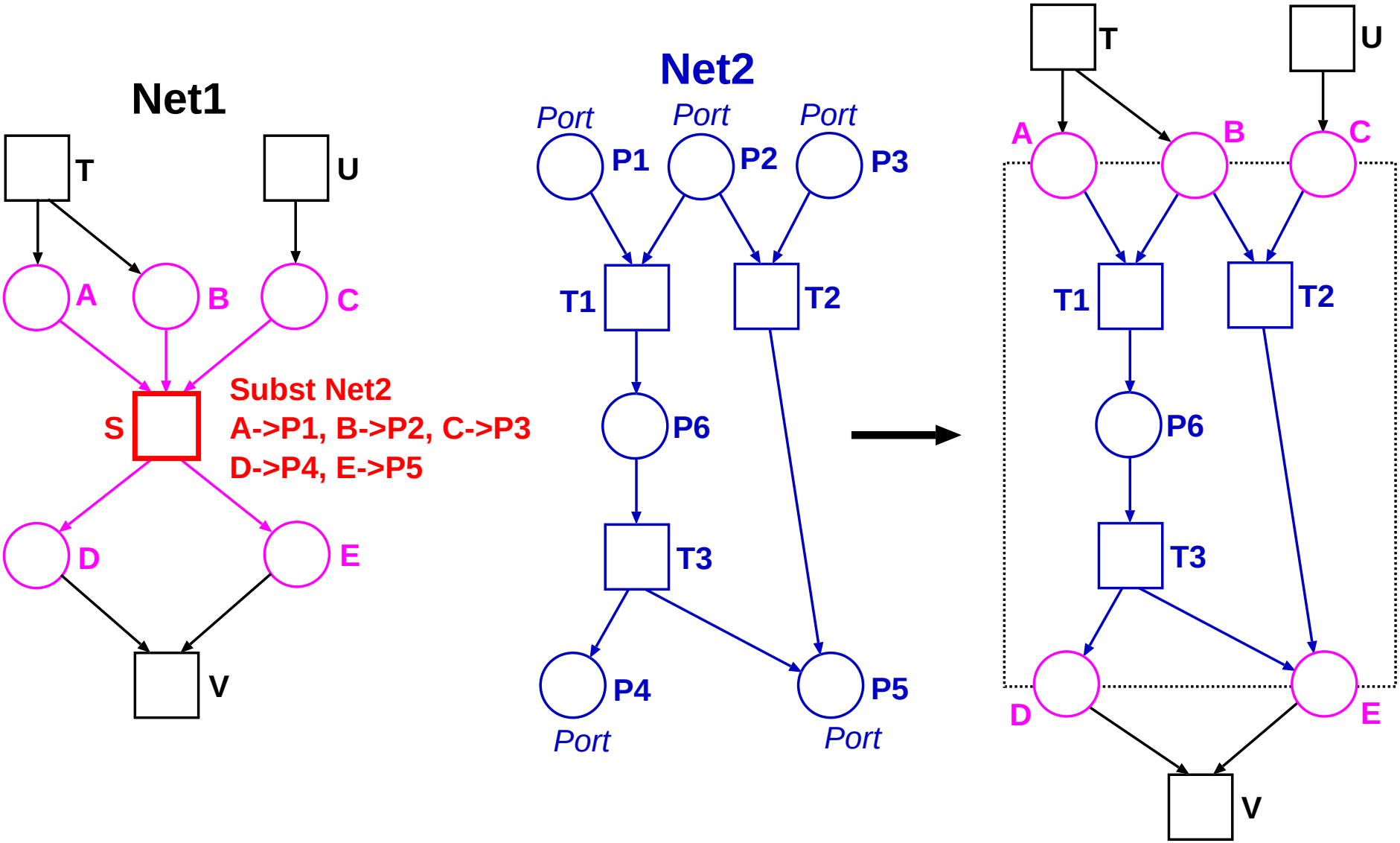
Hierarchické strukturování v PN

❖ Proč strukturování? Modelování a návrh rozsáhlých systémů jsou nemyslitelné na úrovni jednoho, “plochého” modelu.

