

Petri-hálók dinamikus tulajdonságai

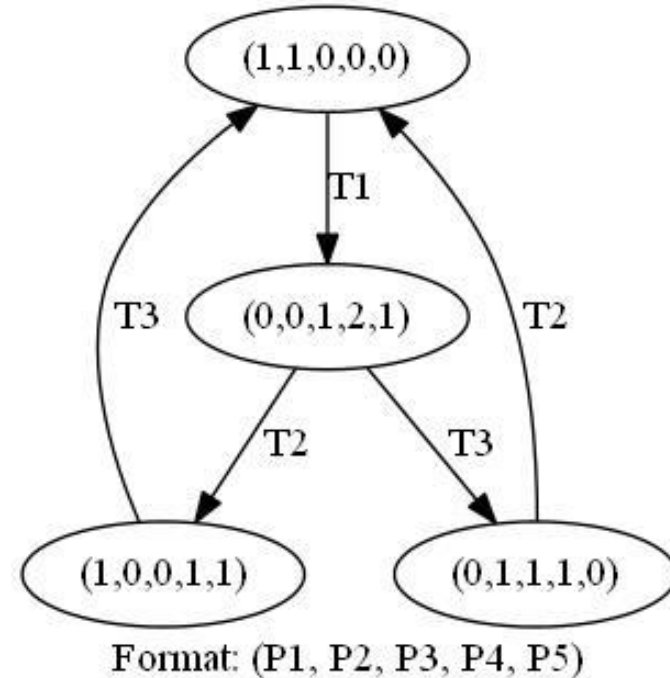
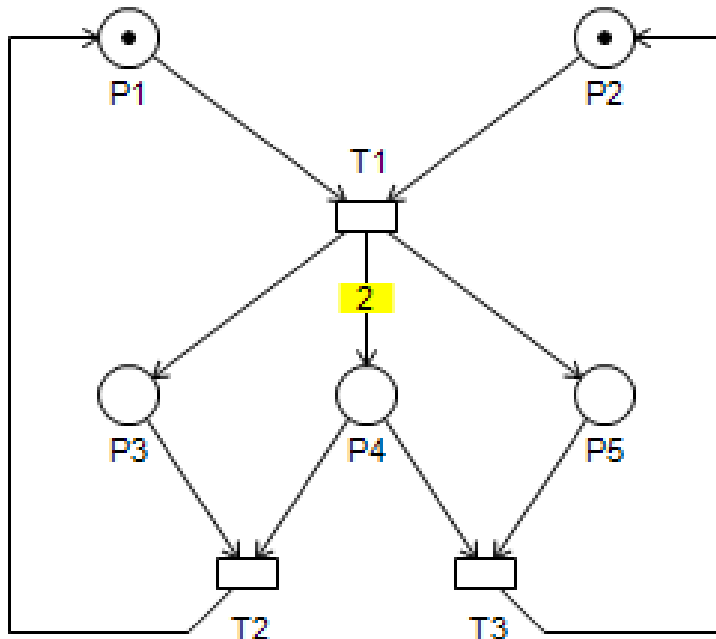
dr. Bartha Tamás

dr. Majzik István

dr. Pataricza András

BME Méréstechnika és Információs Rendszerek Tanszék

Ismétlés: A Petri-hálók működése



Egyszerű Petri-háló és a jelölések változása
(az állapotok elérhetőségi gráfja)

Petri-hálók vizsgálata: Áttekintés a módszerekről

Vizsgálati lehetőségek

Az elemzés mélysége szerint:

- Szimuláció  Egy-egy trajektória bejárása
- Állapottér teljes bejárása  Minden trajektória bejárása adott kezdőállapotból (kimerítő bejárás)
 - Elérhetőségi gráf analízise:
Dinamikus (viselkedési) tulajdonságok
 - Modellellenőrzés
- Háló struktúrájának analízise  Kezdőállapottól független tulajdonságok (bármely kezdőállapotra)
 - Statikus analízis:
Strukturális tulajdonságok
 - Invariáns analízis

ha mindez nem vezet eredményre



- Részleges döntés (pl. absztrakció)

Dinamikus és strukturális tulajdonságok

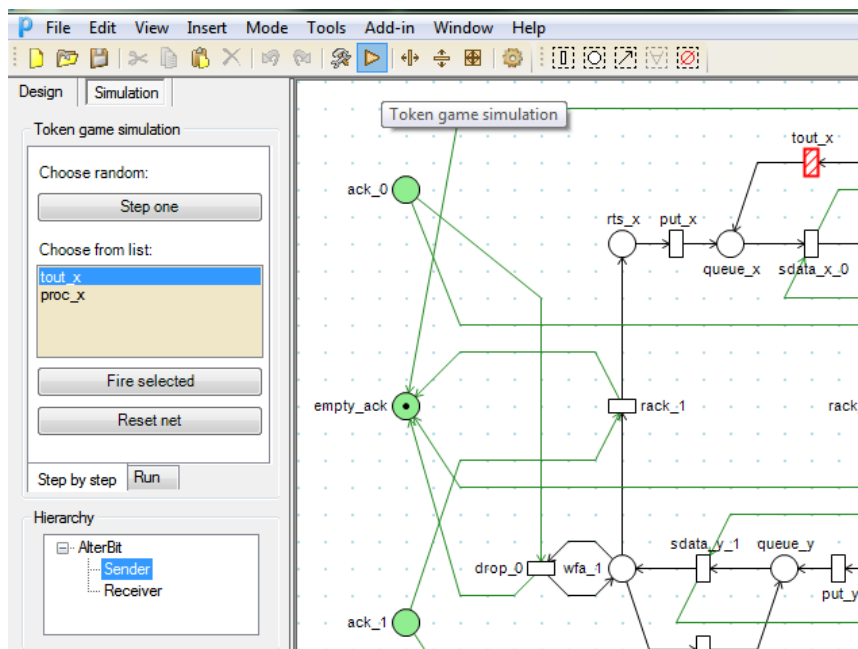
- Dinamikus tulajdonságok az elérhetőségi gráf alapján
 - Kezdőállapot függőek (nem általános érvényűek)
 - Jellegzetes dinamikus tulajdonságok (ld. később):
Elérhetőség, fedhetőség, élőség, holtpontmentesség, korlátosság, fairség, megfordíthatóság
 - Tulajdonságmegtartó redukációs technikák segítenek az analízisben
- Strukturális tulajdonságok a háló struktúrája alapján
 - Kezdőállapottól függetlenek: minden lehetséges működésre
 - Jellegzetes strukturális tulajdonságok (ld. később):
Strukturális élőség, strukturális korlátosság, vezérelhetőség, konzervativitás, ismételhetőség, konzisztencia
 - Invariánsok: T-invariánsok (tranzíciók tüzeléseire), P-invariánsok (helyek jelöléseire)

