

Petri hálók analízise

dr. Bartha Tamás

dr. Majzik István

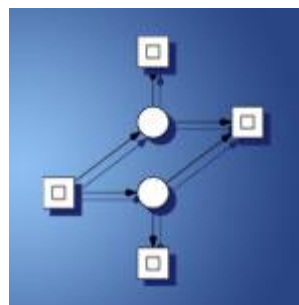
dr. Pataricza András

BME Méréstechnika és Információs Rendszerek Tanszék

Modellező és analízis eszközök: DNAnet, Snoopy, PetriDotNet



UNIVERSITY OF CAPE TOWN
Department of Computer Science



b-tu

Brandenburgische
Technische Universität
Cottbus

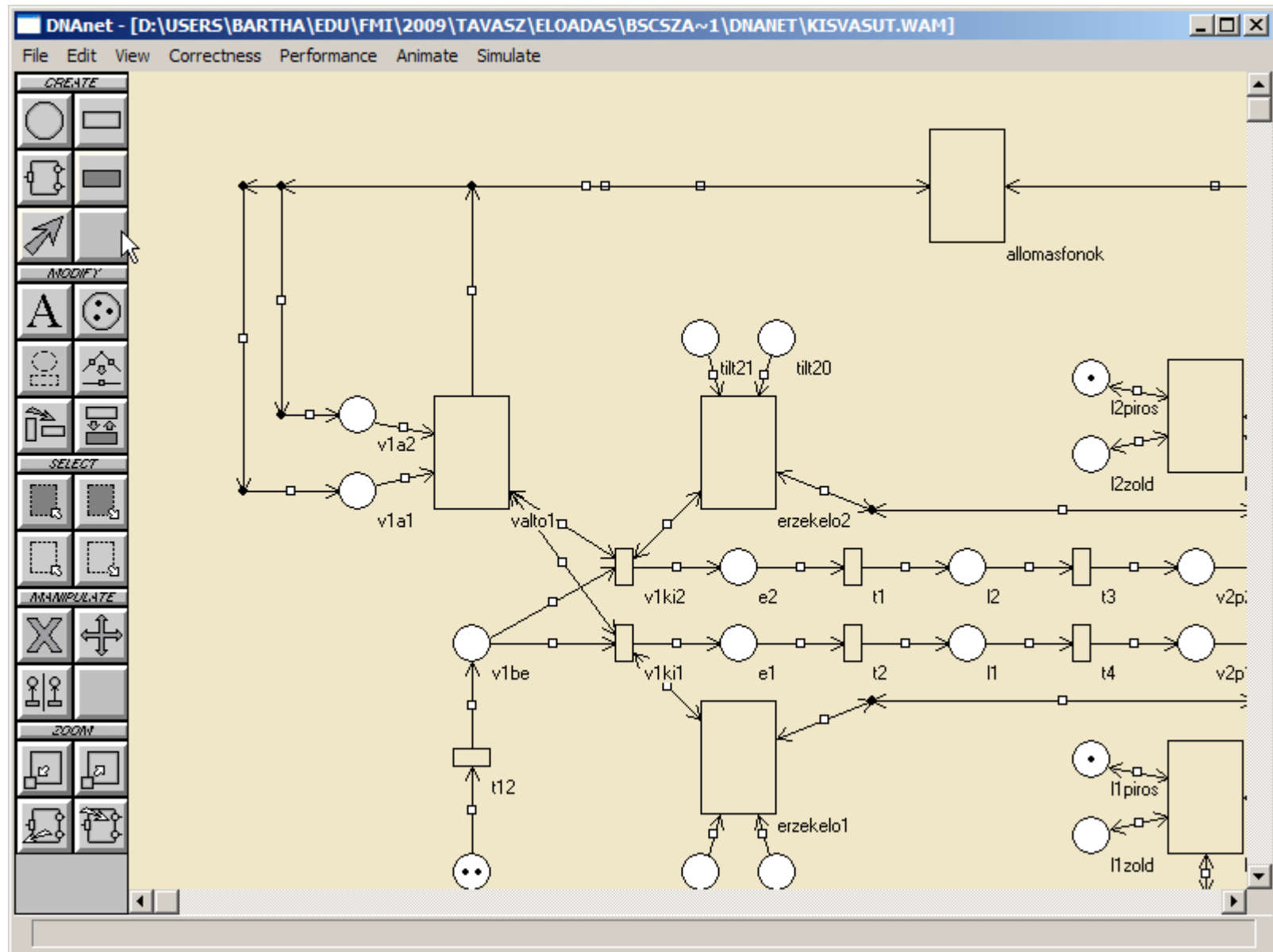
petridotnet
from BME-MIT



A DNAnet modellező program

- Képességei
 - Grafikus szerkesztő
 - Interaktív animáció (token game)
 - Nem interaktív szimuláció (teljesítmény analízishez)
 - Analízisek: egyes dinamikus és statikus tulajdonságok ellenőrzése
- Előnyei
 - Kicsi, kompakt, gyors, egyszerűen kezelhető
 - Méretéhez képest sokat tud
 - Ingyenes, szabad felhasználású
- Hátránya
 - Nem túl stabil ☹

A DNAnet modellező program képe



A DNAnet analízis képességei

Net is bounded.

Deadlock is not possible.

Net is live.

Net has home states.

Coverability graph generation statistics:

108 unique markings

1 strongly connected components

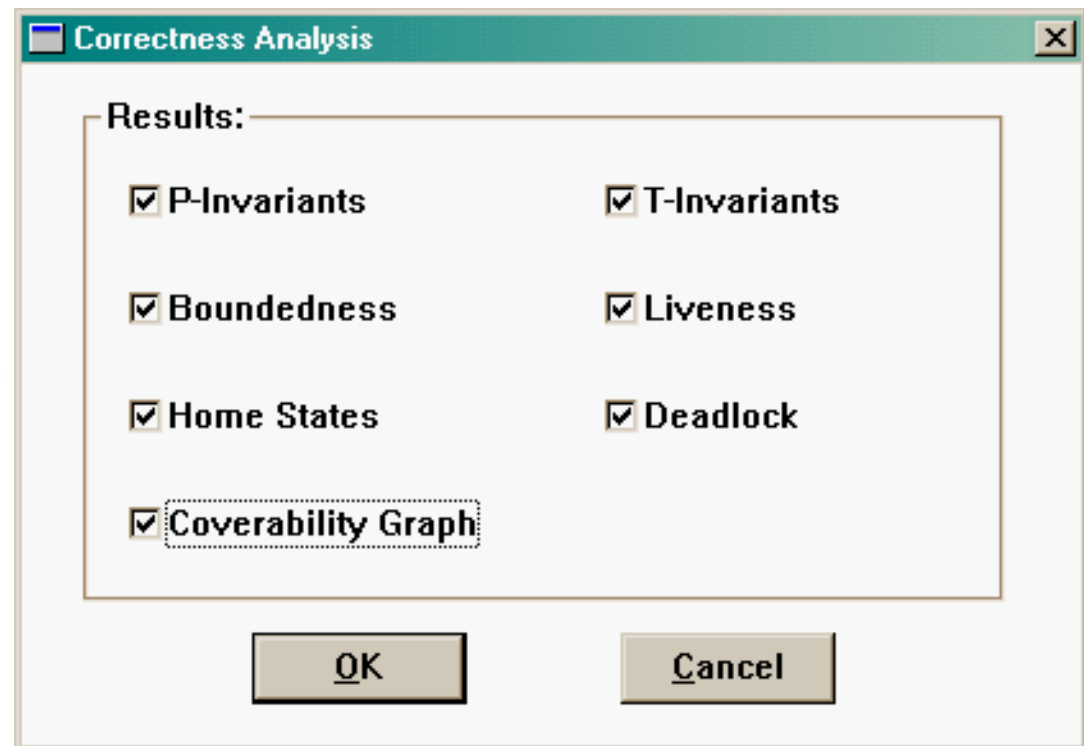
108 hash table entries used

1 was the longest hash list length

1 was the average hash list length

27 was the maximum stack height

3 was the maximum component stack height



A DNAnet invariáns keresése

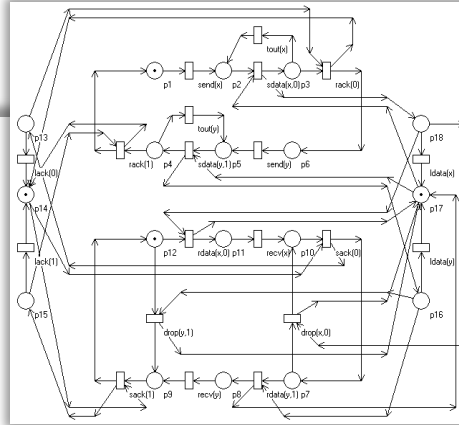
P-invariants:

1	0	0	0	(main.p1)
1	0	0	0	(main.p2)
1	0	0	0	(main.p3)
1	0	0	0	(main.p4)
1	0	0	0	(main.p5)
1	0	0	0	(main.p6)
0	1	0	0	(main.p7)
0	1	0	0	(main.p8)
0	1	0	0	(main.p9)
0	1	0	0	(main.p10)
0	1	0	0	(main.p11)
0	1	0	0	(main.p12)
0	0	1	0	(main.p13)
0	0	1	0	(main.p14)
0	0	1	0	(main.p15)
0	0	0	1	(main.p16)
0	0	0	1	(main.p17)
0	0	0	1	(main.p18)

ie.

$M(\text{main.p1}) + M(\text{main.p2}) + M(\text{main.p3}) + M(\text{main.p4}) + M(\text{main.p5}) + M(\text{main.p6})$
 $M(\text{main.p7}) + M(\text{main.p8}) + M(\text{main.p9}) + M(\text{main.p10}) + M(\text{main.p11}) + M(\text{main.p12})$
 $M(\text{main.p13}) + M(\text{main.p14}) + M(\text{main.p15})$
 $M(\text{main.p16}) + M(\text{main.p17}) + M(\text{main.p18})$

All places are covered by P-invariants.