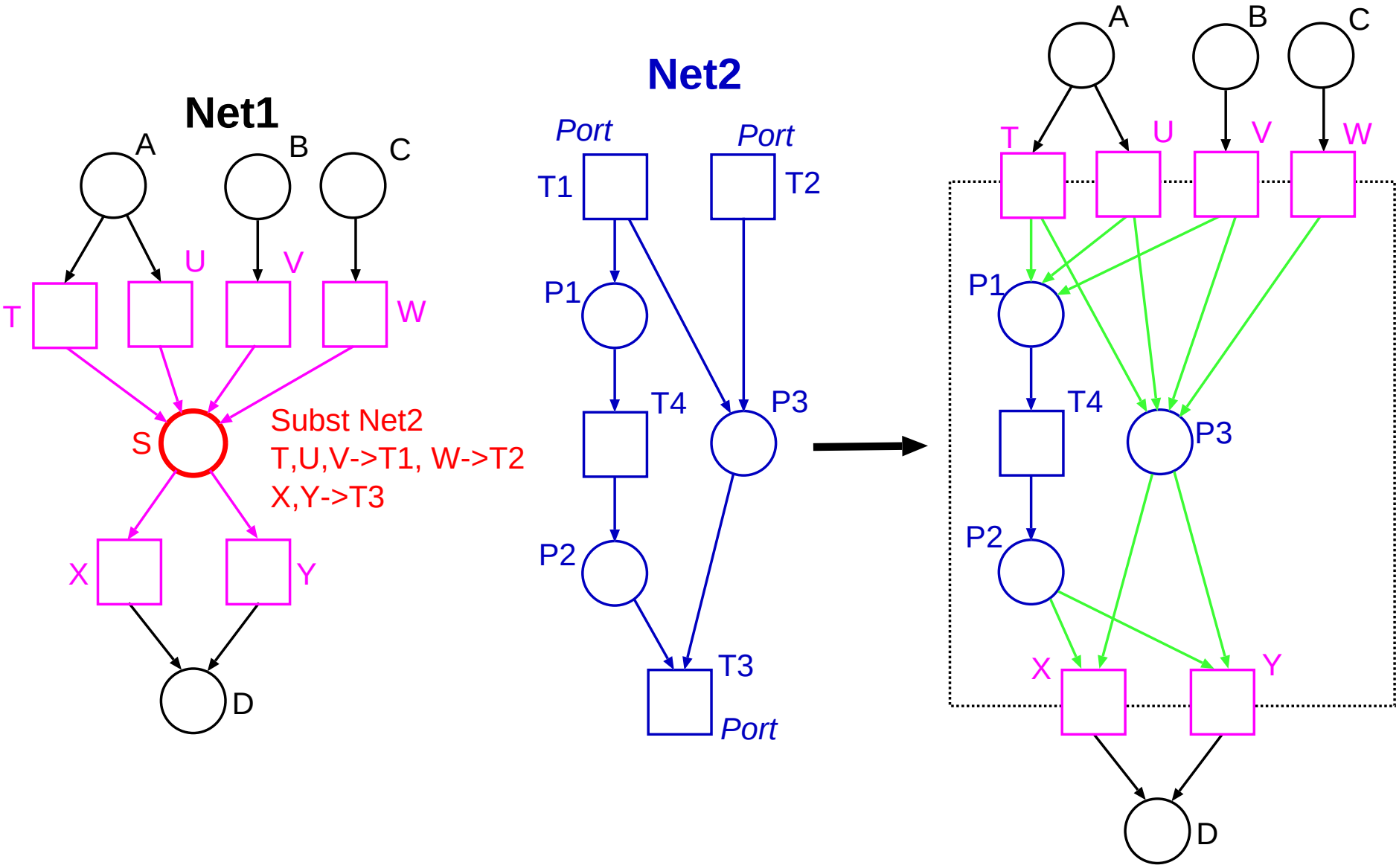
❖ Hierarchické strukturování v PN dle Huber, Jensen, Shapiro:

- substituce přechodů – přechod staticky nahrazen podsítí (podobné rozvoji maker),
- substituce míst – místo staticky nahrazeno podsítí,
- invokace přechodů – přechod *při provádění* dynamicky nahrazen nově vytvořenou kopií příslušné podsítě, která existuje, dokud se neobjeví značka ve zvlášť specifikovaném “výstupním” místě dané podsítě (analogie volání funkce),
- fúze míst – několik míst je staticky spojeno do jednoho (nebo jedno místo je při návrhu rozděleno na několik, aby se předešlo přílišnému křížení hran).