Petri-háló modellek szimulációja

Petri-hálók szimulációja

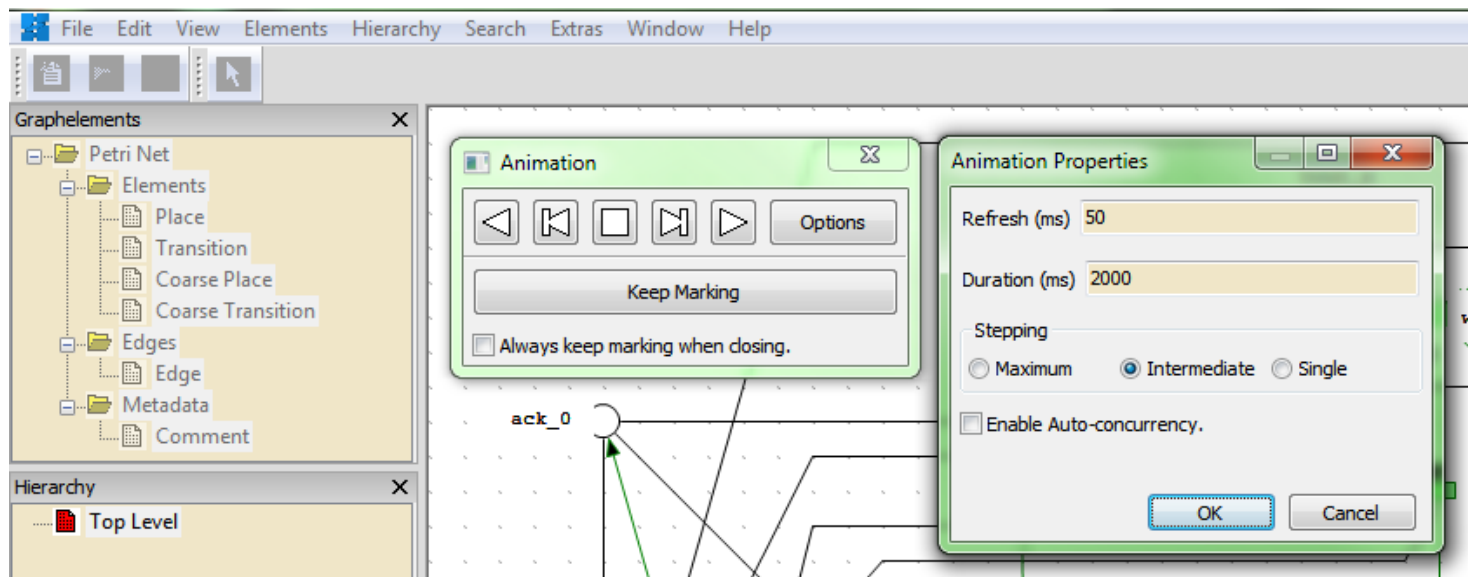
- A rendszer lehetséges trajektóriáinak vizsgálata
 - Állapot: tokeneloszlás (jelölés)
 - Állapotváltás (esemény): tranzíció tüzelése
 - Trajektóriák az állapottérben: Bejárható állapotsorozatok tüzelési szekvenciák hatására
- Petri-háló nemdeterminisztikus lehet
 - Interaktív szimuláció (animation, token game): Felhasználói választással
 - Automatikus szimuláció (large scale simulation) Automatikus választás (ál-)véletlen generálás alapján

Interaktív szimuláció (token game)



- A modell interaktív ellenőrzése
 - Engedélyezett átmenetek jelölve, kattintva tüzel
 - Előállítja az új tokeneloszlást
 - Manuális vagy automatikus (véletlen) választás (pl. PetriDotNet)

PetriDotNet ↑:



Snoopy:

Automatikus szimuláció

- Lépések (tüzelések) számának beállítása
- Statisztika gyűjtése
 - Tüzelések száma és aránya
 - Helyek átlagos tokenszáma

PetriDotNet:

Large scale statistics

Settings

Number of firings: 10 000

Run from current state Keep ending state

Run from initial state Show hierarchical names

Transitions

Transition	Firings	Percentage
put_x	137	1,37 %
drop_1	6	0,06 %
lose_0	413	4,13 %
put_y	137	1,37 %
rack_0	137	1,37 %

Places

Place	Avg token	Avg token in time
ack_0	0,337026793348499	---
empty_ack	0,585617646523382	---
ack_1	0,236643479018611	---
data_x	0,450335110909001	---
empty_data	0,506631204575518	---

Progress: 0,432 s

Run

OK

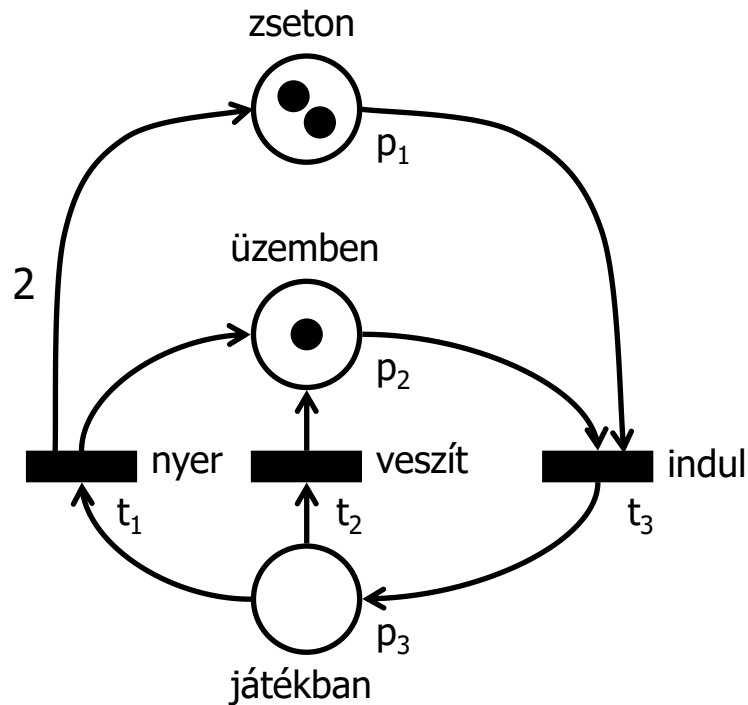
Ismétlés: Egy tranzíció tüzelése

Ha t tüzel M állapotban

- Új állapot: $M' = M + \mathbf{W}^T \cdot \mathbf{e}_t$
 - ahol \mathbf{e}_t a t tranzíciónak megfelelő egységvektor
- Itt \mathbf{W} a súlyozott szomszédossági mátrix
 - Dimenziója: $\tau \times \pi = |T| \times |P|$ ← sorok \times oszlopok
 - Elem: $w(t, p)$ ← t tüzelése hogyan módosítja p jelölését; bemenő és kimenő élsúlyok alapján számítható:

$$w(t, p) = \begin{cases} w^+(t, p) - w^-(p, t) & \text{ha } (t, p) \in E \text{ vagy } (p, t) \in E \\ 0 & \text{ha } (t, p) \notin E \text{ és } (p, t) \notin E \end{cases}$$

Ismétlés: Egy tranzíció tüzelése (példa)



$$\mathbf{W}^T = \begin{matrix} & t_1 & t_2 & t_3 \\ p_1 & 2 & 0 & -1 \\ p_2 & 1 & 1 & -1 \\ p_3 & -1 & -1 & 1 \end{matrix}$$

Állapotváltozás:

$$\mathbf{M}' = \mathbf{M} + \mathbf{W}^T \cdot \mathbf{e}_t$$

t_3 tranzíció tüzelése a fenti kezdőállapotból:

$$\mathbf{M}' = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 2 & 0 & -1 \\ 1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Egyszerű szimulációs algoritmus

```
<Tüzelhető tranzíciók listájának összeállítása>  
while (<van tüzelhető tranzíció>)  
    <Tüzelő tranzíció (nemdeterminisztikus) kiválasztása>  
    <Tüzelés végrehajtása>  
    <Tüzelhető tranzíciók listájának összeállítása>  
end while
```

Tüzelhető tranzíciók listájának összeállítása:

```
function collect_fireable_transitions(M)  
     $L_{\text{fireable}} \leftarrow \emptyset$   
    for all  $t \in T$  do  
        if enabled( $t, M$ ) then  $L_{\text{fireable}} \leftarrow L_{\text{fireable}} \cup \{t\}$   
    return  $L_{\text{fireable}}$   
end function
```