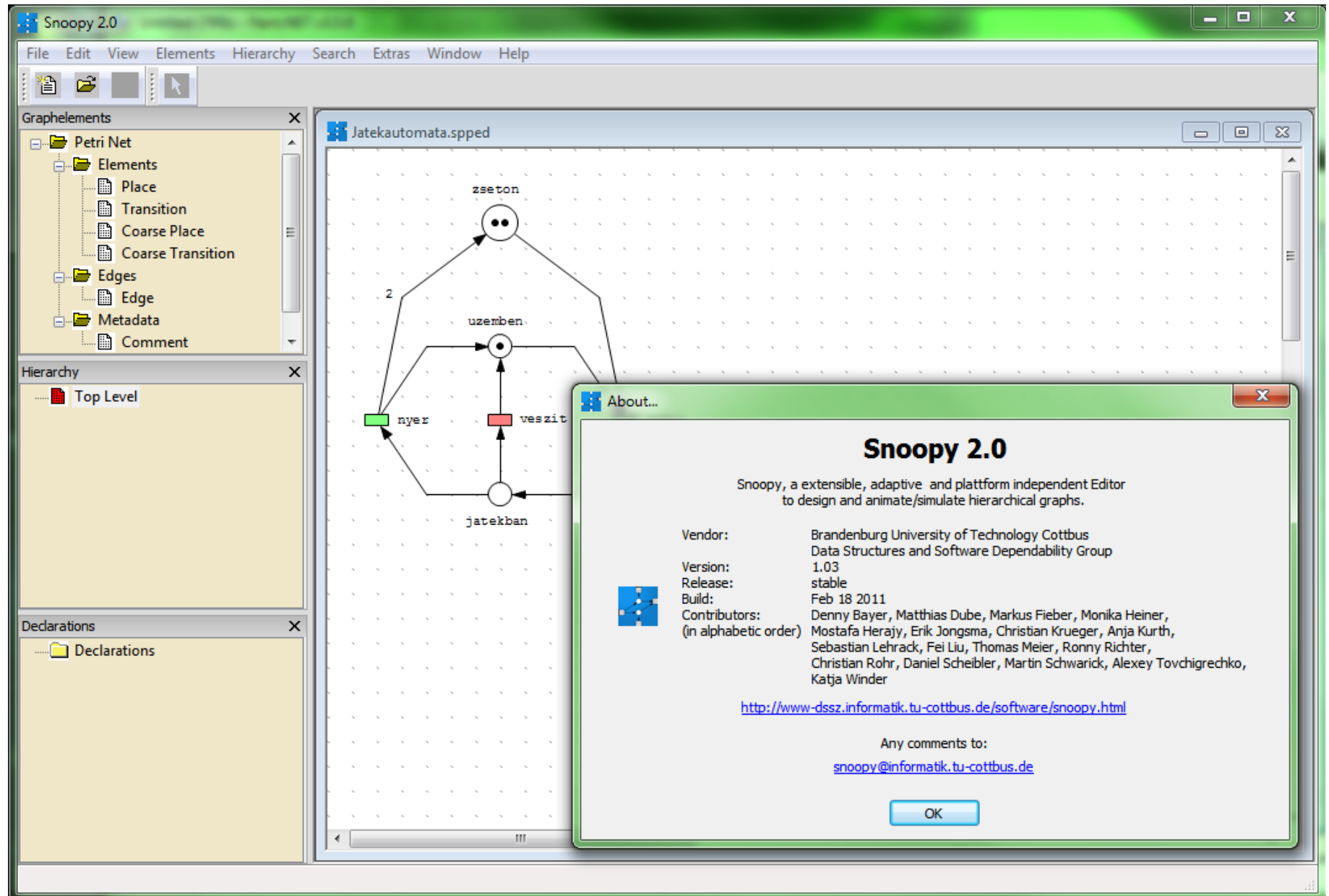
T invariants:

1	0	0	0	0	1	(main.send(x))
1	0	1	1	1	0	(main.sdata(x,0))
1	0	0	0	0	0	(main.rack(0))
1	0	0	0	0	0	(main.send(y))
1	1	0	0	1	1	(main.sdata(y,1))
1	0	0	0	0	0	(main.rack(1))
0	0	1	1	1	0	(main.tout(x))
0	1	0	0	1	1	(main.tout(y))
1	0	0	0	1	1	(main.sack(1))
1	0	0	0	1	0	(main.recv(y))
1	0	0	0	1	0	(main.rdata(y,1))
1	0	0	1	1	0	(main.sack(0))
1	0	0	0	1	0	(main.recv(x))
1	0	0	0	1	0	(main.rdata(x,0))
0	0	0	1	1	0	(main.lack(0))
0	0	0	0	1	1	(main.lack(1))
0	1	0	0	0	0	(main.ldata(y))
0	0	1	0	0	0	(main.ldata(x))
0	0	0	1	0	0	(main.drop(x,0))
0	0	0	0	0	1	(main.drop(y,1))

A Snoopy modellező program

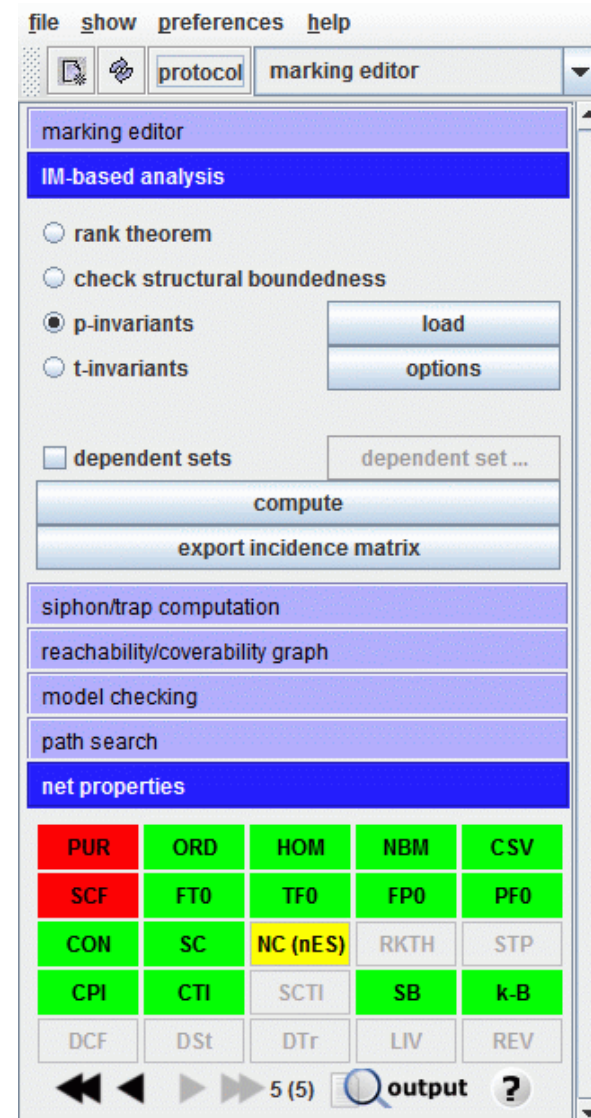
- Snoopy (Windows, Linux)
 - Grafikus szerkesztő + token game (animált)
 - Egyszerűen kezelhető
 - Kényelmi funkciók: copy / paste, undo / redo
 - **Kiterjesztések**: tiltó él, olvasó él, reset él, egyenlőség él
 - Számos háló típus, többek között **színezett háló** is
 - Támogatja **hierarchikus** Petri hálók készítését
 - Elemek színezése, méretezése, élsúlyok kijelzése
 - On-line help
 - Külső analízis eszköz: Charlie (Java)

A Snoopy modellező program képe



Analízis eszközök Snoopy-hoz

- Charlie (Java)
 - Dinamikus tulajdonságok, elérhetőség
 - Strukturális tulajdonságok, invariánsok
 - Explicit CTL és LTL modellellenőrző
- INA (Windows, Linux)
 - Szöveges felületű **parancssori** program
 - Token game (szöveges)
 - Invariáns analízis, elérhetőségi gráf generálás
 - Strukturális tulajdonságok ellenőrzése
 - Szimulációs képességei nincsenek



A PetriDotNet modellező program

- Képességei
 - Grafikus szerkesztő + token game + szimuláció
 - Egyszerűen kezelhető, sok kényelmi funkció
 - **Kiterjesztések**: tiltó él, időzítés, színezett háló
 - Támogatja **hierarchikus** Petri hálók készítését
 - Kiegészítő modulokkal bővíthető, pl. **analízis modulok**
 - Dinamikus tulajdonságok, **CTL modellellenőrző**
 - Elemek színezése, elforgatása, élsúlyok kijelzése
 - Szabványos PNML fájlformátum, van hozzá INA kimenet
- BME MIT fejlesztés: petridotnet.inf.mit.bme.hu

A Petri.NET modellező program képe

The screenshot displays the PetriDotNet software interface. The main window, titled "PetriDotNet - [Jatekautomata]", features a menu bar (File, Edit, View, Insert, Mode, Tools, Add-in, Window, Help) and a toolbar with various editing tools. The interface is divided into three main sections:

- Toolbox:** Located on the left, it contains buttons for "Select", "Place", "Transition", "Edge", "Token", and "Other elements".
- Properties:** Below the toolbox, it shows the "Identity" section with fields for "Id" (n1) and "Name" (Net 1).
- Diagram:** The central workspace displays a Petri net diagram with the following components:
 - Places:** "zseton" (top), "nyer" (bottom left, green), "jatekban" (bottom center), "uzemben" (top center), and "indul" (right).
 - Transitions:** "veszit" (center, red).
 - Edges:** Directed edges connect the places and transitions. A weight of "2" is shown on the edge from "nyer" to "uzemben".

An "About PetriDotNet" dialog box is open in the bottom right corner, providing the following information:

- petridotnet** from BME-MIT
- Version 1.3.4098.34586
- Petri Net Editor by Dániel Darvas, 2009-2011 at BME-MIT (BUTE DMIS)
- based on Petri.NET 1.0 by Bertalan Szilvási (advisor: Gábor Huszerl), 2008
- Advisors:** András Vörös (BME-MIT), dr. Tamás Bartha (BME-MIT)
- Contact us at <http://petridotnet.inf.mit.bme.hu/> or petridotnet@inf.mit.bme.hu.

At the bottom of the dialog box, there is a logo for "MŰEGYETEM 1782" and a row of four colored circles (green, blue, red, orange) containing the letters F, T, S, and RG respectively. Buttons for "Send error feedback" and "OK" are also present.