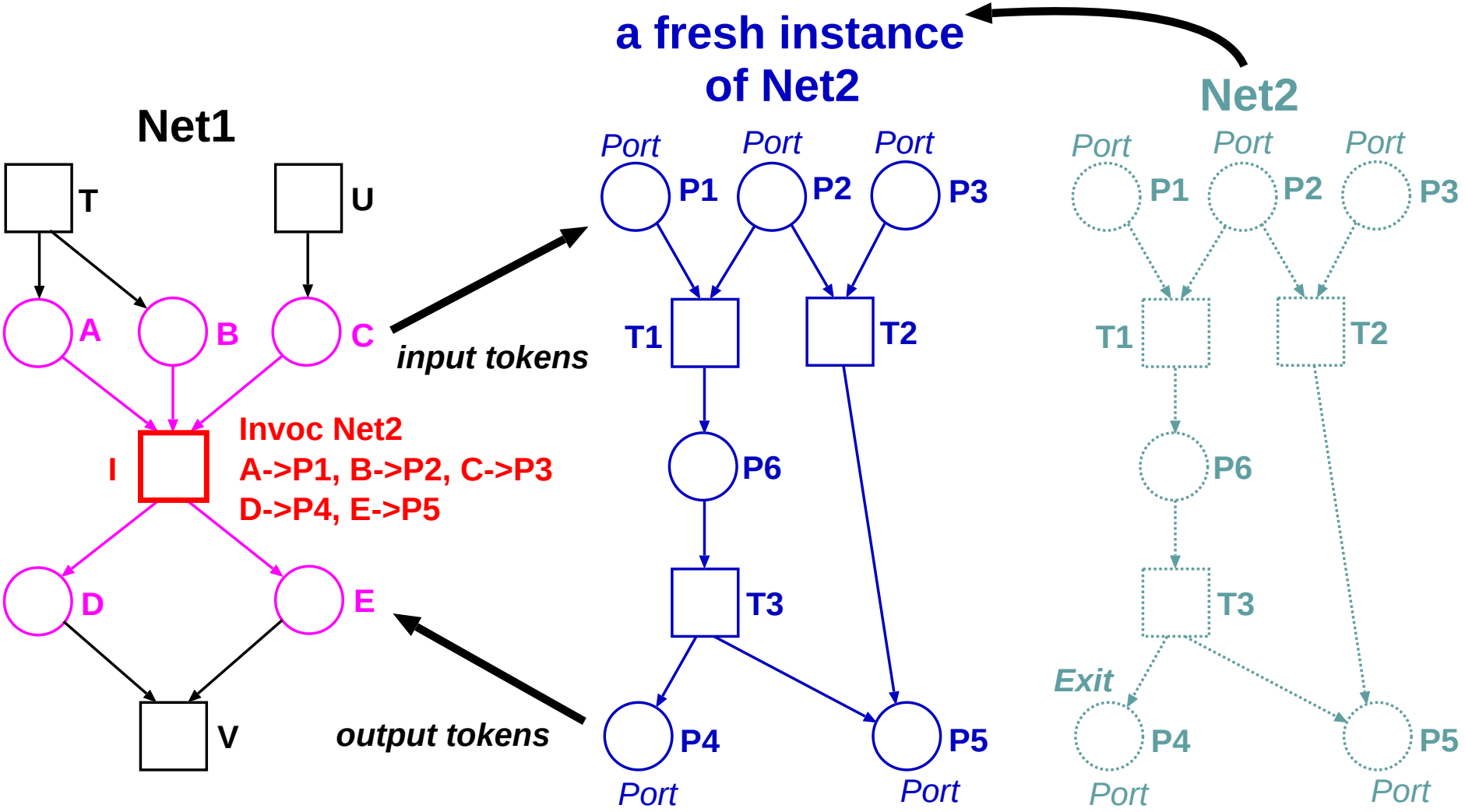
Substituce přechodů



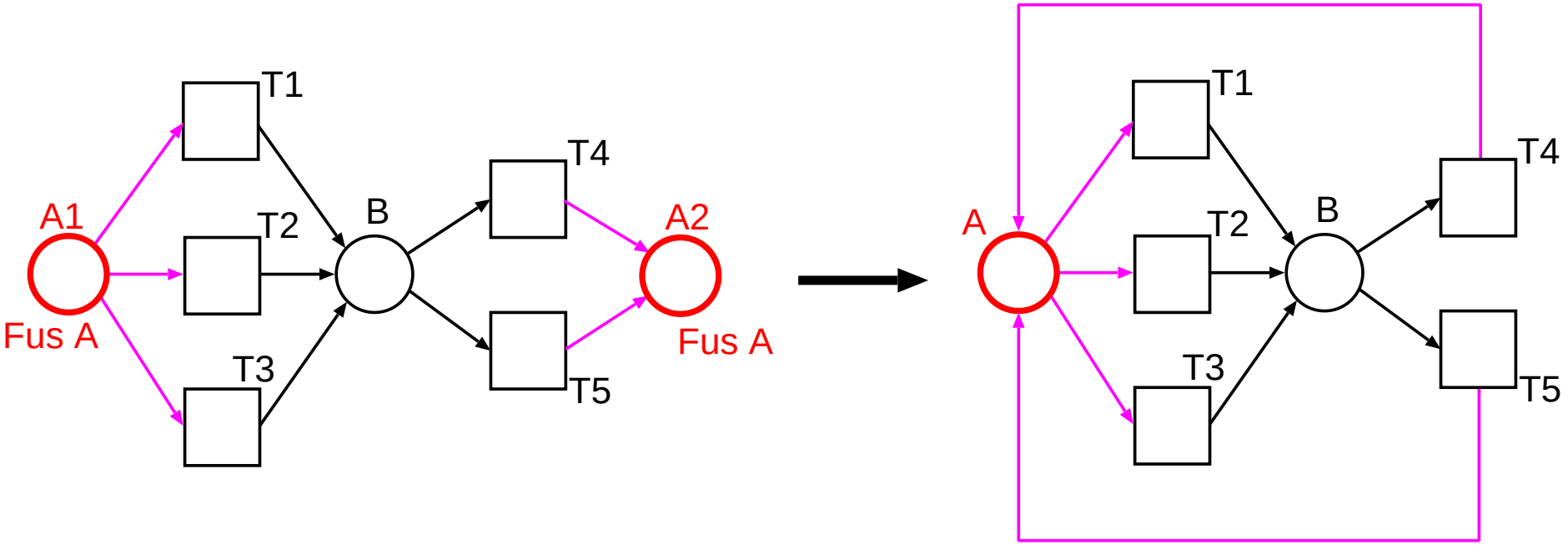
Substituce míst



Invokace přechodů



Fúze míst