Szimulációs algoritmus

// Inicializálás

$M \leftarrow M_0$

$L_{\text{fireable}} \leftarrow \text{collect_fireable_transitions}(M)$

// Tüzelési ciklus

while $L_{\text{fireable}} \neq \emptyset$ do

$t \leftarrow \text{rnd}(L_{\text{fireable}})$

$M' \leftarrow M + W^T \cdot \underline{e}_t$

$L_{\text{fireable}} \leftarrow \text{collect_fireable_transitions}(M')$

$M \leftarrow M'$

end while

Ötlet a hatékonyság növelésére

- A tüzelhetőség változása szempontjából elegendő az utoljára tüzelt tranzíció környezetét vizsgálni
- Egy t tranzíció tüzelése során letiltódhat:
 - Olyan t' tranzíció, amely elől t tüzelése „elveszi a tokent”, azaz bemenete kapcsolódik t bemenetéhez
 - Letiltódhatnak t tüzelése által: $T' = \{(\bullet t)\bullet\}$
- Egy t tranzíció tüzelése során engedélyeződhet:
 - Olyan t'' tranzíció, melynek t tüzelése „odatesz tokent”, azaz bemenete kapcsolódik t kimenetéhez
 - Engedélyeződhetnek t tüzelése által : $T'' = \{(t\bullet)\bullet\}$

Hatékony algoritmus: tüzelési ciklus

```
while  $L_{\text{fireable}} \neq \emptyset$  do
  // Tüzelés
   $t \leftarrow \text{rnd}(L_{\text{fireable}})$ 
   $M' \leftarrow M + W^T \cdot \underline{e}_t$ 
  // Letiltott tranzíciók eltávolítása
  for all  $t' \in \{(\bullet t)\bullet\}$  do
    if not(enabled( $t'$ ,  $M'$ )) then  $L_{\text{fireable}} \leftarrow L_{\text{fireable}} \setminus \{t'\}$ 
  // Engedélyezett tranzíciók bevonása
  for all  $t'' \in \{(t\bullet)\bullet\}$  do
    if enabled( $t''$ ,  $M'$ ) then  $L_{\text{fireable}} \leftarrow L_{\text{fireable}} \cup \{t''\}$ 
   $M \leftarrow M'$ 
end while
```

Prioritás

- Tüzelési szabály kiegészül: t akkor és csak akkor tüzelhet, ha
 - Engedélyezett és
 - Nincs a $\pi(t)$ prioritásánál nagyobb prioritású engedélyezett tranzíció
- Következmény:
 - L_{fireable} nem halmaz, hanem halmazok vektora: $L_{\text{fireable}}[\pi]$ rendezve a $\pi \in \Pi$ prioritási szintek szerint
 - Tüzeléskor a legmagasabb prioritású nem üres $L_{\text{fireable}}[\pi]$ halmazból választunk nondeterminisztikusan

Prioritásos szimulációs algoritmus: inicializálás

// Inicializálás

$M \leftarrow M_0$

for all $\pi \in \Pi$ do

$L_{\text{fireable}}[\pi] \leftarrow \emptyset$

// Tüzelhető tranzíciók kezdeti halmaza

for all $t \in T$ with $\pi(t)$ do

if $\text{enabled}(t, M_0)$ then $L_{\text{fireable}}[\pi(t)] \leftarrow L_{\text{fireable}}[\pi(t)] \cup \{t\}$

Prioritásos szimulációs algoritmus: tüzelési ciklus

```
while  $\bigcup_{\pi \in \Pi} L_{\text{fireable}}[\pi] \neq \emptyset$  do // Összes prioritási szintre
```

```
  for  $\pi = \pi_{\text{max}}$  to  $\pi_{\text{min}}$  step -1 do // Tüzelés, prioritási szintek szerint
```

```
    if  $L_{\text{fireable}}[\pi] \neq \emptyset$  then
```

```
       $t \leftarrow \text{rnd}(L_{\text{fireable}}[\pi])$ 
```

```
       $M' \leftarrow M + W^T \cdot \underline{e}_t$ 
```

```
      exit for
```

```
    end if
```

```
  for all  $\pi \in \Pi$  do // Tüzelhető tranzíciók módosulása
```

```
    for all  $t' \in \{(\bullet t)\bullet\}$  do
```

```
      if not(enabled( $t'$ ,  $M'$ )) then  $L_{\text{fireable}}[\pi(t')] \leftarrow L_{\text{fireable}}[\pi(t')] \setminus \{t'\}$ 
```

```
    for all  $t'' \in \{(t\bullet)\bullet\}$  do
```

```
      if enabled( $t''$ ,  $M'$ ) then  $L_{\text{fireable}}[\pi(t'')] \leftarrow L_{\text{fireable}}[\pi(t'')] \cup \{t''\}$ 
```

```
    end for
```

```
     $M \leftarrow M'$ 
```

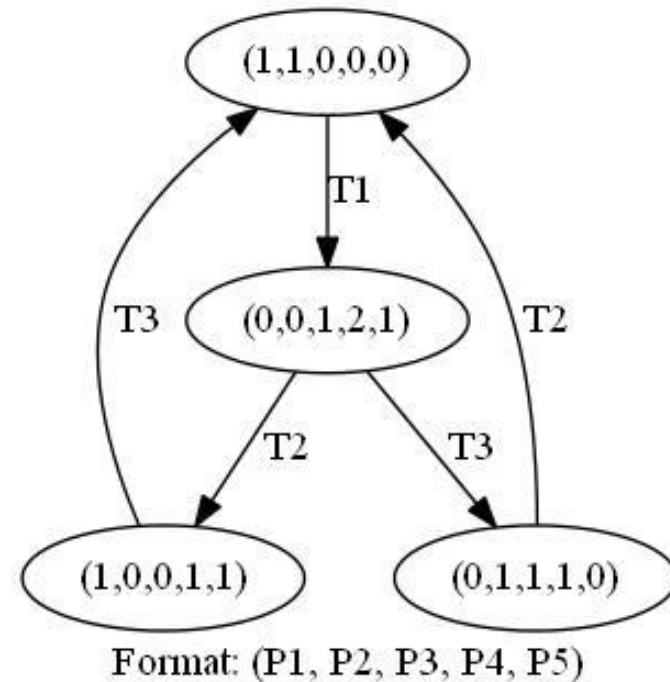
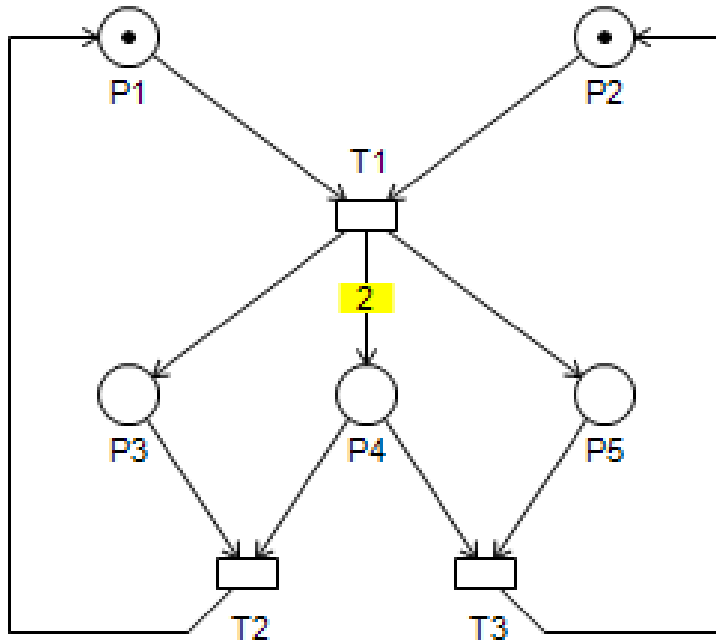
```
end while
```

Elérhetőségi analízis

Elérhetőség az állapottérben

- Állapottér: Kezdőállapotfüggő viselkedés leírása
 - Állapot: Jelölés (tokeneloszlás, marking)
 - Állapotátmenet: Tranzíció tüzelése
 - Trajektória: (M_0, M_1, \dots, M_n) állapotsorozat egy σ tüzelési szekvencia hatására
- M_n állapot elérhető az M_0 kiinduló állapotból, ha
$$\boxed{\exists \vec{\sigma} : M_0 [\vec{\sigma} > M_n]}$$
- Elérhetőségi gráf: állapottér megjelenítése

Példa: Elérhetőségi gráf



Egyszerű Petri-háló és elérhetőségi gráfja
(a PetriDotNet eszközből)

Elérhetőségi analízis

- Az M_0 kiinduló állapotból az N Petri-hálóban
 - Elérhető állapotok

$$R(N, M_0) = \{ M \mid \exists \vec{\sigma} : M_0 [\vec{\sigma} > M] \}$$

Állapot alapú kérdésekre lehet ez alapján válaszolni

- Végrehajtható tüzelési sorozatok

$$L(N, M_0) = \{ \vec{\sigma} \mid \exists M : M_0 [\vec{\sigma} > M] \}$$

Állapotátmenet (esemény, tüzelés) alapú kérdésekre lehet válaszolni

Elérhetőségi problémák

- Petri-hálók elérhetőségi problémája:

- M_n állapot elérhető-e valamilyen M_0 kiinduló állapotból

$$\boxed{M_n \stackrel{?}{\in} R(N, M_0)}$$

- Az elérhetőségi probléma eldönthető, de exponenciális (hely) komplexitású általános esetben

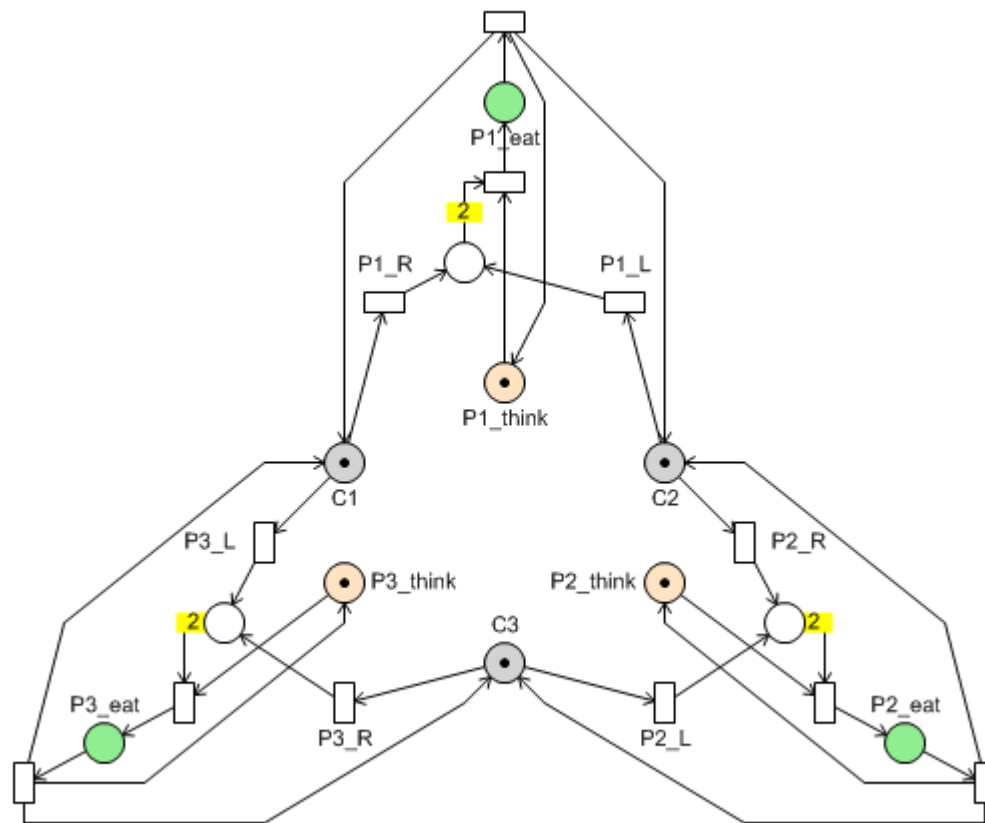
- Rész-tokeneloszlási probléma:

- Helyek egy $P' \subset P$ részhalmazára korlátozva a kérdést, elérhető-e M_n állapot az adott P' -beli helyekre megadott tokeneloszlással

$$\boxed{\stackrel{?}{\exists} M \in R(N, M_0) : \forall p \in P' : M(p) = M_n(p)}$$

Elérhetőség vizsgálata modellellenőrzéssel

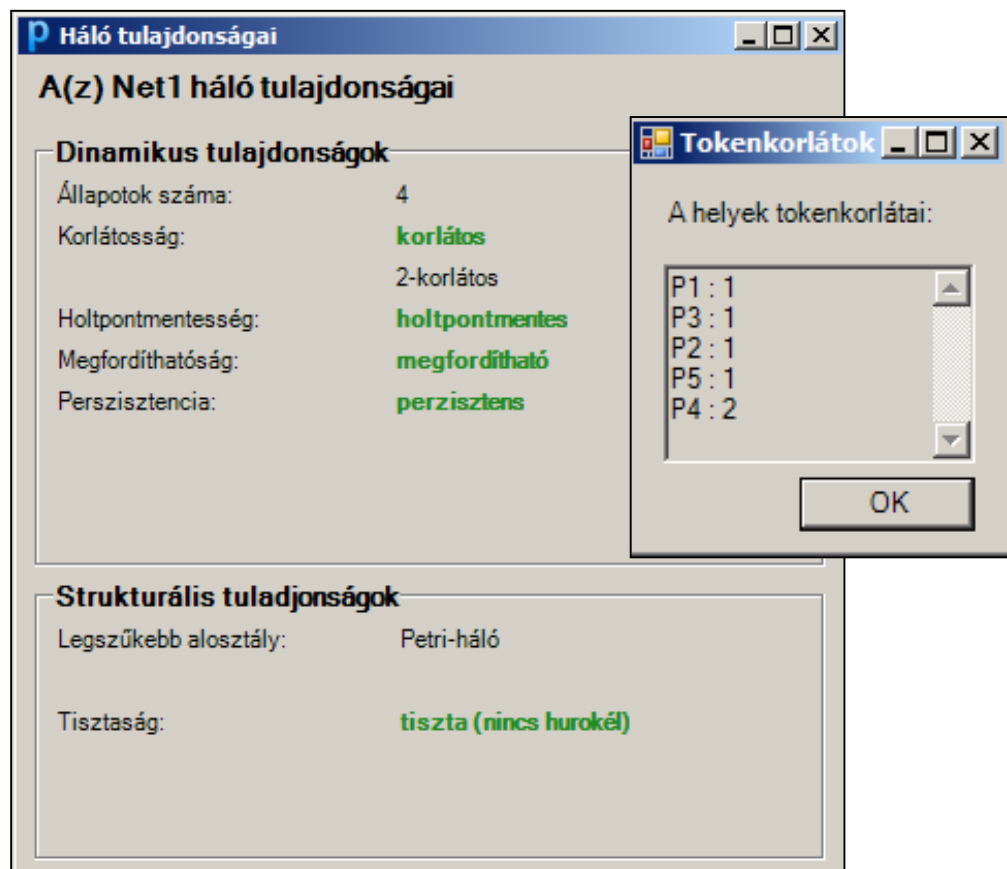
- Étkező filozófusok
- Egy-egy filozófusra:
 - Képes enni legalább egyszer?
 - Mindenképpen fog enni legalább egyszer?
 - Mindig biztos, hogy előbb-utóbb enni fog?
- A teljes modell
 - Holtpontmentes?



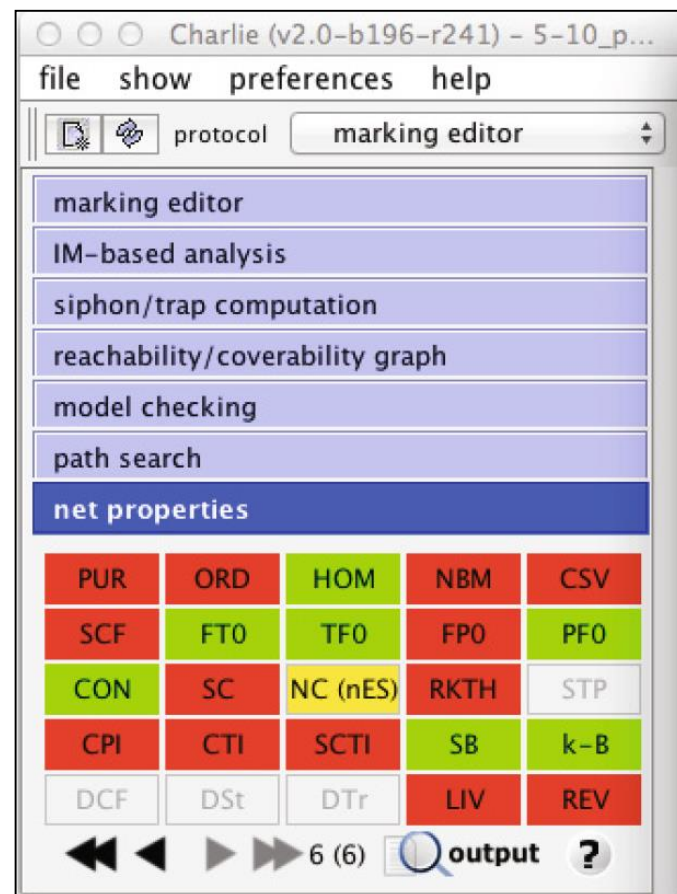
Petri-hálóak dinamikus (viselkedési) tulajdonságai

Tulajdonságok Petri-háló eszközökben

PetriDotNet:



Snoopy + Charlie:



Dinamikus tulajdonságok

- Elérhetőséggel kapcsolatos tulajdonságok
 - **Függenek** a kiinduló állapottól (kezdő jelöléstől)
 - Lesznek olyan tulajdonságok is, amik kiinduló állapottól függetlenek: ezek az ún. strukturális tulajdonságok
 - Tipikusan elérhetőségi analízissel határozhatók meg
- Dinamikus tulajdonságok (áttekintés):
 1. Korlátosság
 2. Élőség
 - Holtpontmentesség
 3. Megfordíthatóság
 4. Visszatérő állapot
 5. Fedhetőség
 6. Perzisztencia
 7. Fair tulajdonság
 - Korlátozott fairség
 - Globális fairség

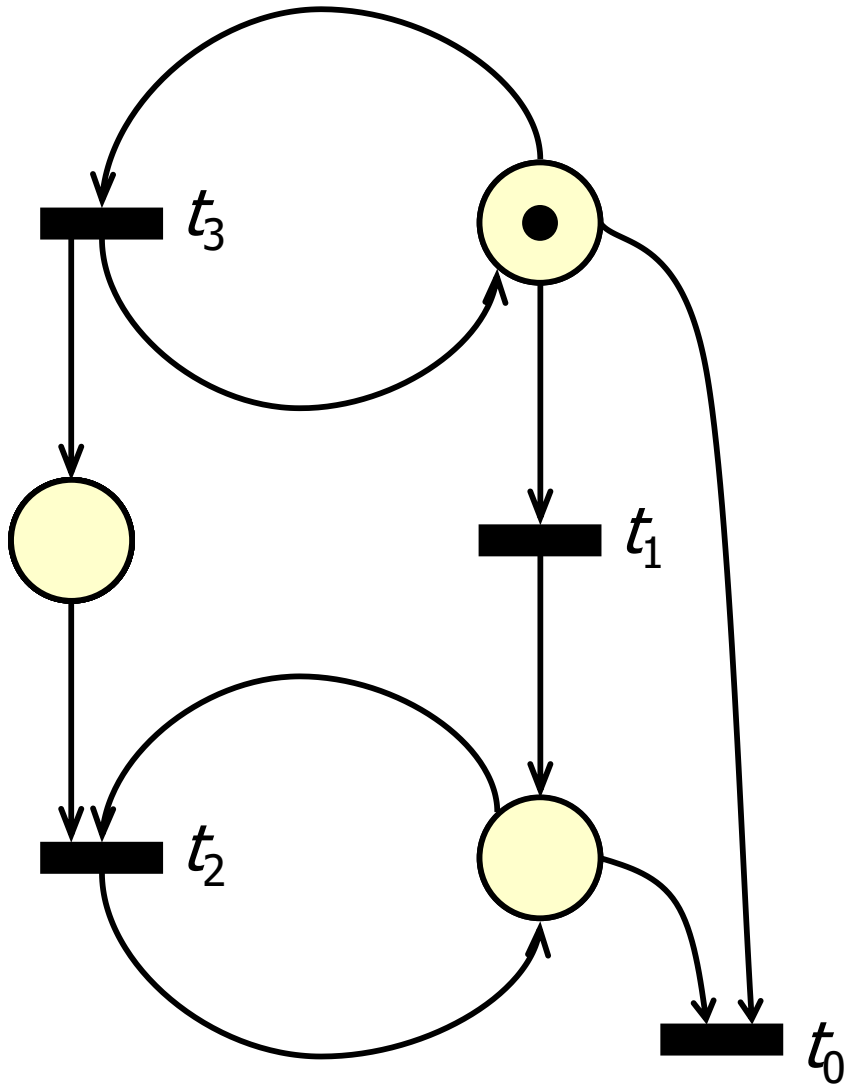
1. Korlátosság

- **k-korlátosság (korlátosság)**
 - Bármely állapotban minden helyre: helyenként maximum k token lehet
 - Biztos Petri-háló: korlátosság speciális esete: $k = 1$
 - „Végesség” kifejezése
 - Korlátosság \Leftrightarrow véges állapottér
- **Megválaszolható gyakorlati kérdések**
 - A rendszerben felgyűlnek-e a feladatok?
 - Megvalósul-e az üzenetek rendszeres feldolgozása?

2. Élőség tranzíciókra

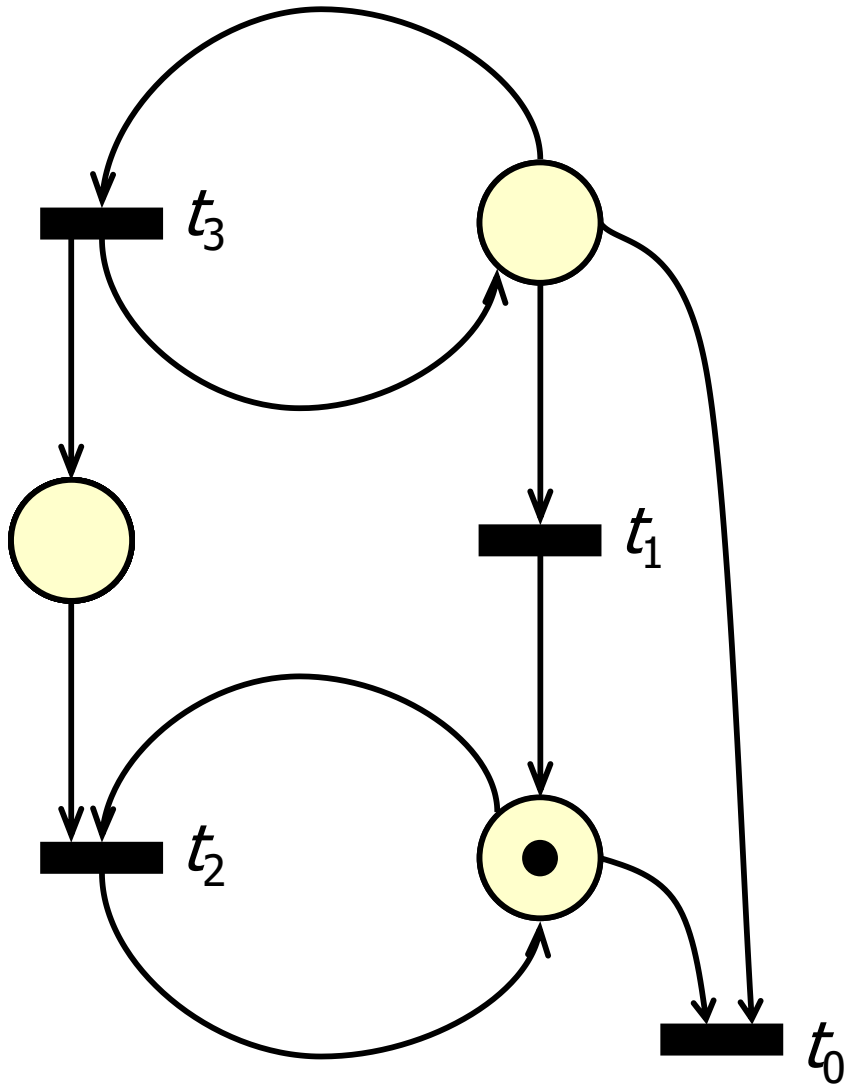
- Nem élő tranzíció:
 - L0-élő (halott): t sohasem tüzelhet
- Gyenge élő tulajdonságok: Tüzelési lehetőségek valamely trajektória mentén
 - L1-élő: t legalább egyszer tüzelhető
 - L2-élő: bármely $k > 1$ egészre t legalább k -szor tüzelhető
 - L3-élő: t végtelen sokszor tüzelhető
- Erősebb élő tulajdonság: Tüzelési lehetőség bármely elérhető állapotból
 - L4-élő: t legalább egyszer tüzelhető bármely állapotból valamely trajektória mentén

Élő tulajdonság: példa



- t_0 tranzíció: L0-élő (halott)
- t_1 tranzíció: L1-élő
- t_2 tranzíció: L2-élő
- t_3 tranzíció: L3-élő

Élő tulajdonság: példa



- t_0 tranzíció: L0-élő (halott)
- t_1 tranzíció: L1-élő
- t_2 tranzíció: L2-élő
- t_3 tranzíció: L3-élő



Alkalmazási példák

- Munkafolyamat esetén, ahol egy tranzíció egy tevékenységet modellez
 - L0-élő (halott): tevékenység sohasem kerülhet sorra (nem létezik tüzelése)
 - L1-élő: tevékenység legalább egyszer végrehajtható
 - L2-élő: tevékenység véges sokszor végrehajtható
 - L3-élő: tevékenység végtelen sokszor végrehajtható (pl. része egy ciklikus tevékenységsorozatnak)
 - L4 élő: tevékenység bármely állapotból legalább egyszer végrehajtható lesz (nincs olyan állapot, hogy abból nem lesz végrehajtható)

Az élőség Petri-hálókra

- Egy Petri-háló **Lx-élő** (Lx: L1, L2, L3, L4)
 - Ha minden tranzíciója **Lx-élő**
 - L4-től L1-ig az élő tulajdonságok **tartalmazzák egymást**
- Egy Petri-háló **holtpontmentes**
 - Ha minden állapotban legalább egy tranzíció tüzeltető
- Egy Petri-háló **élő**
 - Ha **L4-élő**, azaz minden tranzíciója **L4-élő**
 - L4-élő: L1-élő (azaz legalább egyszer tüzeltető valamely trajektória mentén) bármely elérhető állapotból
 - **Bejárasi úttól függetlenül garantáltan holtpontmentes**
 - **Élőség** \Rightarrow **Holtpontmentesség** (de fordítva nem)

3. Megfordíthatóság

- Megfordíthatóság

- Az M_0 kezdőállapot elérhető bármely őt követő állapotból

$$\forall M \in R(N, M_0) : M_0 \in R(N, M)$$

- Gyakorlati példák a megfordíthatóságra:

- Biztonságos kezdőállapot mindenholnan elérhető
- „Reset” hatására mindig a kezdőállapotba vihető rendszer
- Ciklikus működésű hálózat, a kezdőállapoton keresztül

4. Visszatérő állapot

- Visszatérő állapot

- Az állapottér adott állapota elérhető bármely őt követő állapotból

$$\exists M_n \in R(N, M_0) : \forall M \in R(N, M_n) : M_n \in R(N, M)$$

- Gyakorlati példák a visszatérő állapotokra:

- Inicializáló szekvencia után ciklikus működés adott állapotokon keresztül (ezek a visszatérő állapotok)
- Inicializálás után bárhonnán elérhető biztonságos állapot (ez a visszatérő állapot)

5. Fedhetőség

- Fedhetőség

- Létrejön-e korábbi állapotot magában foglaló állapot?

- M' állapot fedi az M állapotot: $\forall p \in P : m'(p) \geq m(p)$

- Fordított megfogalmazás: M állapot fedhető M' állapottal

- Gyenge fedhetőség esetén az azonos állapot is fed

- Erős fedhetőség: $\exists p \in P : m''(p) > m(p)$

- Kapcsolat az élőséggel

- Ha μ a t tranzíciót engedélyező minimális tokeneloszlás

- t akkor és csak akkor nem L1-élő, ha μ nem fedhető le

- Fordítva: μ lefedhetősége garantálja t L1-élő voltát (tüzelhet)

6. Perzisztencia

- Perzisztencia tranzíciókra
 - Egy tranzíció perzisztens, ha engedélyezetté válva engedélyezve is marad tüzelésig
 - Azaz nincs olyan engedélyezett tranzíció, amelynek tüzelése letiltja a tranzíció engedélyezettségét („elveszi előle a tokent”)
- Perzisztencia Petri-hálókra
 - Egy (P, T, M_0) Petri-háló perzisztens, ha minden tranzíciója az összes lehetséges tüzelési szekvenciában perzisztens (azaz nincs másik olyan tranzíció, aminek tüzelése letiltaná)
- Gyakorlati példák perzisztencia alkalmazására:
 - Párhuzamos működések befolyásolják-e egymást?
 - Rendszerbeli funkcionális dekompozíció megmarad-e?

7. Fair tulajdonság: korlátozott fairség

- Kétféle fairség definíció
 - Korlátozott fairség (bounded fairness)
 - Globális fairség (korlátlan, unbounded fairness)
- Korlátozott fairség
 - Egy tüzelési szekvencia korlátozottan fair (B-fair)
 - Ha bármely tranzíció maximum korlátos sokszor tüzelhet anélkül, hogy egy másik engedélyezett tranzíció tüzelne
 - „Nem veheti el folyamatosan a tüzelési lehetőséget más elől”
 - Egy Petri-háló korlátozottan fair (B-fair)
 - Ha az összes lehetséges tüzelési szekvenciája korlátozottan fair

Fair tulajdonság: globális fairség

- **Globális fairség**

- Egy tüzelési szekvencia globálisan fair, ha

- véges, vagy

- az összes tranzíció végtelen sokszor tüzelhet benne

- Egy Petri-háló globálisan fair

- Ha a háló összes lehetséges tüzelési szekvenciája globálisan fair

- **Gyakorlati példák a fairség alkalmazására:**

- Párhuzamos folyamatok nem tartják-e fel egymást?

- Valamennyi tevékenység végbemegy-e (előbb-utóbb)?

- Kérés kiszolgálása előbb-utóbb megtörténik-e?

Dinamikus tulajdonságok (összefoglalás)

- Korlátosság
- Holtpontmentesség
- Élő tulajdonság
 - L0 élő (halott)
 - L1 élő (1-szer tüzelhető)
 - L2 élő (k-szor tüzelhető)
 - L3 élő (∞ -szer tüzelhető)
 - L4 élő (\forall állapotból L1)
- Megfordíthatóság
- Visszatérő állapot
- Fedhetőség
 - Gyenge fedhetőség
 - Erős fedhetőség
- Perzisztencia
- Fair tulajdonság
 - Korlátozott fairség
 - Globális fairség

Állapottér reprezentációk:
az elérhetőségi és fedési gráf

Állapottér reprezentációk: Elérhetőségi gráf

- Elérhetőségi gráf
 - M_0 kezdőállapotból induló állapotgráf
 - Csomópontok: állapotok; címkézés: tokeneloszlások
 - Állapotátmenetek: irányított élek; címkézés: tüzelések
 - Egy csomópont esetén legfeljebb annyi rákövetkező csomópont (kimenő él), ahány engedélyezett tranzíció
 - Kevesebb, ha prioritásos a Petri-háló
 - Csomópont, amiből nem indul ki él: **holtpont**
 - Nem korlátos a Petri-háló → végtelen sok állapot
 - Korlátosság \Leftrightarrow véges állapottér
 - Vizsgálat: Szélességi típusú bejárás az állapotból a tüzelések mentén
 - Mélységi bejárás nem korlátos állapottérben rossz ötlet

Állapottér reprezentációk: Fedési gráf

- Végtelen állapotgráf: token „túlszaporodás”
 - Hol, „milyen módon” lesz végtelen?
 - Milyen analízisre ad lehetőséget?
- Fedési gráf: végtelen állapottér esetére is
 - Hasonló felépítés: M_0 kezdőállapot, élek: tüzelések
 - Trajektória: $M_0, \dots, M'', \dots, M'$
 - Ha itt M'' egy fedett állapot, erős fedettséggel:
$$\exists p \in P : m'(p) > m''(p)$$
akkor az erősen fedett helyekre speciális szimbólum:
 ω a végtelenség kifejezője

Fedési fa generáló algoritmus

Építkezés gráfcsomópontokkal:

$L_{\text{vizsgálandó}} \leftarrow \{ M_0 \}$

MAIN: **if** $L_{\text{vizsgálandó}} \neq \emptyset$

A következő $M \in L_{\text{vizsgálandó}}$ gráfcsomópont kivétele

if M a gyökértől idáig vezető úton már szerepelt

then M -et „régiből csomópontként” jelöljük

goto MAIN // ciklus

if M -ben nincs engedélyezett tranzíció

then M -et „végcsomópontként” jelöljük

goto MAIN // ciklus

(folytatás a következő lapon)

Fedési fa generáló algoritmus (folytatás)

else // (van M -ben engedélyezett tranzíció)

for all t engedélyezett tranzícióra:

Az M' rákövetkező csomópont meghatározása: $M[\vec{e}_t > M'$

if létezik az M_0 -tól M -ig vezető úton olyan M'' , amelyet M' fed

$$M' \neq M'' \wedge \forall p \in P : m'(p) \geq m''(p) \wedge \exists p \in P : m'(p) > m''(p)$$

then M'' fedett csomópont:

az M' csomópontot jelölő tokeneloszlásban

az erősen fedett helyek jelöléseit ω -val helyettesítjük

$$\forall p \in P : m'(p) > m''(p) \rightarrow m'(p) = \omega$$

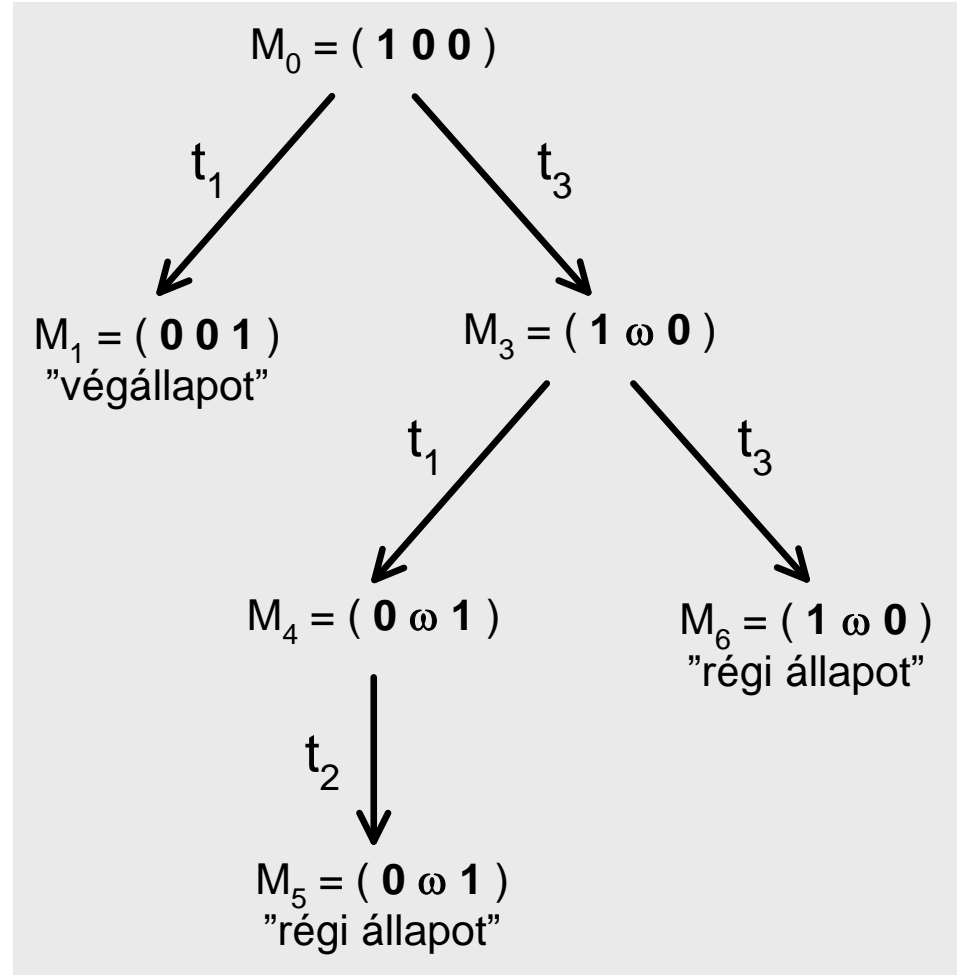
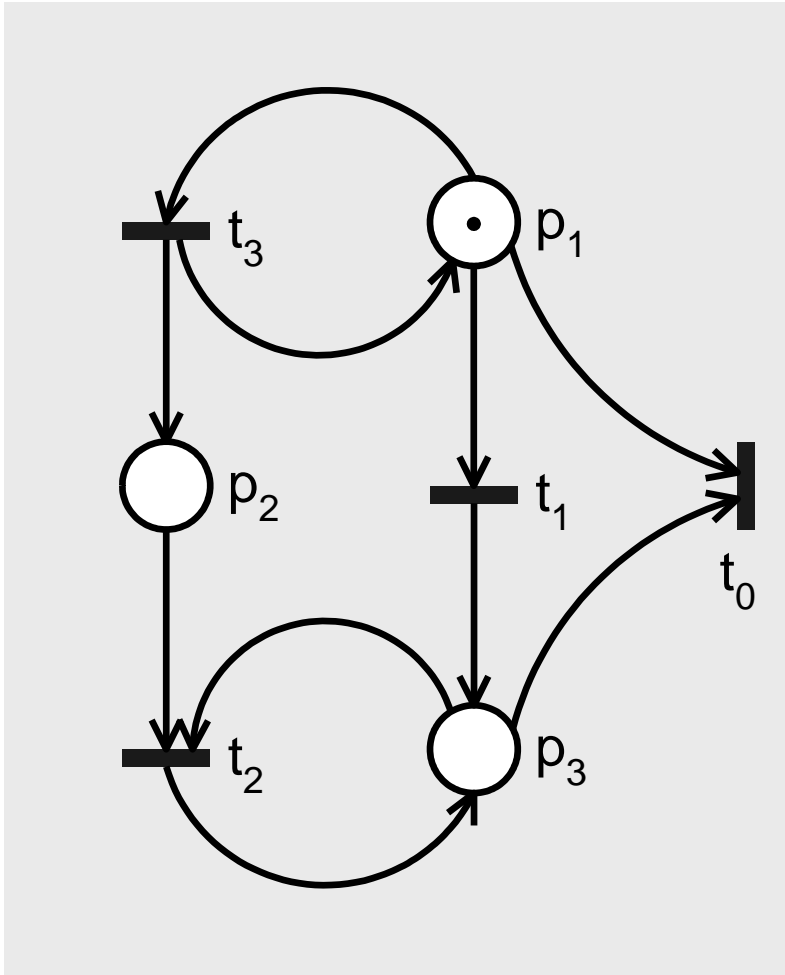
M' felvétele, vizsgálandók lesznek: $L_{\text{vizsgálandó}} \leftarrow L_{\text{vizsgálandó}} \cup M'$

M -ből M' -hez egy t -vel jelölt élet húzunk

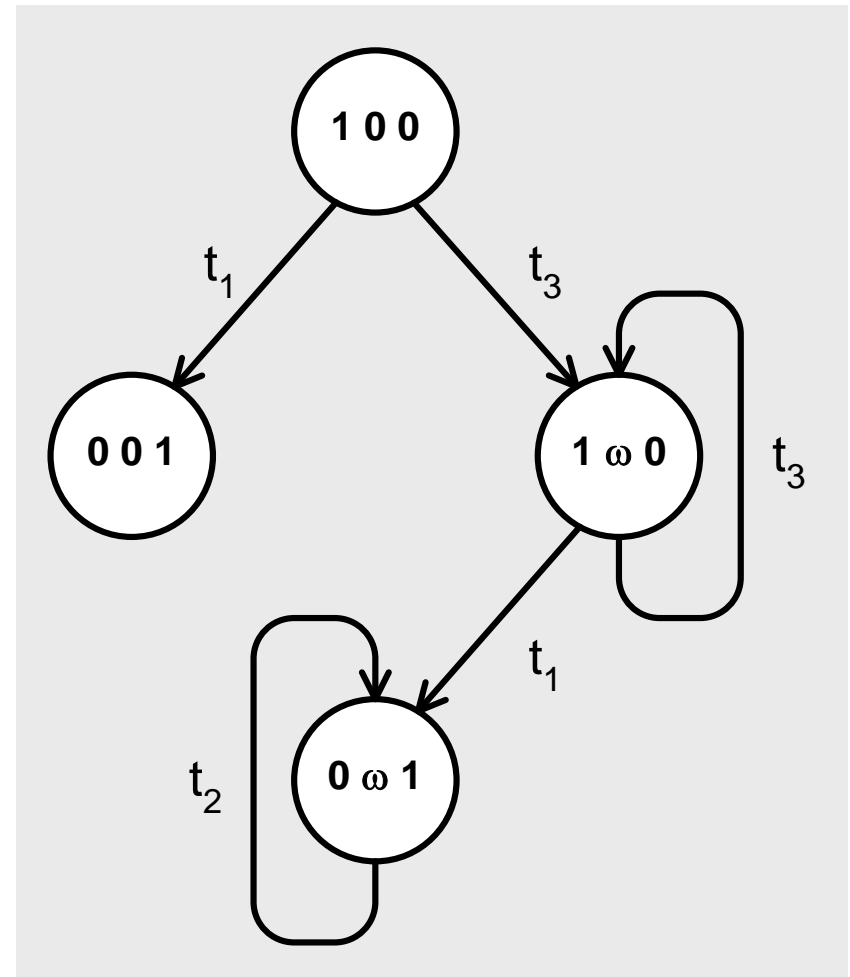
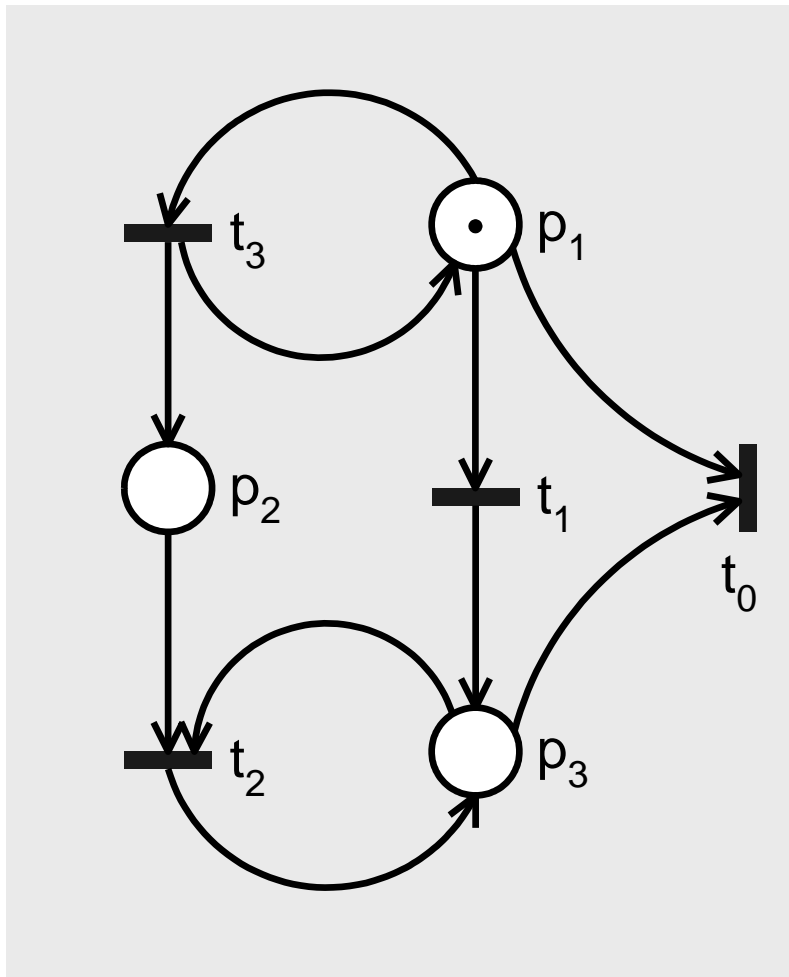
goto MAIN // ciklus

Fedési gráf: Azonos jelölést reprezentáló csomópontok összevonása

Egy példa és annak fedési fája



Egy példa és annak fedési gráfja



Petri-hálóok fedési fájának analízise

Közvetlenül is leolvasható tulajdonságok:

- Petri-háló **korlátos** $\Leftrightarrow R(N, M_0)$ elérhetőségi gráfja **véges**
 \Leftrightarrow Fedési fában ω **nem jelenik meg** címkeként
- Petri-háló **biztonságos** \Leftrightarrow Csak **0 és 1** jelenik meg csomópont címkeként a fedési fában
- Petri-háló egy tranzíciója **halott** \Leftrightarrow tranzícióhoz tartozó tüzelés **nem jelenik meg** élcímkeként a fedési fában

Tipikus dinamikus tulajdonságok analízis eszközökben

- Korlátosság: Boundedness
 - Élő (L4-élő) tulajdonság: Liveness
 - Holtpont: Deadlock
 - Visszatérő állapot: Home state
 - Elérhetőségi gráf: Reachability graph
 - Fedési gráf: Coverability graph
 - Tüzelési invariánsok: T-invariants
 - Hely invariánsok: P-invariants
- } Id. később