A PetriDotNet analízis képességei

A(z) AlterBit háló tulajdonságai

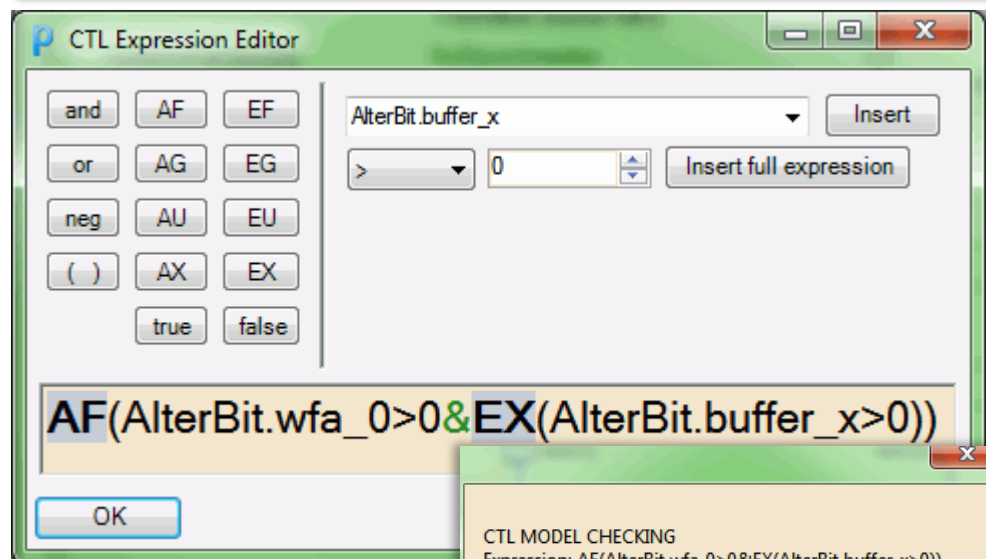
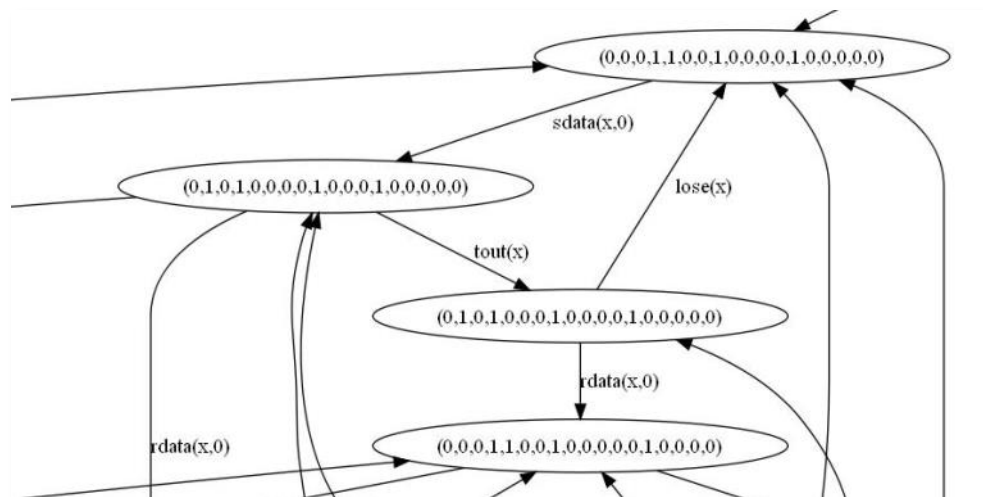
Dinamikus tulajdonságok

Állapotok száma: 108
Korlátosság: **korlátos**
1-korlátos (biztos háló)
Holtpontmentesség: **holtpontmentes**
Megfordíthatóság: **megfordítható**
Perszisztencia: **nem perzisztens**

Strukturális tulajdonságok

Legszűkebb osztály: Petri-háló
Tisztaság: **nem tiszta (van hurokél)**

[Elérhetőség vizsgálata:](#) [CTL-kifejezés vizsgálata:](#)
[Elérhetőségi gráf mentése:](#) [Szomszédossági mátrix mentése:](#)
[T-invariánsok keresése:](#) [P-invariánsok keresése:](#)
[Helyek tokenkorlátjainak kiírása:](#)



CTL MODEL CHECKING
Expression: AF(AlterBit.wfa_0>0&EX(AlterBit.buffer_x>0))
Model: AlterBit
Result: True
Runtime: 0,01 s

OK

A PetriDotNet invariáns analízis

The screenshot displays the PetriDotNet application interface. On the left, a sidebar contains navigation links: [Elérhetőségek](#), [Elérhetőségi oldal mentése](#), [Szomszedsági matrix mentése](#), [T-invariánsok keresése](#), [P-invariánsok keresése](#), and [Helyek tokenkorlátjainak kiírása](#). The main area features three windows:

- ShowInvariants**: A window with a dropdown menu showing `{lose(x), sdata(x,0), tout(x)}` and a **Show** button.
- ShowInvariants**: A window with a dropdown menu showing `{ack_0, ack_1, empty(ack)}` and a **Show** button.
- P-Invariants**: A window titled "List of P-Invariants calculated by Martinez-Silva algorithm". It reports "Calculation finished in 0,00 ms. (places=18, transitions=22)" and lists the following invariants:

```
{ack_0, ack_1, empty(ack)}  
{data_x, empty(data), data_y}  
{rts_x, queue_x, wfa_0, rts_y, wfa_1, queue_y}  
{wait_0, buffer_x, ok_x, ok_y, buffer_y, wait_1}
```

An **OK** button is at the bottom.

On the right, a larger window titled **T-Invariants** displays the "List of T-Invariants calculated by Martinez-Silva algorithm". It reports "Calculation finished in 15,60 ms. (places=18, transitions=22)" and lists 22 invariants, each represented as a set of transitions. The list includes invariants such as `{lose(x), sdata(x,0), tout(x)}`, `{lose(y), sdata(y,1), tout(y)}`, `{rack(1), put(x), sdata(x,0), rack(0), sdata(y,1), put(y), drop(y), sack(0), drop(x), sack(1)}`, and others. An **OK** button is at the bottom.

Példa modell: Az alternáló bit protokoll

A modellezési feladat

Alternating Bit Protocol

- Átviteli protokoll veszteséges csatornához
 - Üzenet **elveszhet** (véges számú alkalommal)
 - Üzenet tartalma nem változhat
- Cél: a protokoll biztosítsa, hogy minden üzenet (véges számú próbálkozással) eljusson a vevőhöz

Küldő folyamat

- A küldő az üzenetekhez egy ellenőrző bitet kapcsol
- Az üzenetek megérkezését a vevőtől nyugta jelzi, ugyanazzal az ellenőrző bittel
- Ha a küldött üzenethez csatolt bit **b**, akkor
 - Ha az üzenet elvész, a küldő időtúllepéssel észleli a nyugta hiányát → újra küldi
 - Ha a folyamat **b** bittel ellátott nyugtát kap (ilyet várt), akkor a következő üzenethez $\neg\mathbf{b}$ bitet csatol
 - Ha a folyamat $\neg\mathbf{b}$ bittel ellátott nyugtát kap (nem ilyet várt), akkor egyszerűen eldobja (majd időtúllépés lesz a **b** bites nyugta hiánya miatt)

Fogadó folyamat

- Az üzenet vételét nyugtázza a kapott ellenőrző bit visszaküldésével
- Ha **b** ellenőrző bittel jelölt üzenetet kap valamint ezt a **b** bit visszaküldésével nyugtázza, akkor
 - Ha a következő üzenetben az ellenőrző bit értéke $\neg \mathbf{b}$ (helyesen), akkor az új üzenetet is feldolgozza és a $\neg \mathbf{b}$ bit visszaküldésével nyugtázza
 - Ha a következő üzenetben az ellenőrző bit értéke **b** (azaz nem megfelelő), akkor az üzenetet eldobja, de **b** nyugtát küld (mondván, hogy bizonyára újraküldés történt a nyugta hiánya miatt)

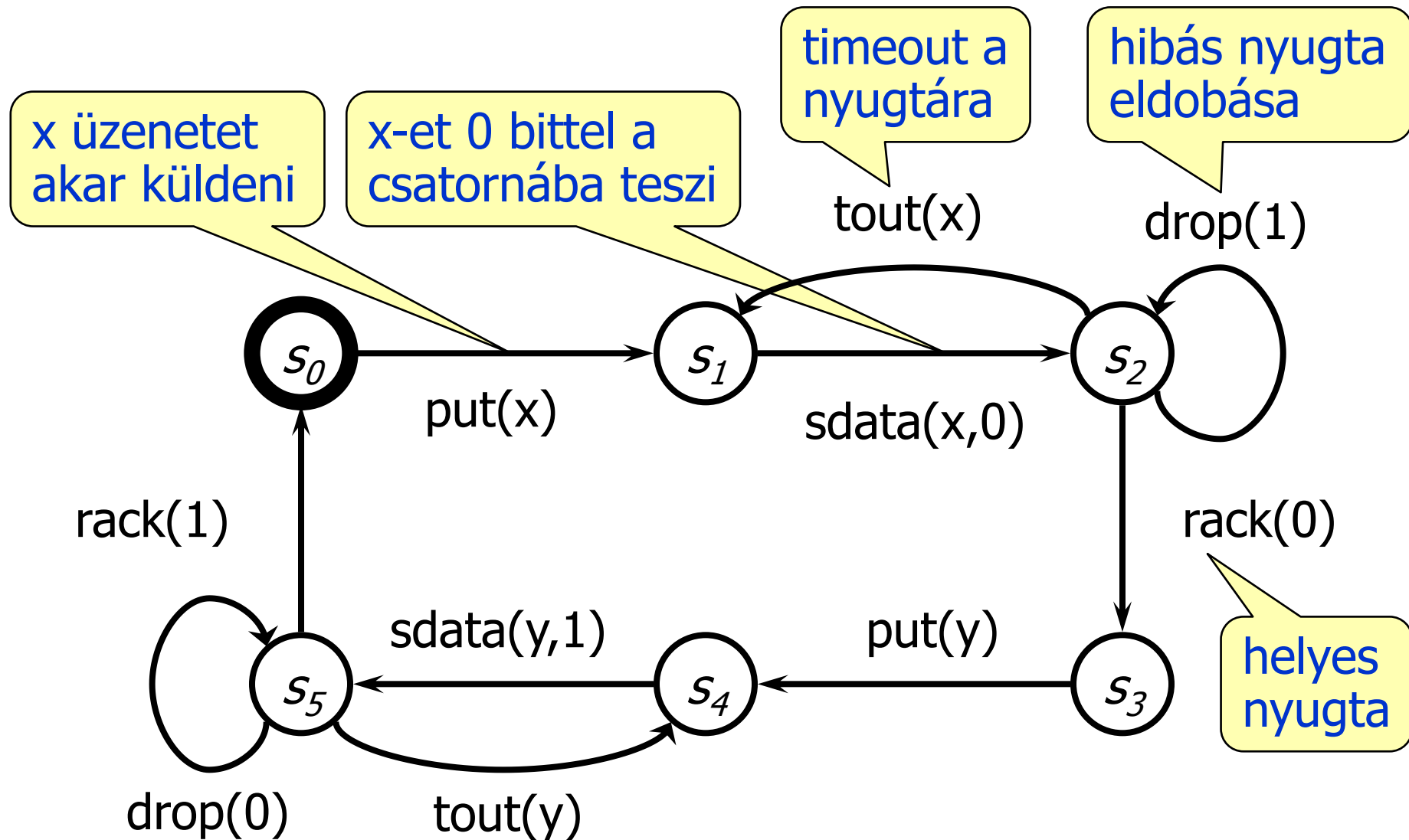
A modellalkotás lépései

1. A feladat felbontása folyamatokra és erőforrásokra
2. Folyamatok állapotainak meghatározása
3. Erőforrások állapotainak meghatározása
4. Állapot alapú modellekből Petri-háló modell készítése
5. Folyamatok és erőforrások modelljeinek integrálása
 - Folyamat lépése módosítja az erőforrás állapotát
 - Erőforrás állapota feltétel egy folyamat lépéséhez
6. Integrált modell helyességének ellenőrzése
7. Modell felhasználása a feladat megoldására