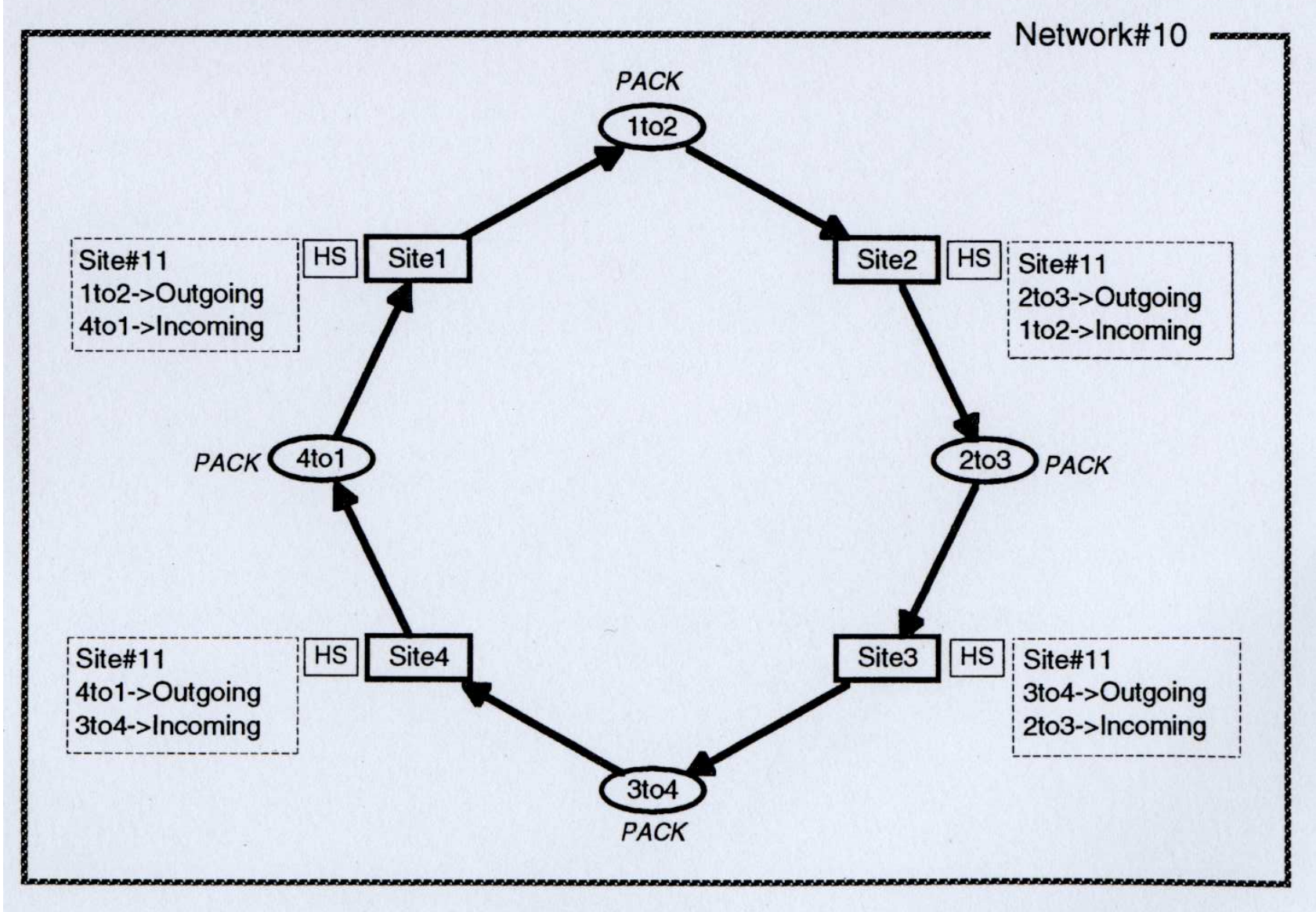


❖ Při kombinaci fúze míst se substitucí přechodů či míst (či s invokací přechodů) je možné zavést různé typy fúze:

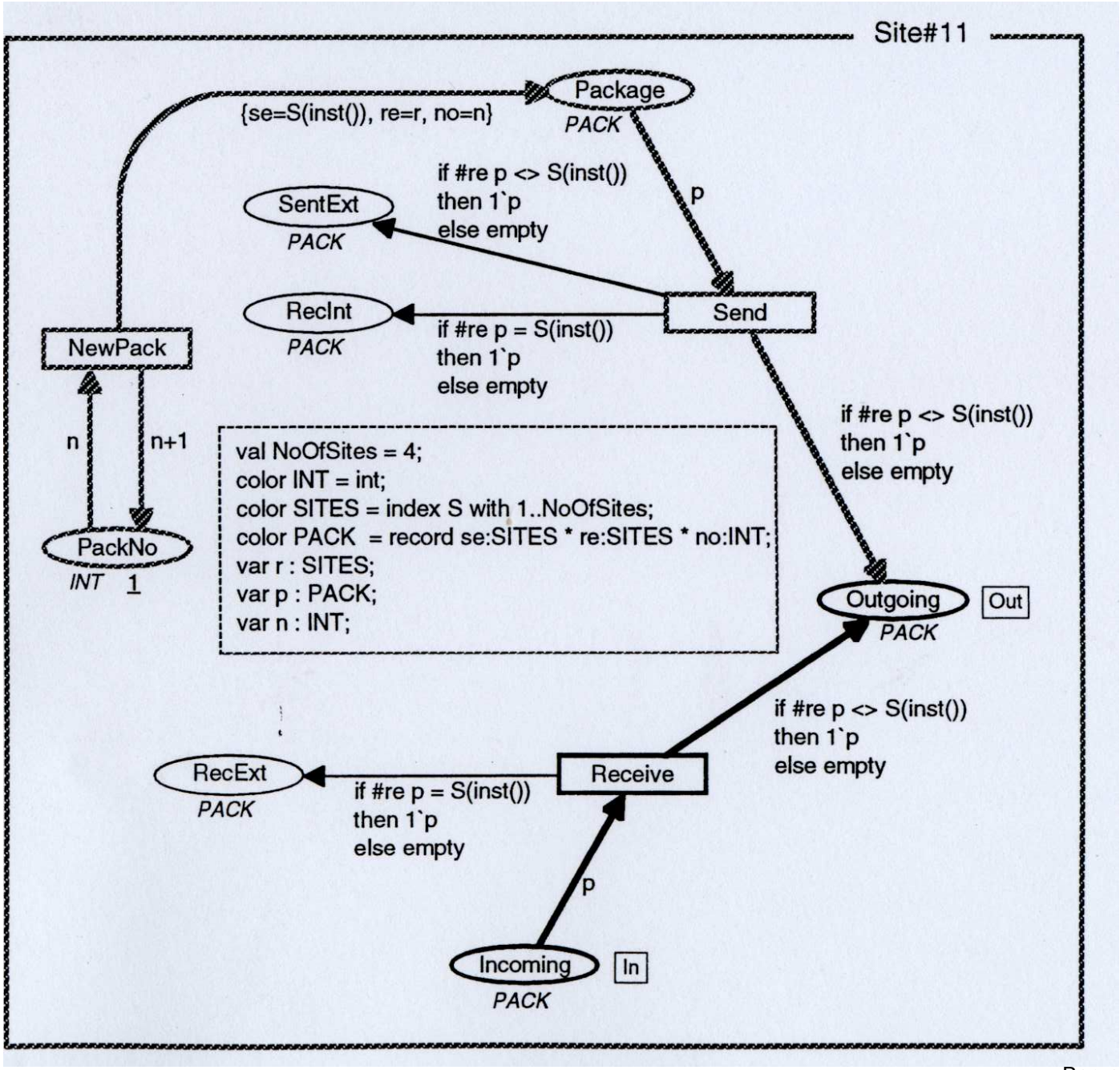
- lokální v rámci jedné instance sítě,
- napříč všemi instancemi dané sítě a
- napříč všemi instancemi všech sítí.