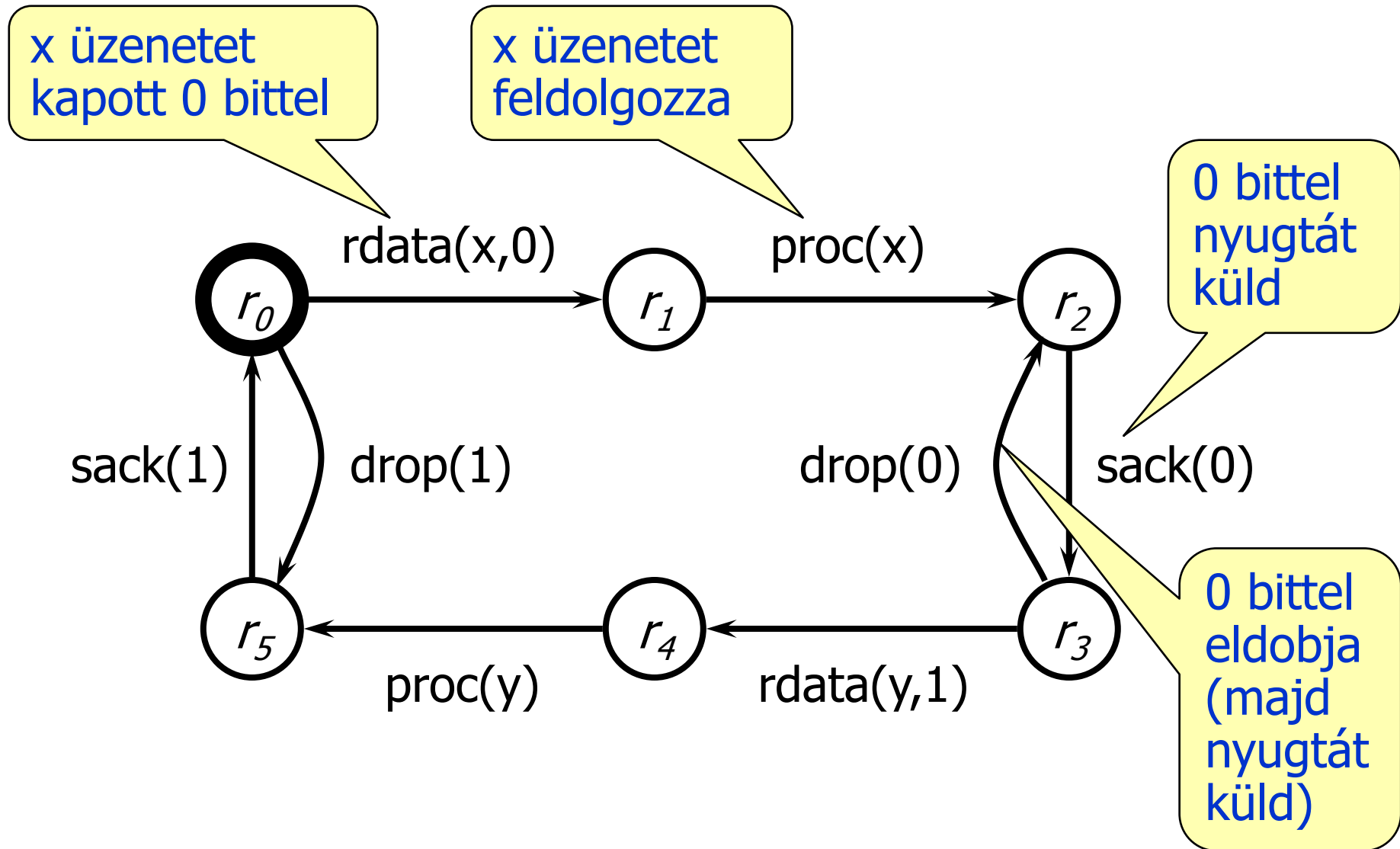
Komponensek és állapotaik

- Komponensek (alrendszerek)
 - Folyamatok: küldő folyamat, fogadó folyamat
 - Erőforrások: adat csatorna, nyugtázó csatorna
- Minden komponens saját állapottal rendelkezik
 - Állapotgráf: állapotok körökkel, események nyilakkal
- Azonos események egy időben mennek végbe: szinkronizáció