Příklad – hierarchická CPN kruhové sítě

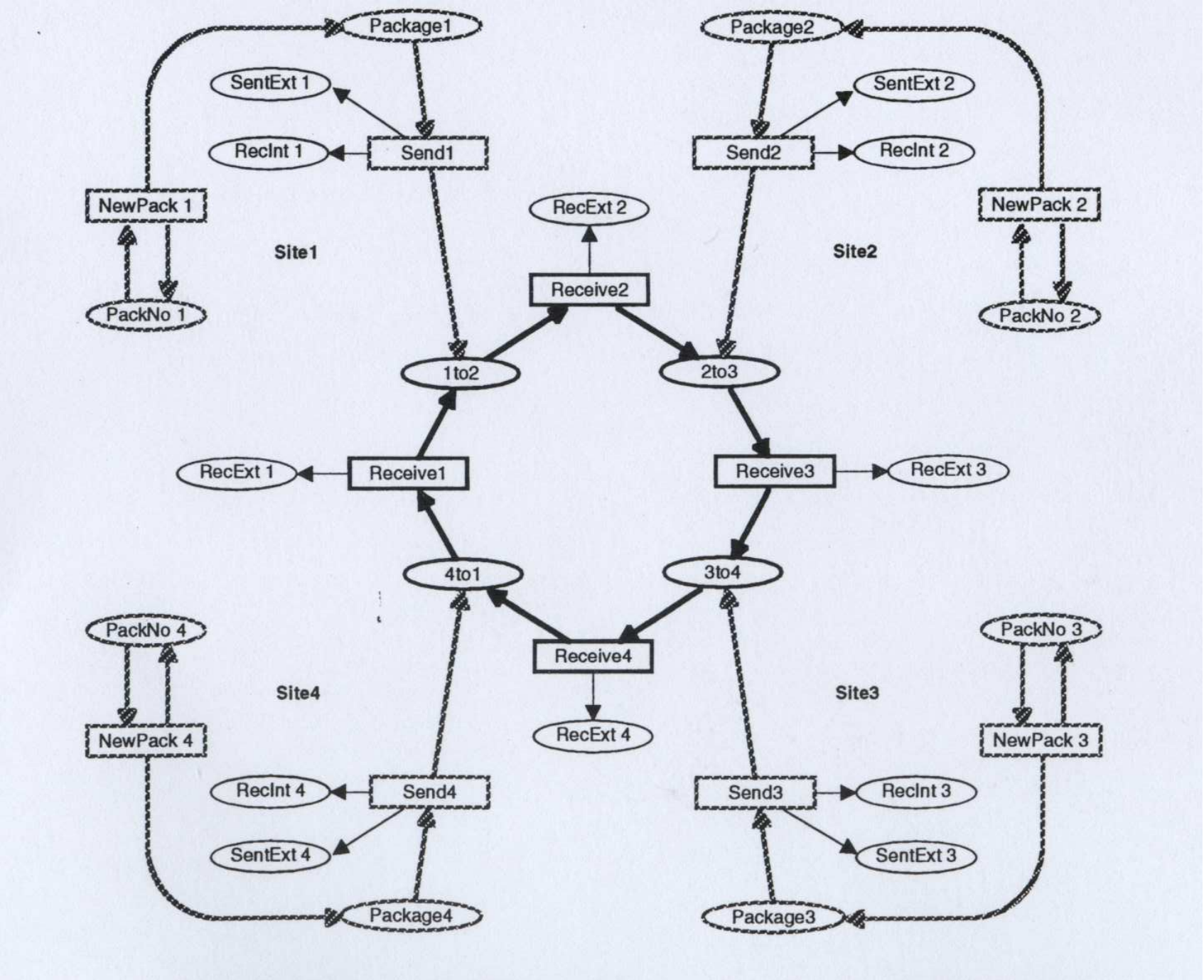
❖ Nejvyšší hierarchická úroveň – model sítě, jejíž jednotlivé uzly jsou popsány dále podsítěmi:



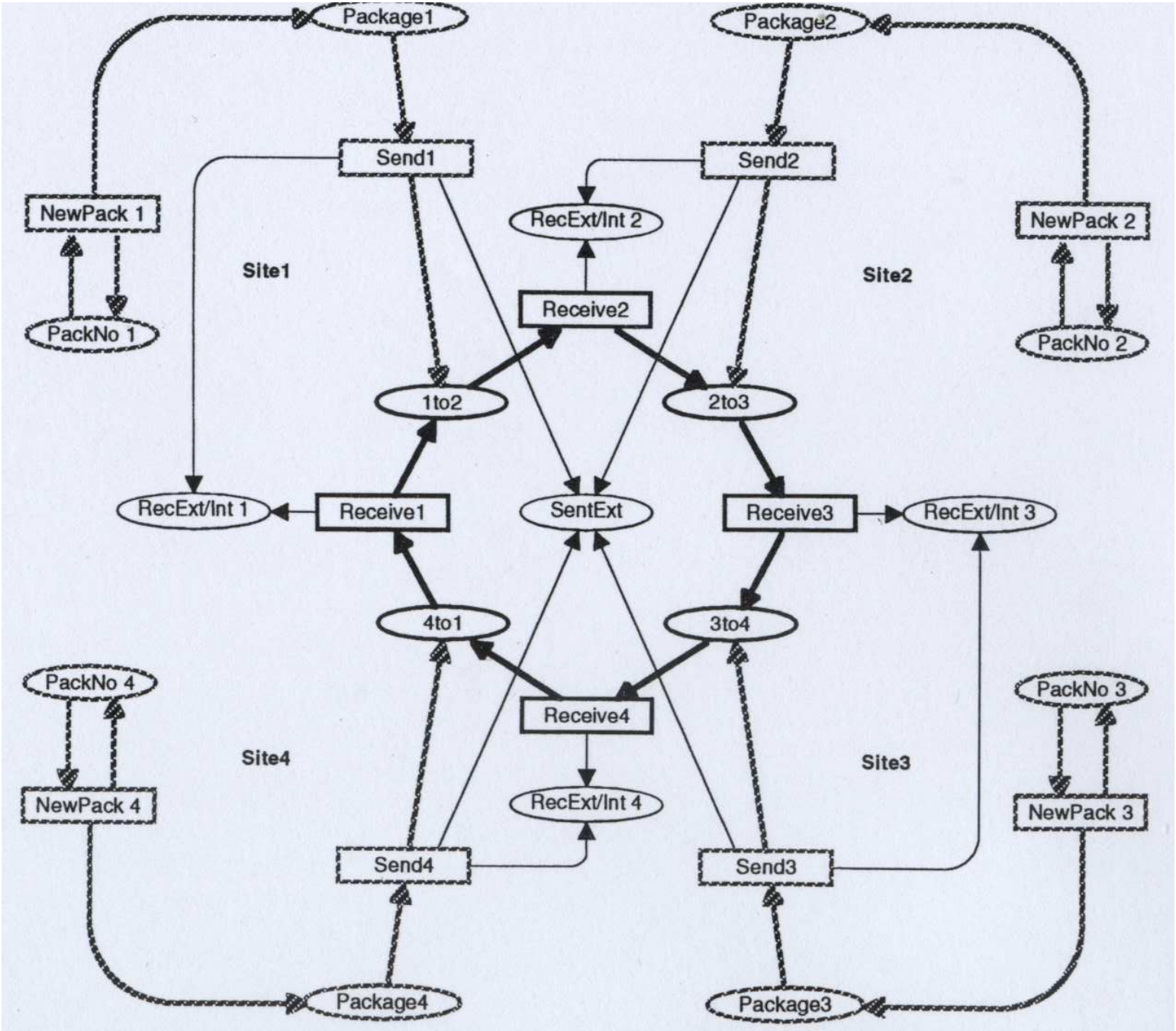
❖ Podsíť modelující jeden uzel sítě (mezi `RecInt` a `RecExt` předpokládáme fúzi v rámci instance sítě a u `SentExt` globální fúzi):