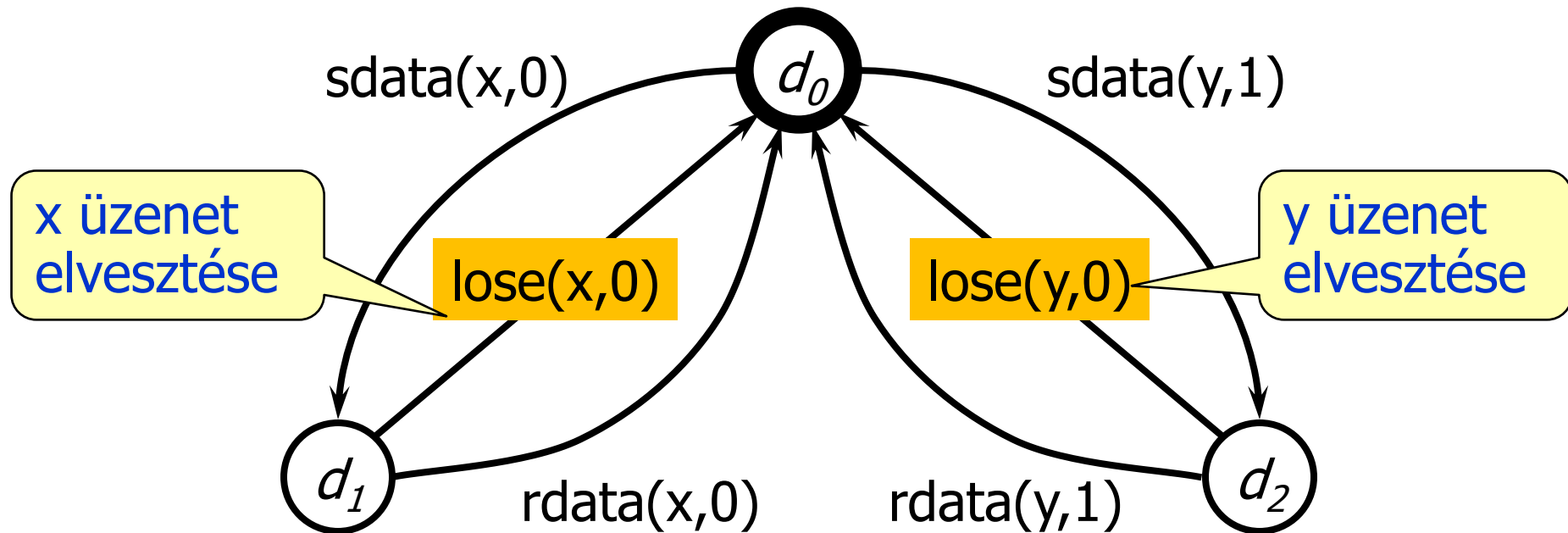
Küldő folyamat állapotgráfja



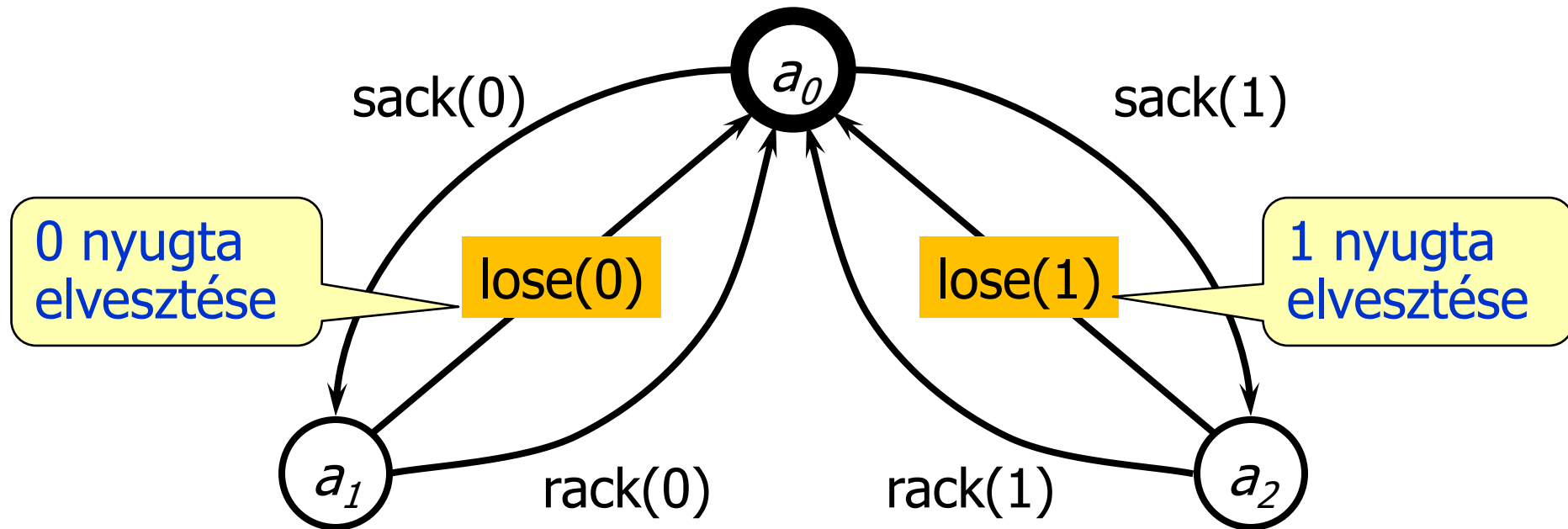
Fogadó folyamat állapotgráfja



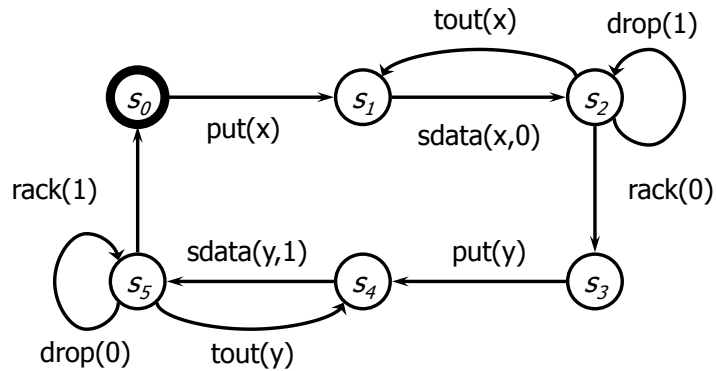
Adat csatorna állapotgráfja



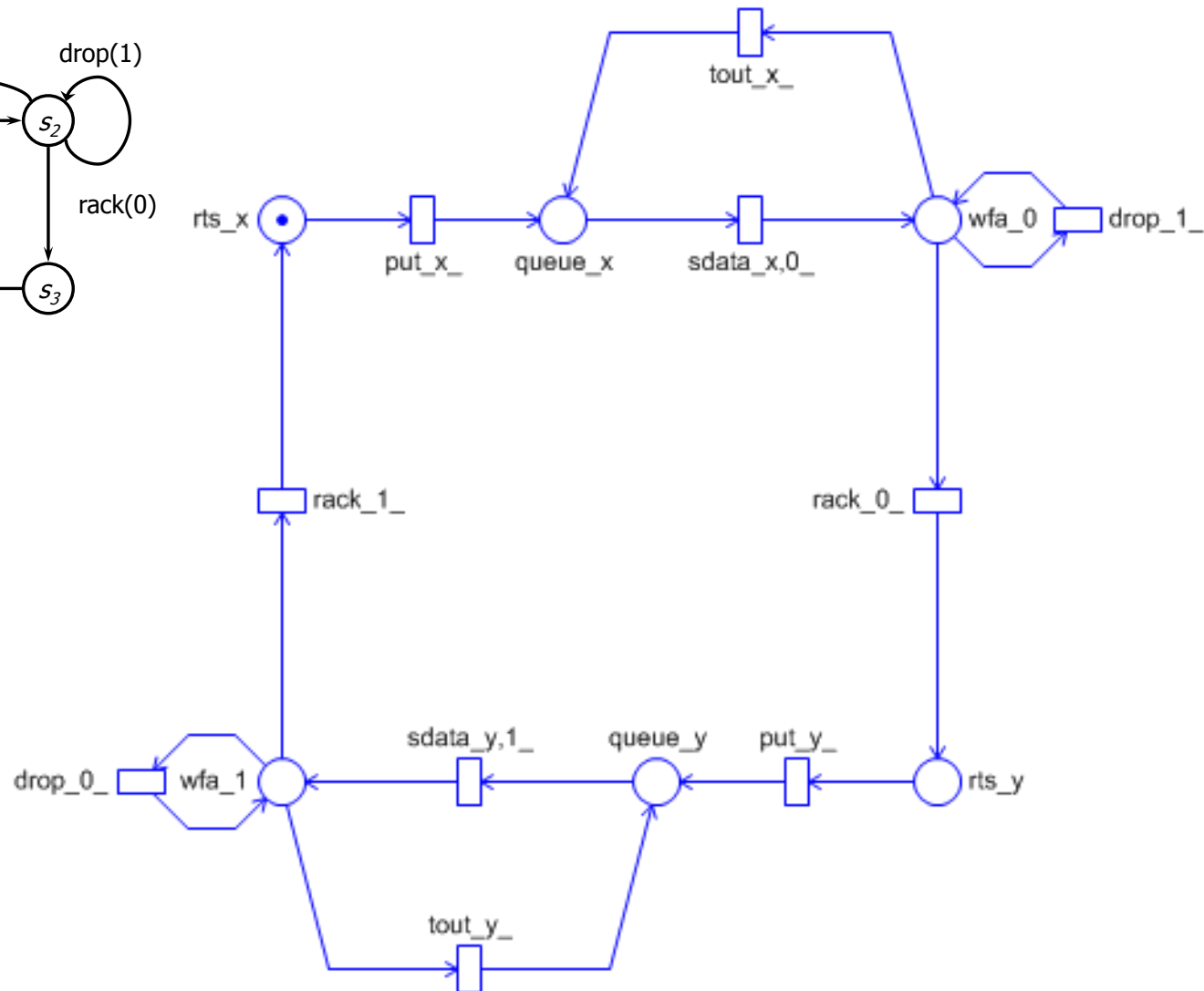
Nyugtázó csatorna állapotgráfja



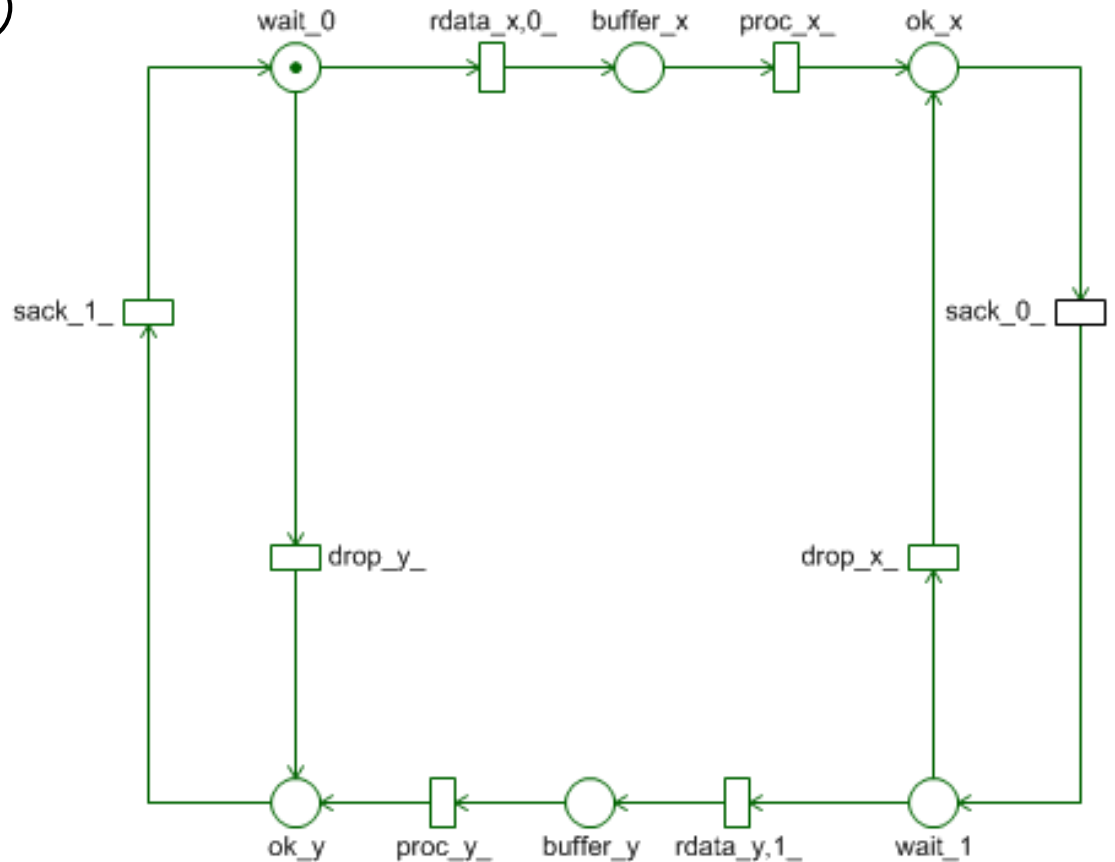
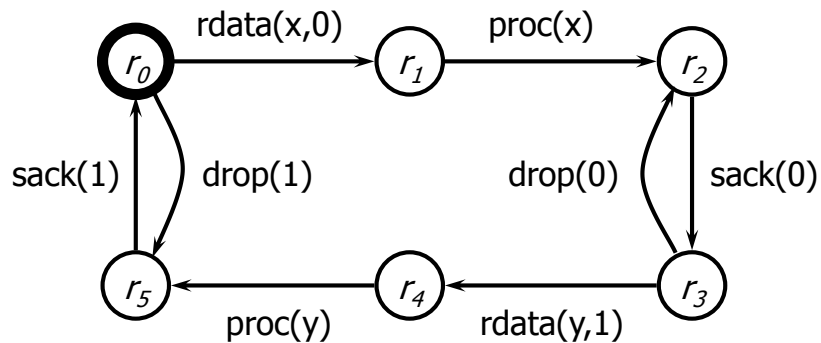
Küldő folyamat Petri háló modellje



Zárójelek
helyett ' ' van
a modellben



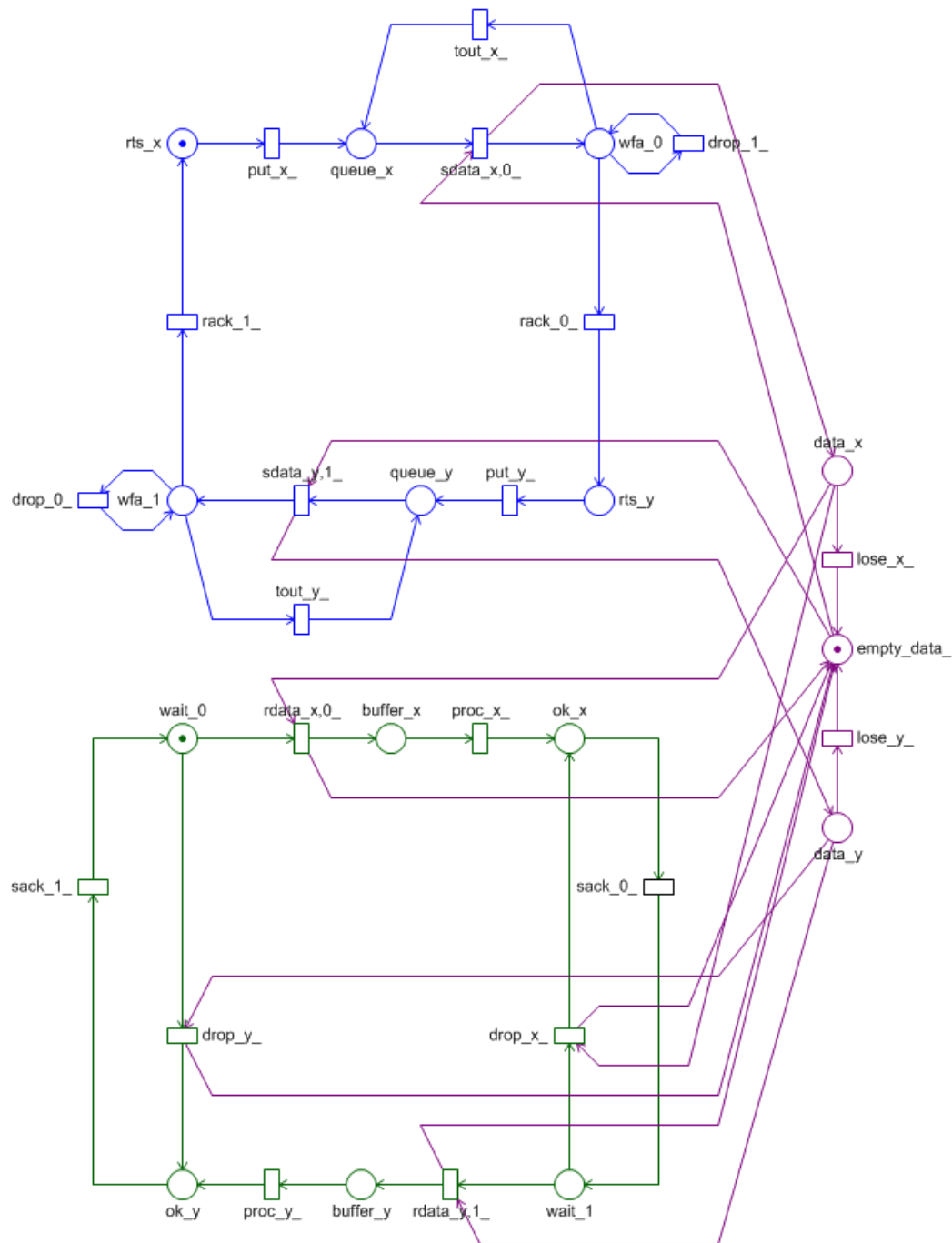
Fogadó folyamat Petri háló modellje



Adat csatorna és adatátvitel

Adat csatorna állapotának módosítása (élekkel bekötve):

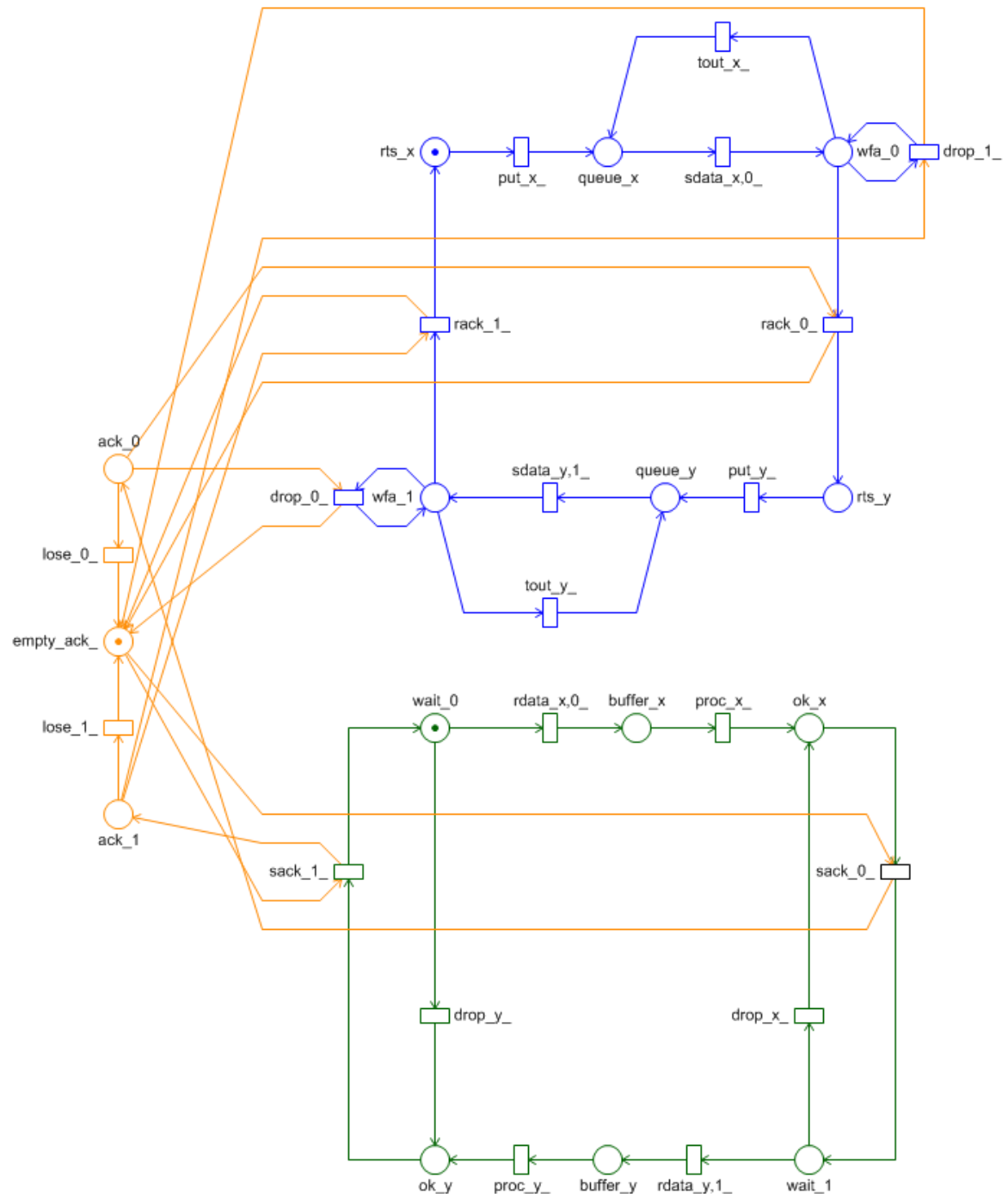
- sdata()
- rdata()
- drop()
- lose()



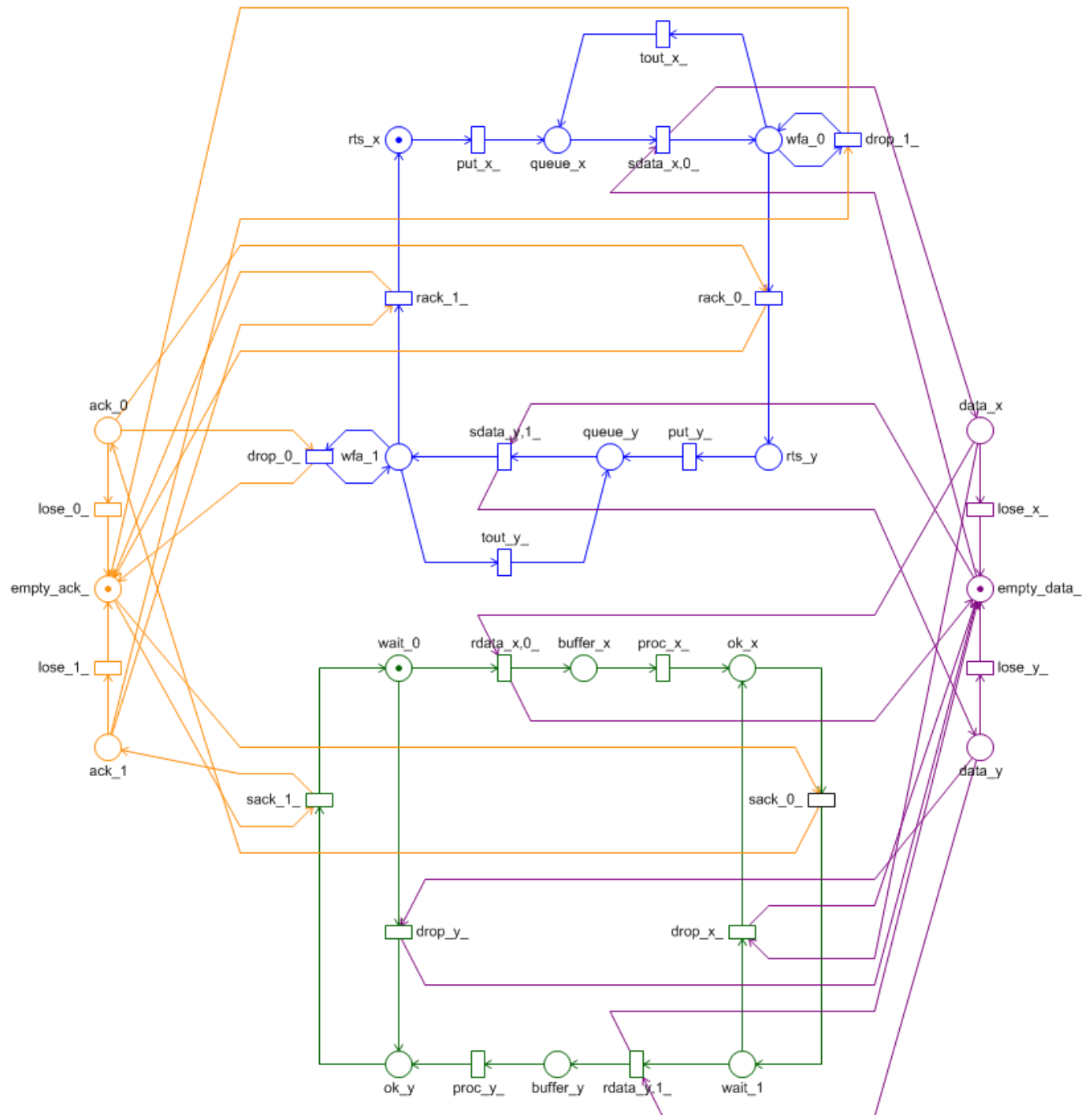
Nyugtázó csatorna és nyugtázás

Nyugtázó csatorna állapotának módosítása (élekkel bekötve):

- sack()
- rack()
- drop()
- lose()



Teljes modell



A példa modell analízise

DNAnet: A modell dinamikus tulajdonságai

Net is bounded.

Deadlock is not possible.

Net is live.

Net has home states.

Coverability graph generation statistics:

108 unique markings

1 strongly connected components

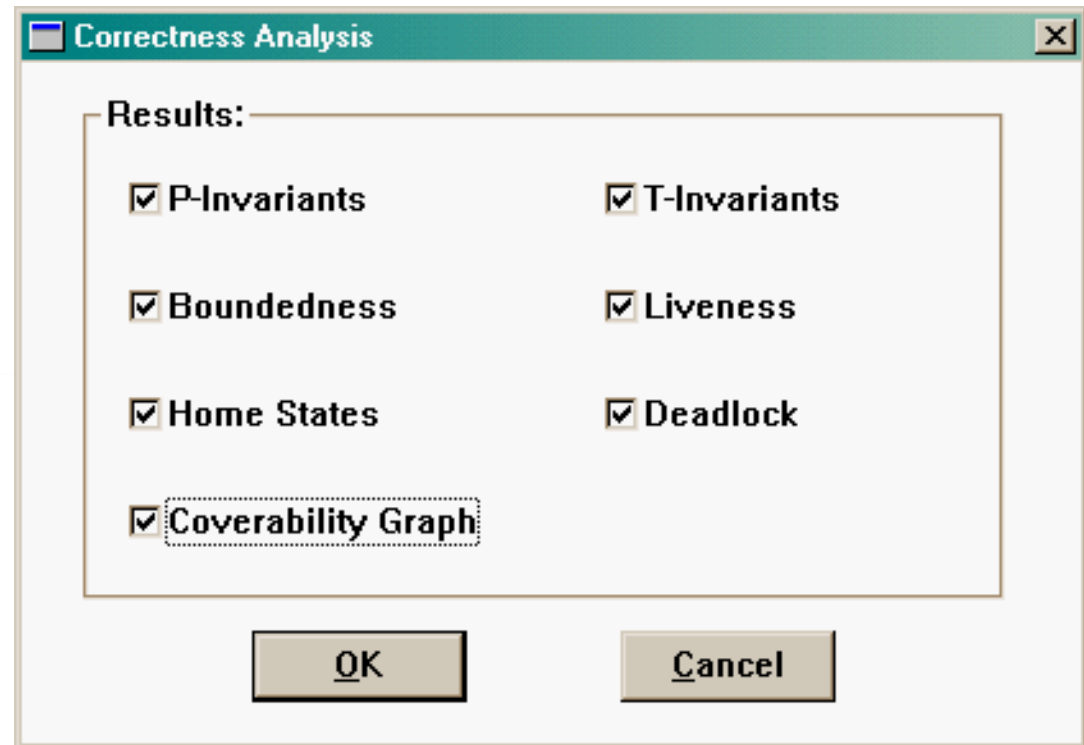
108 hash table entries used

1 was the longest hash list length

1 was the average hash list length

27 was the maximum stack height

3 was the maximum component stack height



PetriDotNet: A modell dinamikus tulajdonságai

A(z) AlterBit háló tulajdonságai

Dinamikus tulajdonságok

Állapotok száma:	108
Korlátosság:	korlátos 1-korlátos (biztos háló)
Holtpontmentesség:	holtpontmentes
Megfordíthatóság:	megfordítható
Perszisztencia:	nem perzisztens

Strukturális tulajdonságok

Legszűkebb osztály:	Petri-háló
Tisztaság:	nem tiszta (van hurokél)

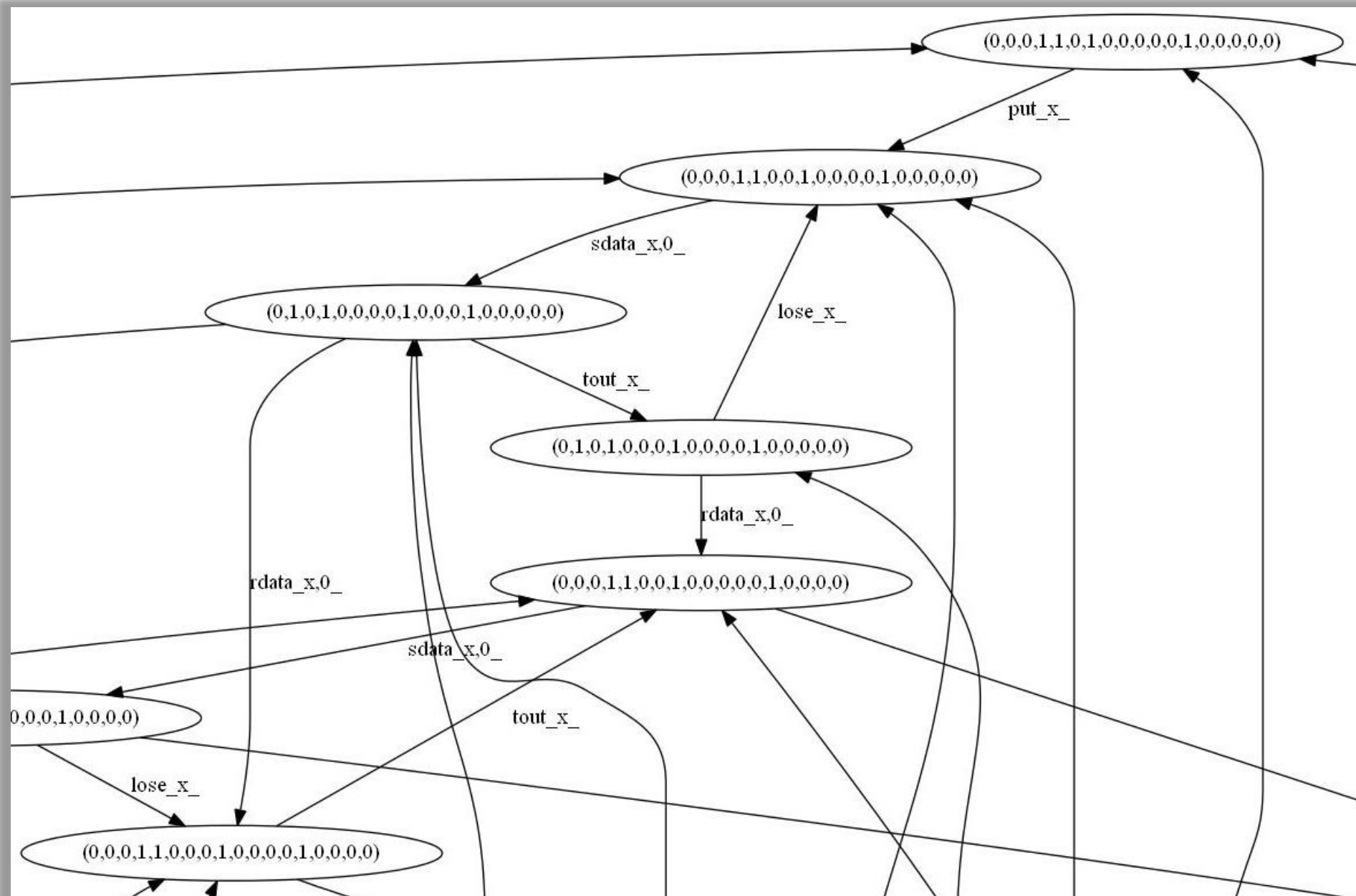
[Elérhetőség vizsgálata:](#) [CTL-kifejezés vizsgálata:](#)

[Elérhetőségi gráf mentése:](#) [Szomszédossági mátrix mentése:](#)

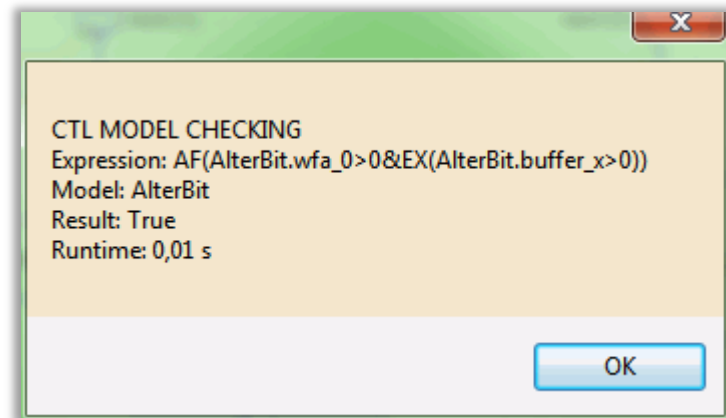
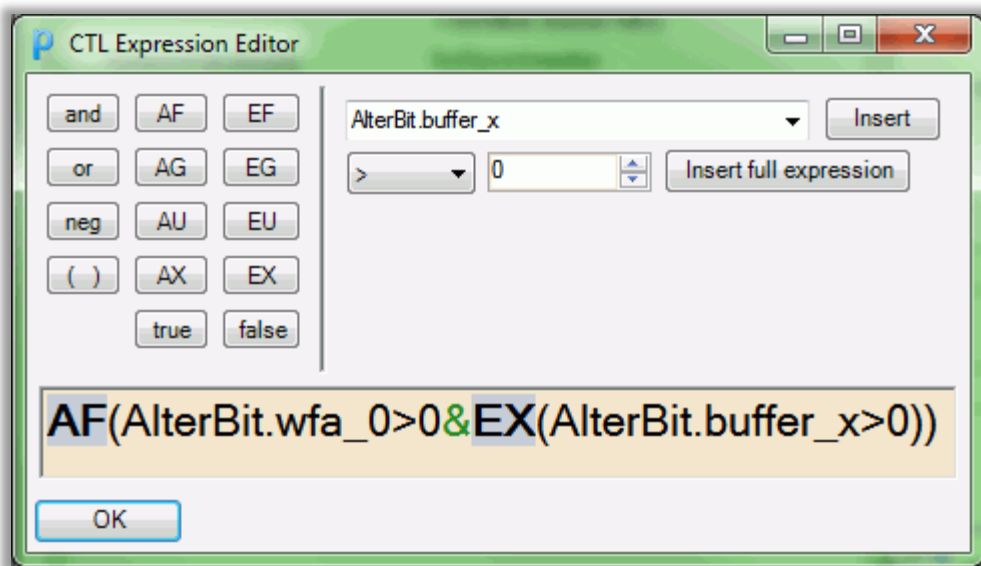
[T-invariánsok keresése:](#) [P-invariánsok keresése:](#)

[Helyek tokenkorlátjainak kiírása:](#)

PetriDotNet: Elérhetőségi gráf kirajzolása (GraphViz)



PetriDotNet: CTL modellellenőrzés



AF(AlterBit.wfa_0>0 & **EX**(AlterBit.buffer_x>0))

⇒ True

AG(**AF**(AlterBit.buffer_y>0))

⇒ False

AF(**EG**(AlterBit.buffer_x=0))

⇒ True

EF(AlterBit.wfa_0>0 & AlterBit.data_x=0)

⇒ True

AF(AlterBit.queue_x>0 & **AX**(AlterBit.wfa_0>0 & AlterBit.data_x>0)) ⇒ True

PetriDotNet: Invariáns analízis

The screenshot displays the PetriDotNet application with three windows open:

- Háló tulajdonságai** (Network Properties):
 - ShowInvariants**: Shows the expression `{lose(x), sdata(x,0), tout(x)}` with a **Show** button.
 - ShowInvariants**: Shows the expression `{ack_0, ack_1, empty(ack)}` with a **Show** button.
 - P-Invariants**:
 - Title: List of P-Invariants calculated by Martinez-Silva algorithm
 - Message: Calculation finished in 0,00 ms. (places=18, transitions=22)
 - Content:

```
{ack_0, ack_1, empty(ack)}  
{data_x, empty(data), data_y}  
{rts_x, queue_x, wfa_0, rts_y, wfa_1, queue_y}  
{wait_0, buffer_x, ok_x, ok_y, buffer_y, wait_1}
```
 - Buttons: **OK**
- T-Invariants**:
 - Title: List of T-Invariants calculated by Martinez-Silva algorithm
 - Message: Calculation finished in 15,60 ms. (places=18, transitions=22)
 - Content: A long list of T-invariant expressions, including:

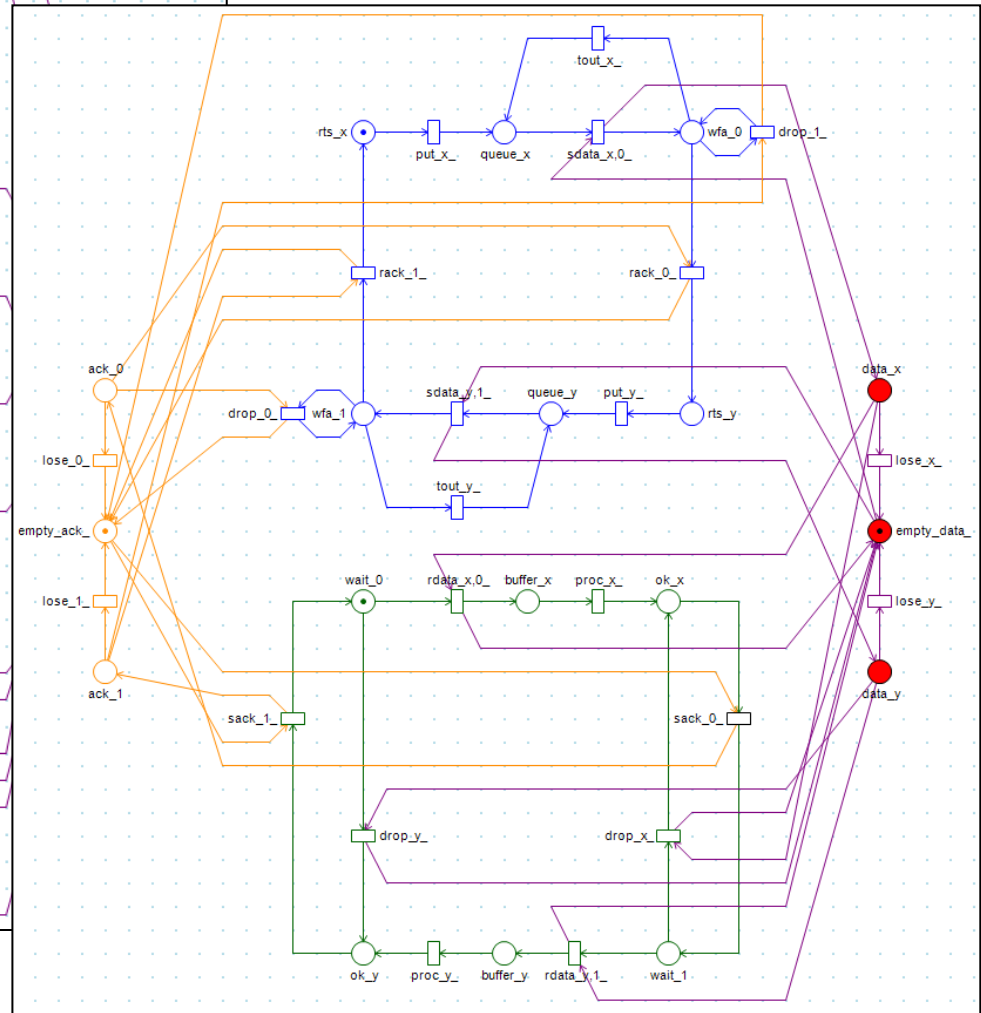
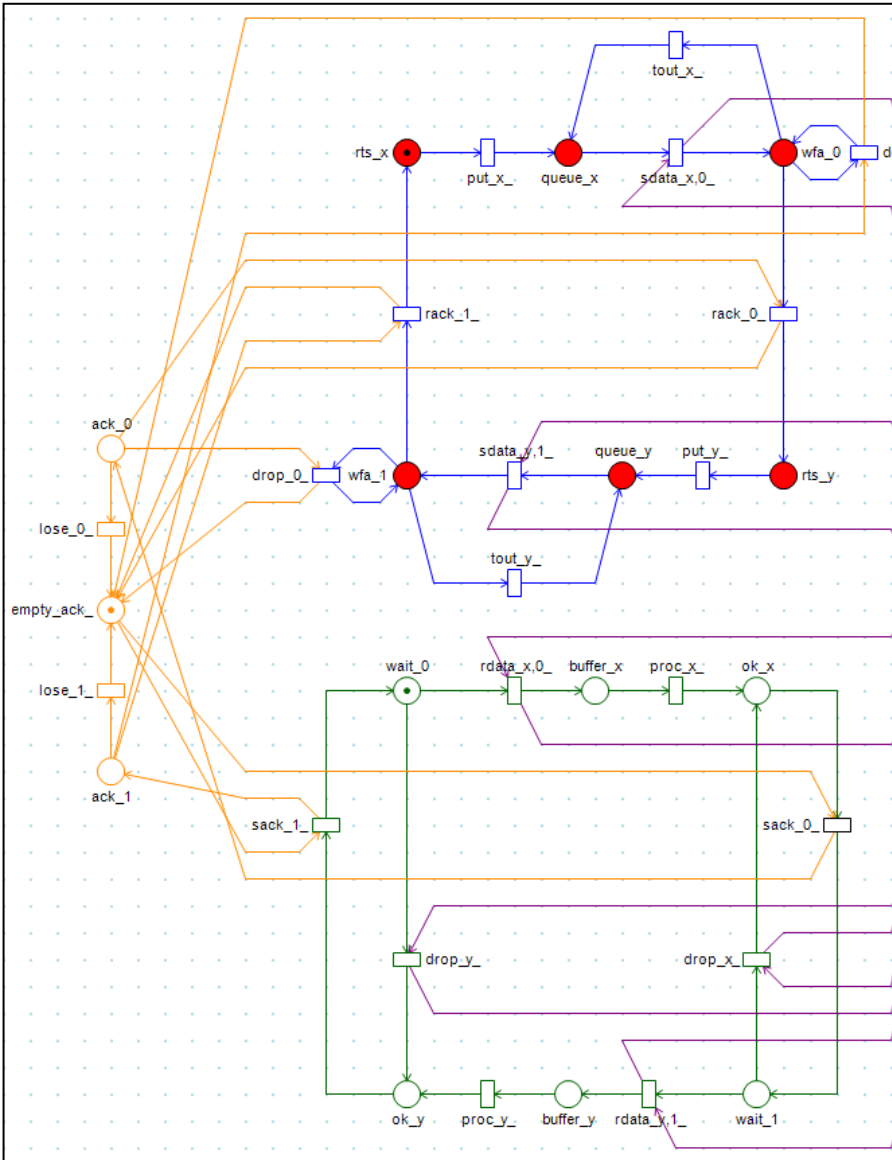
```
{lose(x), sdata(x,0), tout(x)}  
{lose(y), sdata(y,1), tout(y)}  
{rack(1), put(x), sdata(x,0), rack(0), sdata(y,1), put(y), drop(y), sack(0), drop(x), sack(1)}  
{lose(1), sdata(y,1), tout(y), drop(y), sack(1)}  
{drop(1), sdata(y,1), tout(y), drop(y), sack(1)}  
{lose(y), rack(1), put(x), sdata(x,0), rack(0), sdata(y,1), put(y), tout(y), drop(y), sack(0), drop(x), sack(1)}  
{lose(0), sdata(x,0), tout(x), sack(0), drop(x)}  
{sdata(x,0), tout(x), drop(0), sack(0), drop(x)}  
{lose(x), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), drop(y), sack(0), drop(x), sack(1)}  
{lose(x), lose(y), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), tout(y), drop(y), sack(0), drop(x), sack(1)}  
{rack(1), put(x), sdata(x,0), rack(0), sdata(y,1), put(y), rdata(x,0), proc(x), sack(0), proc(y), rdata(y,1), sack(1)}  
{lose(x), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), rdata(x,0), proc(x), sack(0), proc(y), rdata(y,1), sack(1)}  
{rack(1), put(x), sdata(x,0), rack(0), sdata(y,1), put(y), rdata(x,0), proc(x), drop(y), sack(0), drop(x), proc(y), rdata(y,1), sack(1)}  
{lose(0), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), rdata(x,0), proc(x), sack(0), drop(x), proc(y), rdata(y,1), sack(1)}  
{rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), drop(0), rdata(x,0), proc(x), sack(0), drop(x), proc(y), rdata(y,1), sack(1)}  
{lose(x), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), rdata(x,0), proc(x), drop(y), sack(0), drop(x), proc(y), rdata(y,1), sack(1)}  
{lose(x), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), rdata(x,0), proc(x), drop(y), sack(0), drop(x), proc(y), rdata(y,1), sack(1)}  
{lose(x), lose(y), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), tout(y), rdata(x,0), proc(x), drop(y), sack(0), drop(x), proc(y), rdata(y,1), sack(1)}  
{lose(x), lose(1), rack(1), put(x), sdata(x,0), tout(x), rack(0), sdata(y,1), put(y), tout(y), rdata(x,0),
```
 - Buttons: **OK**

At the bottom of the **Háló tulajdonságai** window, there are links for further actions:

- [Elérhetőségek](#)
- [Elérhetőségi oldal mentése](#)
- [Szomszedsági matrix mentése](#)
- [T-invariánsok keresése](#)
- [P-invariánsok keresése](#)
- [Helyek tokenkorlátjainak kiírása](#)

PetriDotNet: P-invariánsok (példák)

Komponensek
állapotgépei



PetriDotNet: T-invariánsok (példák)

Ciklikus működések (itt:
hibátlan ill. adatvesztés)