❖ Rozvinutím hierarchie dostaneme tuto CPN (fúze míst ještě není provedena):



❖ Konečně po fúzi míst dostaneme tuto CPN:



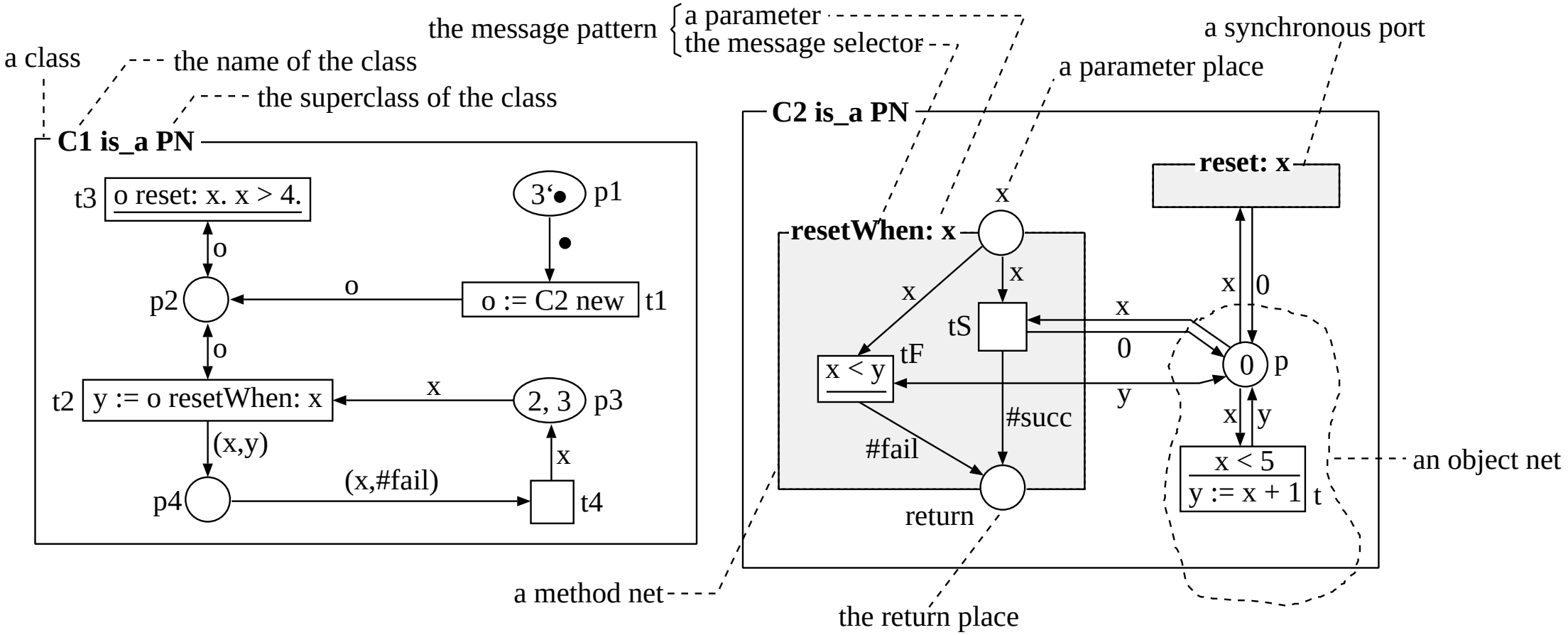
Objektově orientované Petriho sítě

❖ Existují různé koncepty zavedení objektové orientace do Petriho sítí, které v zásadě staví na mechanismech podobných invokaci přechodů.

❖ OOPN/PNtalk vyvinutý na FIT (`http://www.fit.vutbr.cz/~janousek/pntalk/`) pracuje s:

- třídami s dědičností,
- objektovými sítěmi – v každé třídě popisují strukturu stavu jejích instancí a jejich případné aktivní chování,
- metodami – mohou být vyvolány k asynchronní komunikaci s objekty,
- synchronními porty – zvláštní forma přechodů aktivovaná voláním a umožňující synchronní komunikaci s objekty,
- objekty jako instancemi tříd a s běžícími instancemi metod,
- pozdní vazbou.

❖ OOPN/PNtalk umísťuje výpočty do stráží a akcí přechodů (na rozdíl od dříve popsaného konceptu CPN) a používá inskripčním jazyk inspirovaný Smalltalkem (na rozdíl od SML), jak je vidět v níže uvedeném jednoduchém modelu systému s čítači:



❖ Ukázka dědičnosti v OOPN/PNtalk: