

Kommunikációs rendszerek teljesítőképeség-vizsgálata

(2. előadás)

Dr. Lencse Gábor
`lencse@sze.hu`

<https://www.tilb.sze.hu/cgi-bin/tilb.cgi?0=m&1=targyak&2=krtv>

Miről lesz szó?

- Emlékeztető: az eseményvezérelt diszkrét idejű szimuláció algoritmus
- Párhuzamos szimuláció (PDES)
- Forgalom-folyam analízis (TFA)
- Kombinált módszerek
- További gyorsítási lehetőségek

Az eseményvezérelt diszkrét idejű szimuláció működése

- FES: Future Event Set – a jövőbeli események halmaza
- A működés algoritmus:

Inicializálás, indító esemény(ek) berakása a FES-be;

repeat

legkisebb időbélyegű esemény kivétele a FES-ből;

MOST := a kivett esemény időbélyege;

esemény feldolgozása, közben új események felidőzítése (ha szükséges);

until (MOST > határ) v (elfogytak az események) v
(más miatt meg kell állni)

A szekvenciális eseményvezérelt diszkrét idejű szimuláció korlátai

- Nagy rendszer részletes modellje esetén gondot okozhat:
 - A modell tárolása (pl. 32 bites gépen 4GB RAM)
 - Az igényelt számítási kapacitás (elfogadhatatlanul hosszú végrehajtási idő)
- Megoldásként kínálkozik:
 - Párhuzamosítás
 - Más, szimulációval rokon teljesítőképeség-vizsgálati módszer (TFA)
 - A fentiek kombinációja

Párhuzamos diszkrét idejű szimuláció

- A rendszer modelljét szegmensekre osztjuk
- A szegmenseket processzorokhoz rendeljük
- Az egyes szegmensek modellbeli idejét szinkronizálni kell (kauzalitás!)
- Az elérhető gyorsulás (a szekvenciálishoz képest) függ:
 - A modell szegmensekre bontásától
 - Az alkalmazott szinkronizációs módszertől
 - A processzorok közötti kommunikációs csatornától

Párhuzamos diszkrét idejű szimulációs módszerek

- Konzervatív (kauzalitás nem sérülhet)
- Optimista (kauzalitás sérülésnél „roll-back”)
- Statisztikai szinkronizációs módszer
(csak az üzenetfolyamot leíró statisztikákat
cseréli ki bizonyos időnként)

A konzervatív módszer

- A kauzalitás biztosítása érdekében egy eseményt csak akkor szabad feldolgozni, ha garantált, hogy annál kisebb időbélyegű esemény (üzenet) már nem fog érkezni (más szegmensből)
- Bizonyos feladatosztályoknál a modell (modellezett rendszer) jellegéből adódóan ilyen garanciák lehetségesek (fogalmak: lookahead, null messages)
- Általános esetben nem ad gyorsulást, hanem lassulás lesz: mindig csak egy proc. dolgozik... 7

Az optimista módszer

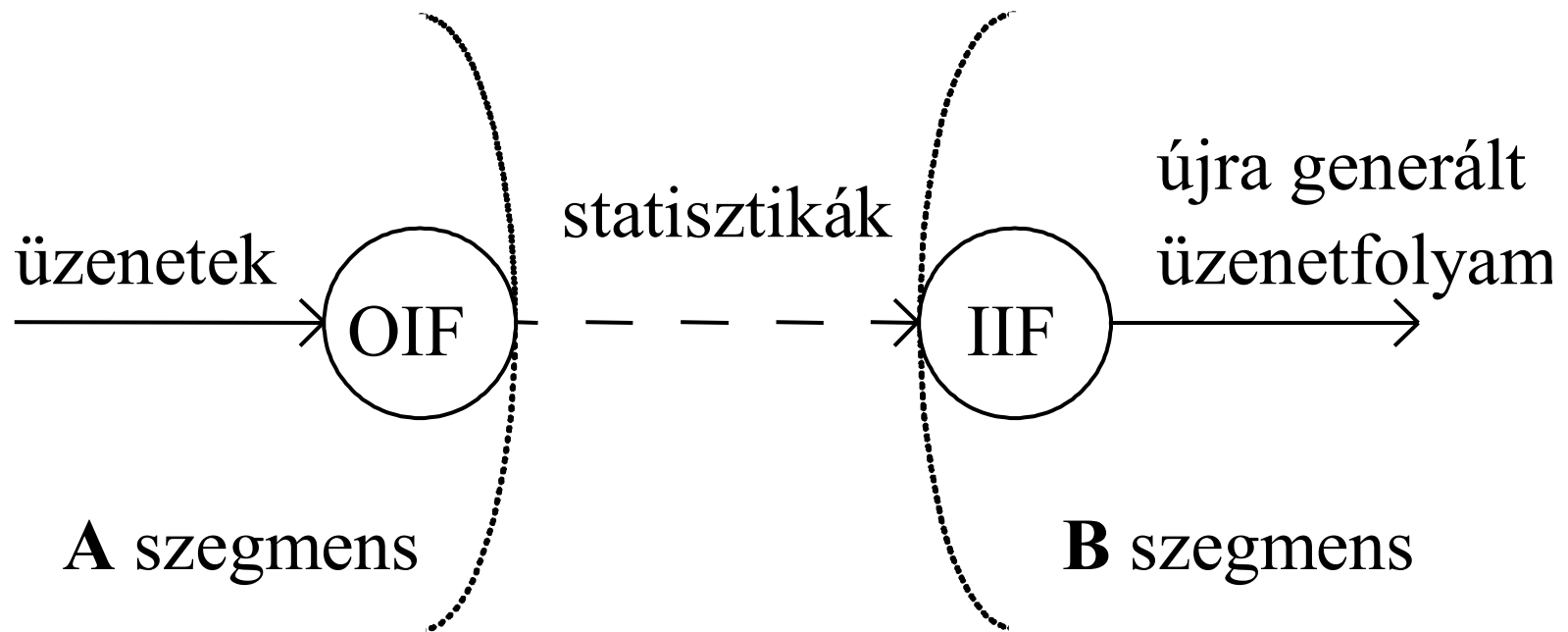
- Megengedi a kauzalitás sérülését
- Kauzalitás sérülésénél (az aktuális modellidőnél kisebb időbélyegű üzenet érkezik) „*roll-back*”: visszacsinálni mindent, ami azóta történt:
 - állapotváltozók visszaállítása
 - elküldött üzenetek visszavonása (anti-messages) (beleértve a saját magának felidőzítettek törlését)
- Az állapotmentés/visszaállítás komoly probléma
- Rosszul skálázható: méret (szegmens szám) növekedésnél egyre nagyobb problémát okoz a roll-back

Az eredeti statisztikai szinkronizációs módszer

- Az egyes szegmenseket az egymással való kommunikációs csatornák mentén kimeneti és bemeneti interfészekkel látjuk el
- A más szegmensnek szóló üzeneteket a kimeneti interfészek nem továbbítják, hanem statisztikát gyűjtenek róluk
- A statisztikákat a kimeneti interfészek a megfelelő bemeneti interfészeknek továbbítják
- A bemeneti interfészek a kapott statisztikák alapján generálnak üzeneteket

Az eredeti statisztikai szinkronizációs módszer

A módszer alapötlete rajzban:



rendszerek állandósult állapotbeli viselkedésének vizsgálatára alkalmas (Pongor, 1992)

Az SSM-T módszer

Az eredeti SSM kiegészítése *laza időszinkronizációval*:

Az **A** és **B** szegmensek közötti laza időszinkronizáció formális definíciója a következő:

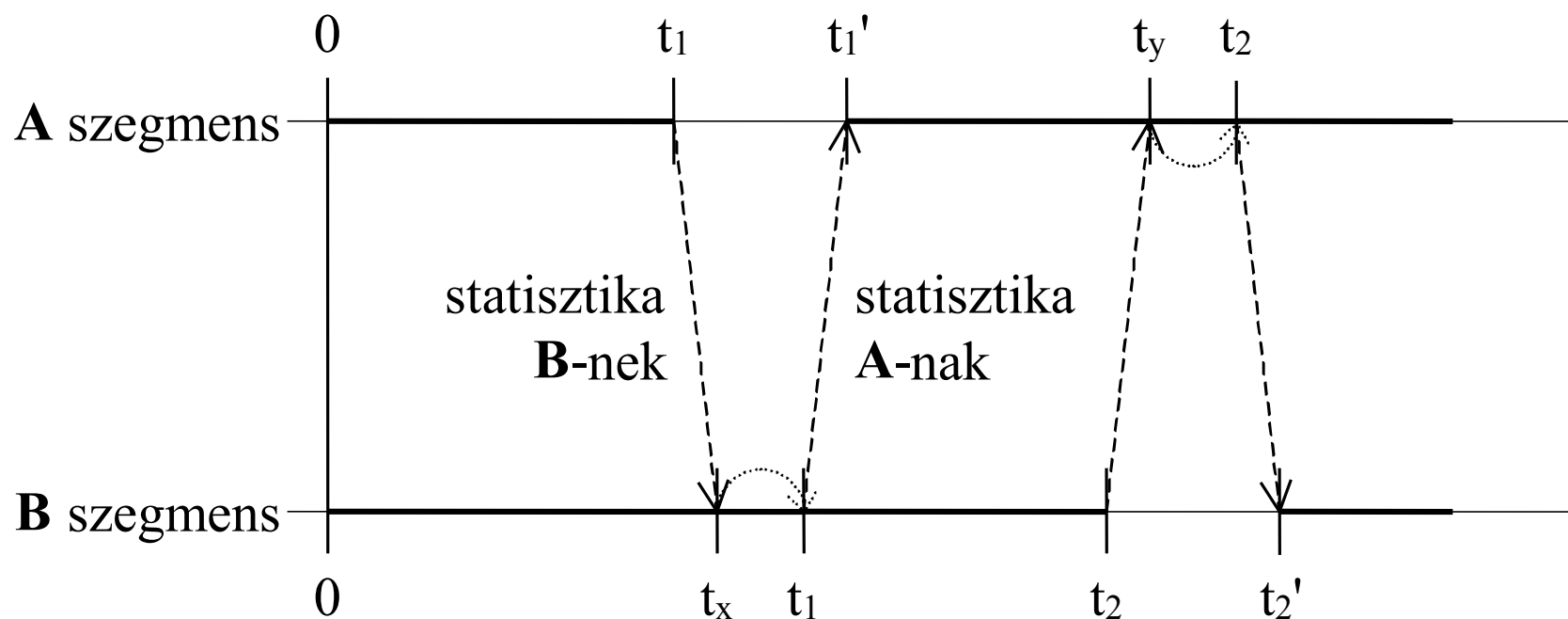
Legyenek $t_1, t_2, \dots, t_i, \dots, t_n$ szinkronizációs időpontok. Jelölje t_A és t_B az **A** illetve **B** szegmens lokális virtuális idejét. Az **A** és **B** szegmens között laza időszinkronizáció van, ha:

$$((t_A < t_i) \Rightarrow (t_B \leq t_i)) \wedge ((t_A > t_i) \Rightarrow (t_B \geq t_i)), \quad i = 1, 2, \dots, n.$$

A két szegmens közötti laza időszinkronizáció azt jelenti, hogy egyik sem hagyhat el egy szinkronizációs időpontot, míg a másik el nem érte azt.

Az SSM-T működésének illusztrációja

A statisztikacsere lezajlása két szegmens között:



A szegmensek virtuális ideje közt laza időszinkronizáció van.

Az SSM-T alkalmazhatósági kritériumok - I.

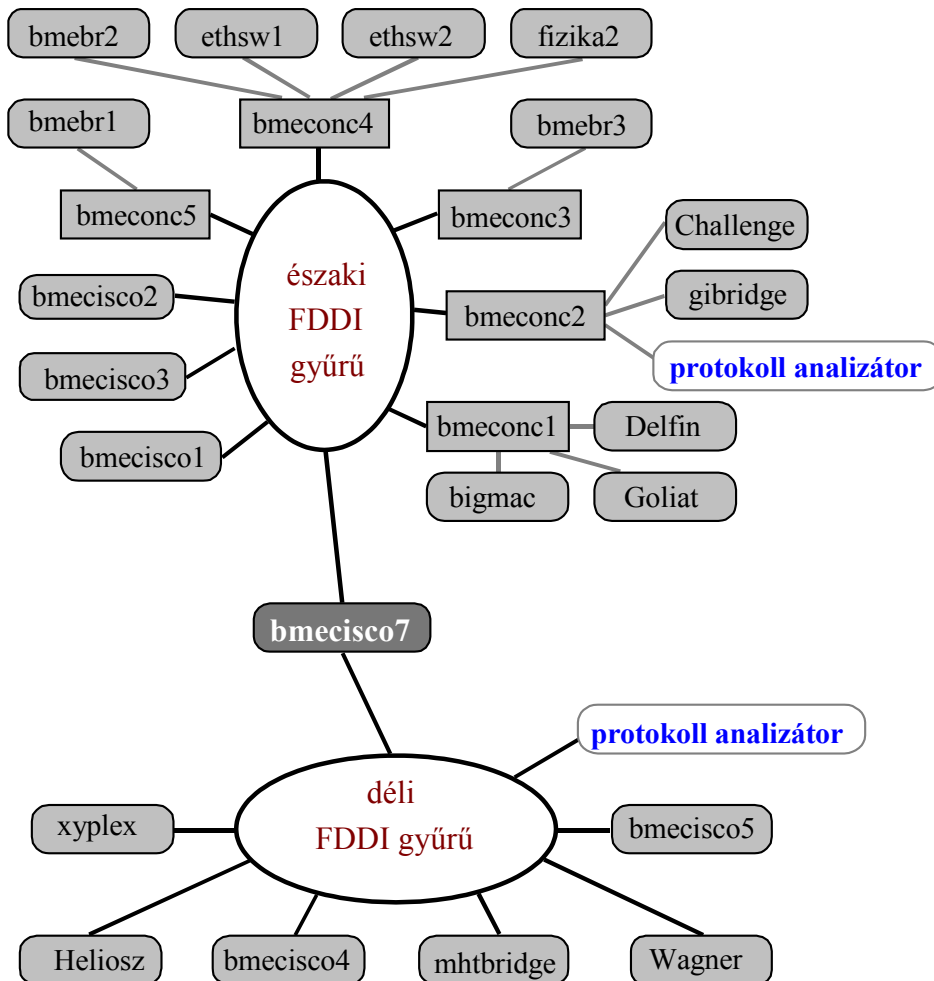
- A szimulált rendszer szegmensekre osztható úgy, hogy a szegmensek közötti forgalomban csak az üzenetfolyam statisztikai jellemzői számítanak, az egyes üzenetek maguk a rendszer működése és az általunk megfigyelt kimeneti jellemzők szempontjából nem fontosak.
- Az üzenetfolyam statisztikai jellemzőinek közelítésében egy *kis hiba* a kimenetben is *kis hibát* okoz, ami *csak a közelítés hibájának a mértékétől függ.*
- A modell paramétereit változhatnak a szimuláció alatt, de a szegmensek közötti üzenet-folyamok statisztikai jellemzőinek változásai *kellően ritkák.*

Az SSM-T alkalmazhatósági kritériumok - II.

- Mivel egy üzenetfolyam statisztikai jellemzőinek változása csak akkor jut el más szegmensekhez, amikor a kimeneti interfész elküldi az általa a megváltozott statisztikai jellemzők szerint gyűjtött statisztika csomagot, a negyedik kritérium az, hogy ez *a késleltetés csak a késleltetés alatt, vagy még legfeljebb a késleltetés idejével arányos ideig okoz hibát a szimuláció kimenetében.*

Formalizálás és bizonyítás: (Lencse, 1999a)
magyarul: (Lencse, 2000)

Az SSM-T alkalmazásához használt példahálózat topológiája



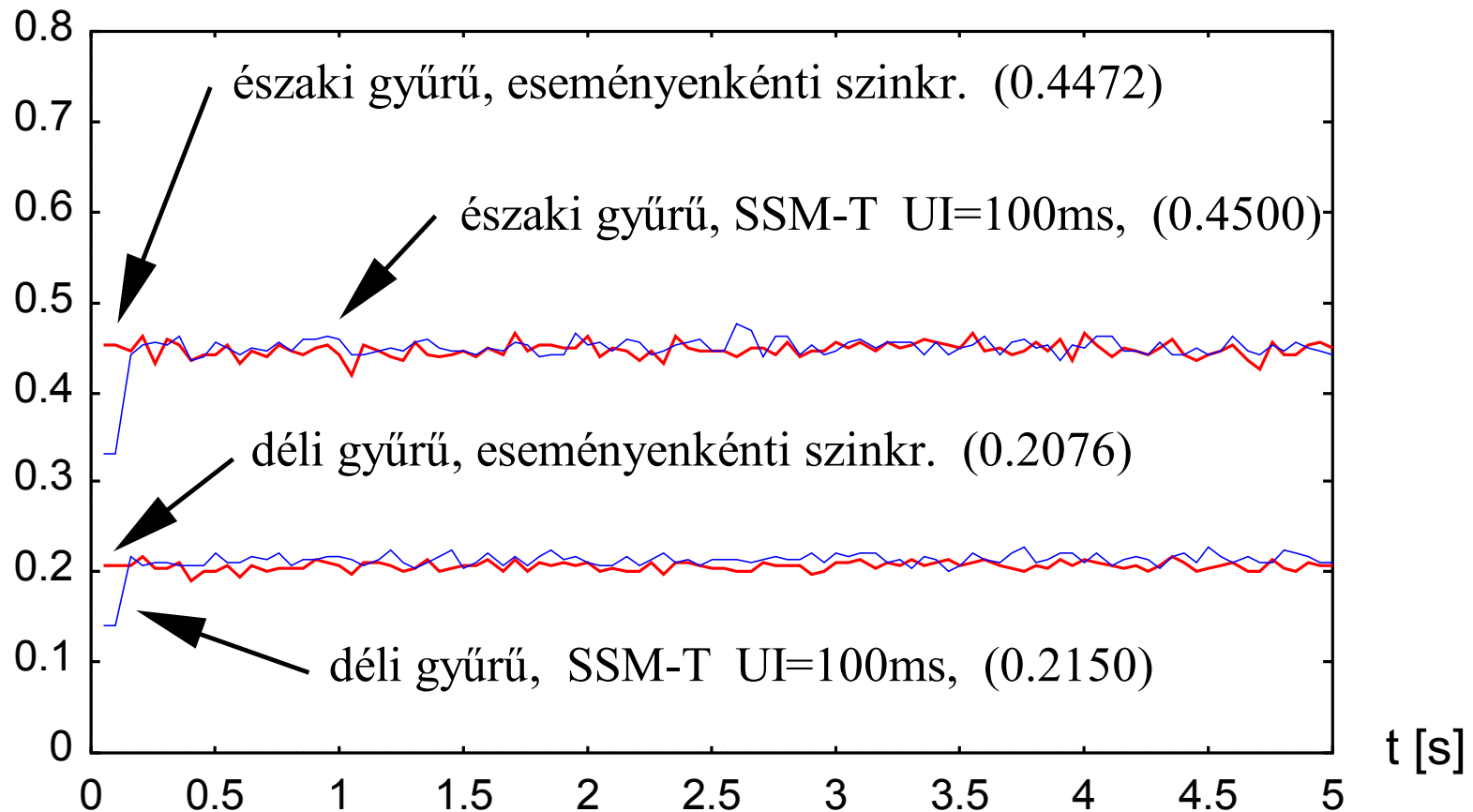
Jelmagyarázat:



A vizsgált hálózat a BME FDDI gerinchálózata volt az 1996/97-es tanévben

Az SSM-T alkalmazásához használt példahálózat szimulációjának eredményei

kihasználtság (1.0=100%)



Az FDDI gyűrűk kihasználtsága az idő függvényében az SSM-T módszer használata esetén, illetve anélkül

Az SSM-T alkalmazásával elért gyorsulás (speed-up)

- A két összekapcsolt FDDI gyűrű párhuzamos szimulációjánál a szimulációt SSM-T alkalmazásával két processzoron futtatva 1.91 illetve 1.86 szoros gyorsulást értem el az egy processzoros szimulációhoz képest (statisztika csere 100, illetve 10 ms-onként).
- A kimenet hibája az 1.91 szeres gyorsulást adó esetben az egyik gyűrű esetén 0.63% a másik gyűrű esetén pedig 3.56% volt az egyprocesszoros szimulációhoz képest.

Traffic-Flow Analysis (TFA) (forgalom-folyam analízis)

- A TFA célja, hogy pillanatfelvételt adjon a hálózat forgalmi viszonyairól
- A TFA maga is szimuláció és numerikus módszerek kombinációja
- Az alkalmazások forgalmát statisztikákkal reprezentáljuk (aggregált forgalommodell)
- Többféle forgalommodellel működhet (amiknek ki kell elégíteniük az alkalmazhatósági kritériumokat), jelenleg egy van kidolgozva

A TFA működési lépései

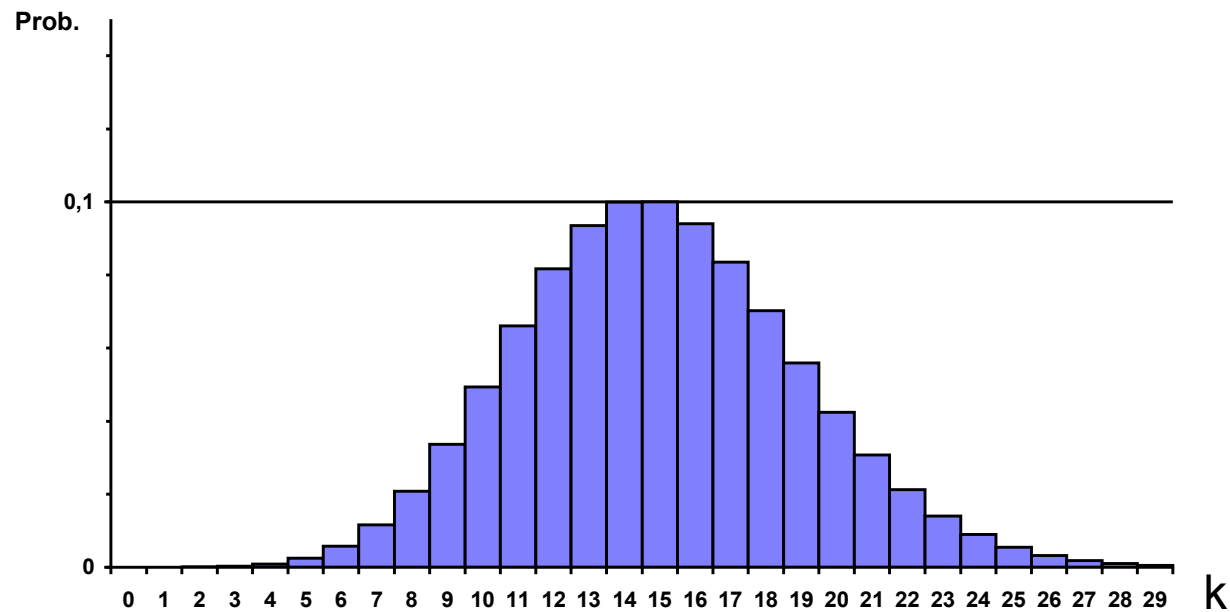
1. A statisztikák a hálózat eredeti útvonal választási (routing) algoritmus szerint haladnak a hálózatban (szimuláció) és a hálózat elemei (vonalak, csomópontok) a rajtuk áthaladó statisztikákat összegzik (numerikus módszer)
2. Az összes statisztika átvitele után a hálózati elemek a véges kapacitásaiknak megfelelően transzformálják az összegzett statisztikákat

A TFA részletesebben

- A forgalom modell:
 - bitmennyiség-eloszlás (bit-throughput histogram) a vonalak számára
 - csomaggyakoriság-eloszlás (packet-throughput histogram) a csomópontok kapcsológépei számára
- Az összeadást konvolúcióval végezzük
- A véges kapacitások figyelembevétele egy iteratív algoritmussal történik
- Az eredmények: (bitmennyiség-eloszlás vagy csomaggyakoriság-eloszlás) és késleltetés-eloszlás

TFA kapacitáskorlátozási algoritmusának illusztrációja

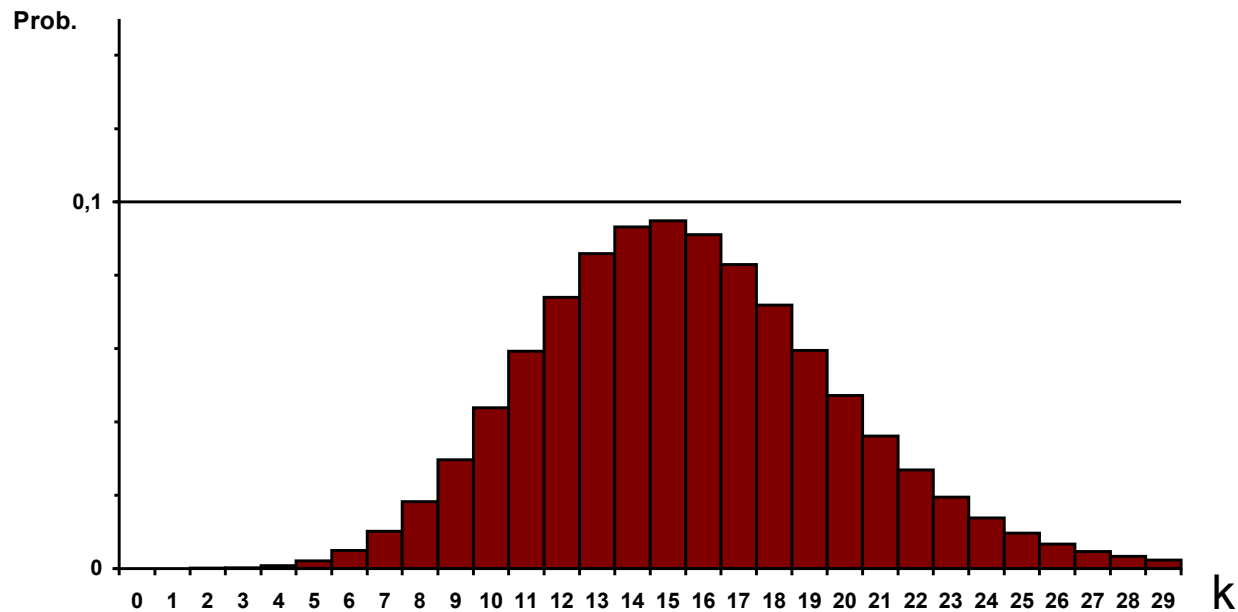
Annak a valószínűsége, hogy T idő alatt éppen k darab csomag érkezik a kapcsológéphez: $p_T[k]$.



A $p_T[k]$ eredeti sűrűségfüggvény

TFA kapacitáskorlátozási algoritmusának illusztrációja

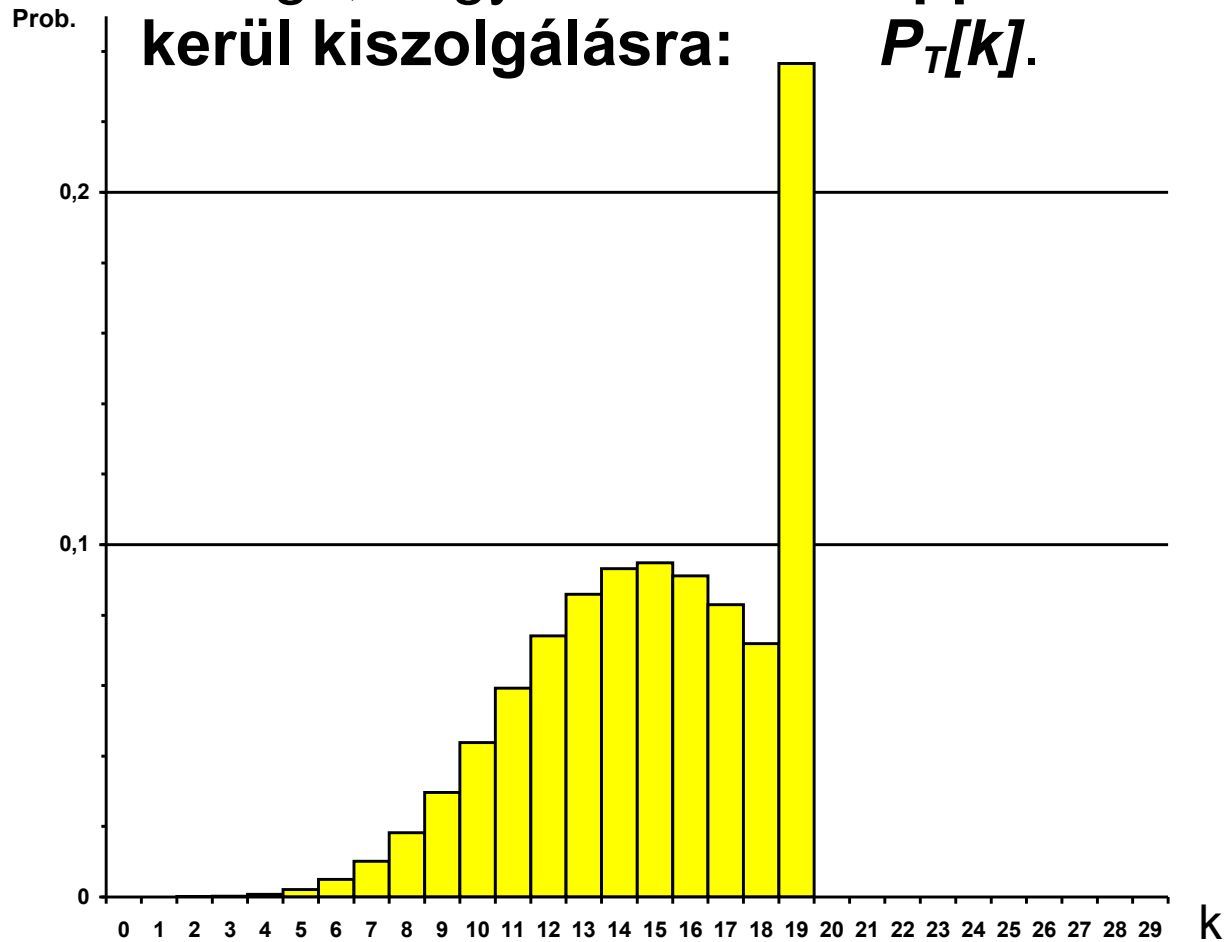
Annak a valószínűsége, hogy **állandósult állapotban éppen k darab csomag igényel kiszolgálást T idő alatt: $p_T^*[k]$.**



A konvolúciók végeredményeként kapott $p_T^*[k]$

TFA kapacitáskorlátozási algoritmusának illusztrációja

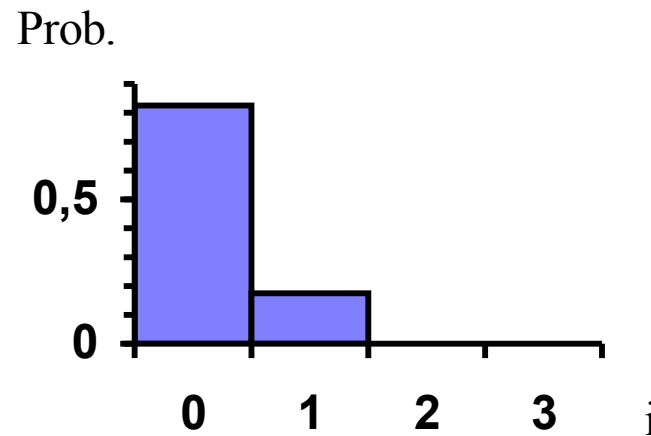
Annak a valószínűsége, hogy T idő alatt éppen k darab csomag kerül kiszolgálásra: $P_T[k]$.



A kapacitás korlátozás eredményeként kapott $P_T[k]$

TFA kapacitáskorlátozási algoritmusának illusztrációja

Annak a valószínűsége, hogy a csomagok iT időrésnyi késleltetést szenvednek: $D_T[i]$.



A $D_T[i]$ késleltetés eloszlás alakulása

A TFA és a DES kombinációja - I.

- Egy tipikus eset: kritikus részt tartalmazó hálózat, azaz:
 - adott egy hálózat, ami nagy számú elemből épül fel
 - van egy kritikus része, ami sokkal kevesebb elemből áll
- az események többsége a nem kritikus részben fordul elő, és ezek az események csak statisztikai jellemzőkön keresztül befolyásolják a kritikus rész viselkedését

A TFA és a DES kombinációja - II.

- Miért érdemes kombinálni őket?
 - Az egész rendszer (részletes) szimulációval (DES) való vizsgálata túl sok eseményt \Rightarrow kezelhetetlenül hosszú végrehajtási időt eredményezne
 - Az egész rendszer TFA-val való vizsgálata nem modellezné elég pontosan a kritikus részt
- Hogyan lehet őket kombinálni?
 - A hálózat kritikus részét (részletes) szimulációval vizsgáljuk (DES)
 - A hálózat többi (és sokkal nagyobb) részének vizsgálatára pedig TFA-t használunk

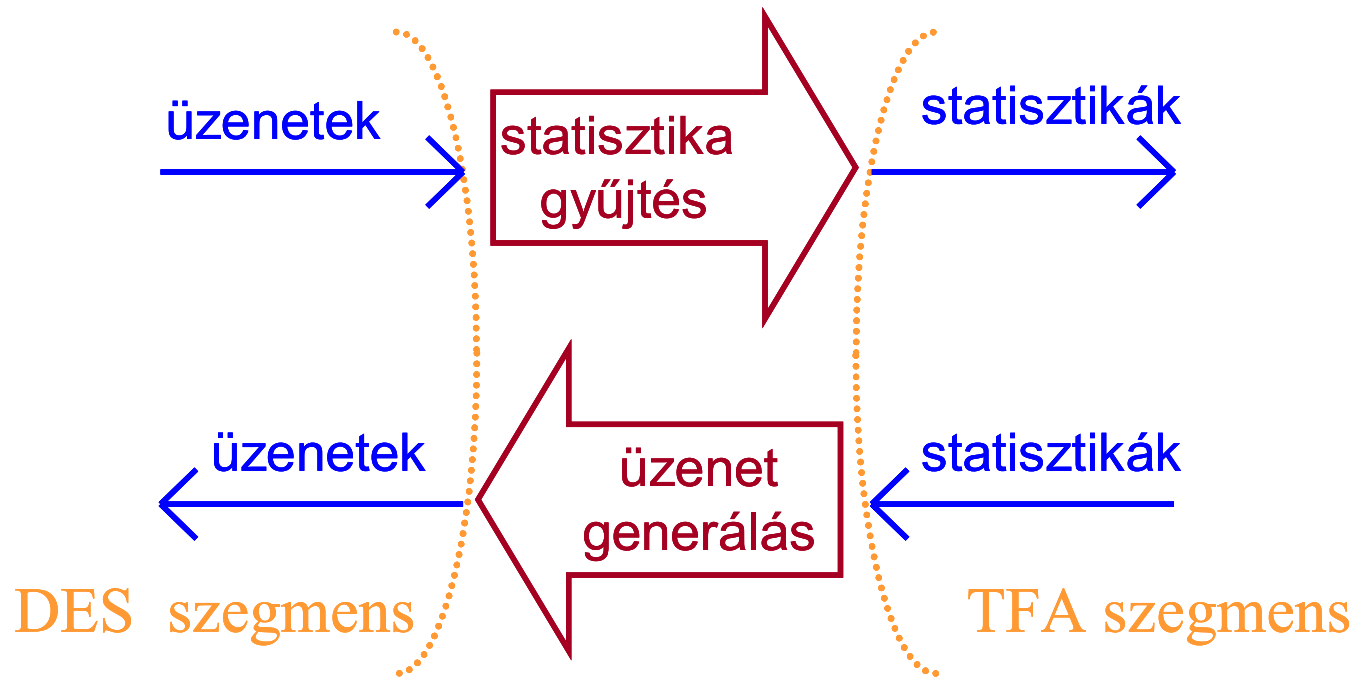
A TFA és a DES kombinációja - III.

- Megoldandó problémák:
 - A hálózat két részének forgalmi információt kell cserélnie. *A forgalmi információk cseréjéhez a kétféle forgalomreprezentáció közötti konverzióra van szükség!*
 - A TFA és a DES virtuális idő használata eltérő
 - TFA: pillanatfelvételt ad – virtuális idő csak egy paraméter
 - DES: virtuális időnek kulcs szerepe van – lásd az eseményvezérelt DES algoritmusát

A TFA és a DES együttműködéséhez az eltérő virtuális idő használatot kezelni kell!

Kétirányú konverzió a statisztikák és az üzenetek között

A megoldás alapötlete:



A TFA és a DES együttműködése

A TFA és a DES együttműködését a következőképpen definiáljuk (egy vagy több alkalommal az történik, hogy):

0. A DES beállítja a TFA virtuális idejét
1. A DES bemeneti paramétereket ad a TFA számára
2. A DES meghívja a TFA-t, hogy értékelje ki a rendszer rábízott részét
3. A TFA fut
4. A TFA paramétereket ad vissza a DES-nek

A TFA szegmens és a DES szegmens közötti kommunikáció módja

- A) A DES rész és a TFA rész két külön program, és valamilyen inter-processz kommunikációs (IPC) módszert használnak (pl. PVM/MPI, named pipe-ok).
- B) A TFA egy függvény a DES programon belül, amit néha meghívnak, de nem használja a DES kernel virtuális idejét saját belső céljaira.
- C) A TFA egy függvény(halmaz) a DES programon belül, amit néha meghívnak, és használja a DES kernel virtuális idejét (esemény mechanizmusát és egyéb szolgáltatásait) saját belső céljaira.

Az előnyei miatt "C"-vel foglalkozunk!

A TFA és a DES együttműködése a "C" kommunikációs megoldás használatával

1. A DES beállítja a TFA virtuális idejét a TFA indító eseményének a megfelelő virtuális időpontra való felidőzítésével. (Az esemény bekerül a FES-be.)
2. A DES a bemeneti paramétereket a TFA-nak a TFA-ért felelős modul(ok)nak való üzenetküldéssel adja át.
3. A DES egyáltalán nem hívja meg a TFA-t, akkor fut, ha elérkezik a virtuális ideje.
4. Amikor a TFA fut, használhatja az eseménykezelő rendszert. – A virtuális idő ellentmondásos kezelésének feloldása megtalálható: (Lencse, 2005)
5. A TFA a DES-ért felelős modul(ok)nak való üzenetküldéssel ad vissza paramétereket a DES-nek.

A kombinált DES és TFA párhuzamosítása

- Mit lehet párhuzamosítani?
- Hogyan lehet párhuzamosítani?
- Milyen gyorsulást várhatunk?

Mit lehet párhuzamosítani?

- Ötletek:
 - Párhuzamosítsuk a DES szegmens és a TFA szegmens futtatását!
 - Legyen több párhuzamosan futó DES szegmens!
 - Legyen több párhuzamosan futó TFA szegmens!

A DES és a TFA szegmens párhuzamos végrehajtása 2 processzoron

- Az eredeti kombinált DES+TFA kétirányú adatcserét használ
- A kombináció célja általában az, hogy a TFA rész segítségével megkapjuk a DES rész számára a megfelelő forgalmi terhelést (tehát a TFA → DES forgalomra van szükségünk)
- A párhuzamos kombinált (és együttműködő) DES+TFA (PCDT: Parallel Combined DES and TFA) előrejelzést használ a DES szegmens → TFA szegmens forgalomra

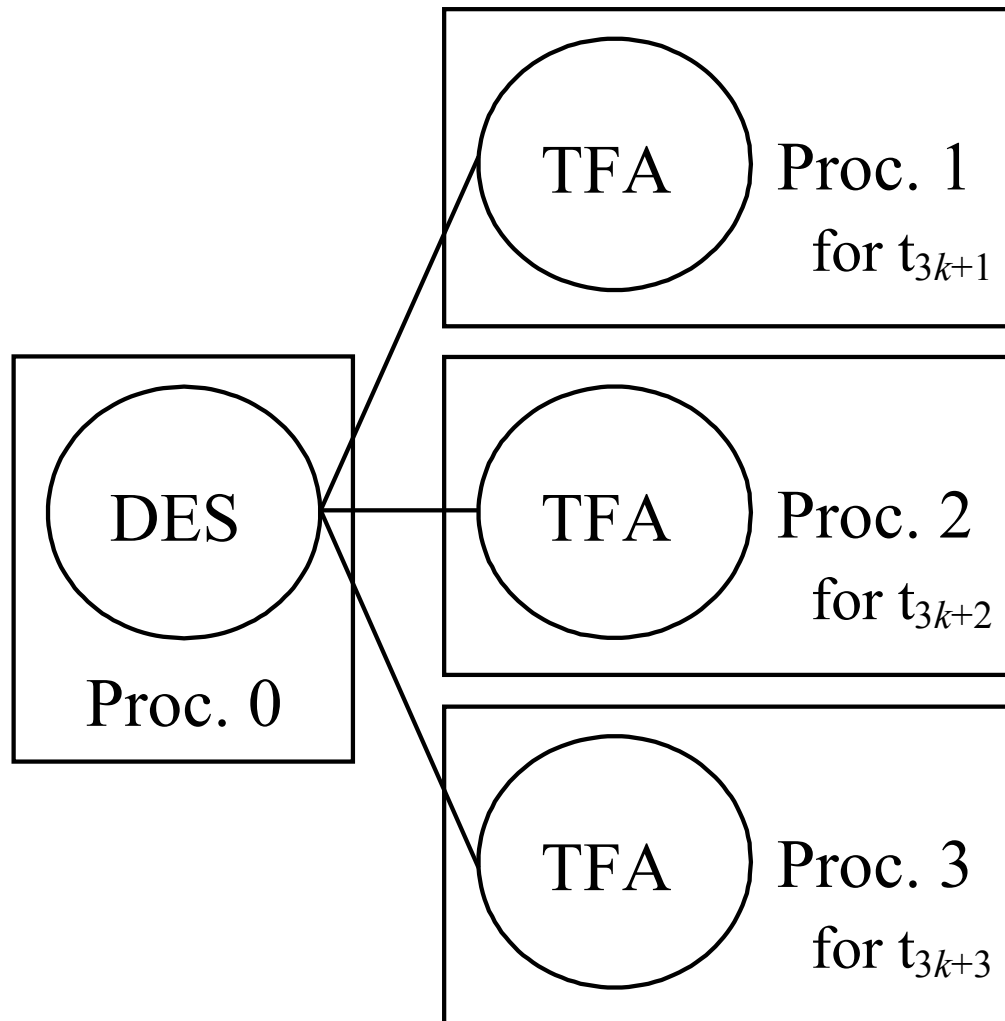
PCDT 2 processzoron

- Kezdetben
 - t_0 -kor: DES \rightarrow TFA : “TFA kérés” t_1 időbélyeggel és más paraméterekkel
 - t_0 -tól to t_1 -ig: a DES és a TFA szegmensek egymástól függetlenül futnak
- Általános lépés:
 - t_i -kor: TFA \rightarrow DES “TFA eredmény” (forgalom)
 - t_i -kor: DES \rightarrow TFA : “TFA kérés” t_{i+1} időbélyeggel és más paraméterekkel
 - t_i -től t_{i+1} -ig: a DES és a TFA szegmensek egymástól függetlenül futnak

PCDT n+1 processzoron

- Képzeljük el:
 - Ha például:
 - $TFA_szegmens_mérete = 100 * DES_szegmens_mérete$
 - $TFA_hatékonysága = 10 * DES_hatékonysága$
 - Akkor:
 - $TFA_sz._számításigénye = 10 * DES_sz._számításigénye$
- Ötlet:
 - Futtassunk párhuzamosan több TFA szegmenst több processzoron, amelyek **ugyanazt a hálózatot vizsgálják különböző t_i virtuális időpontokra!**

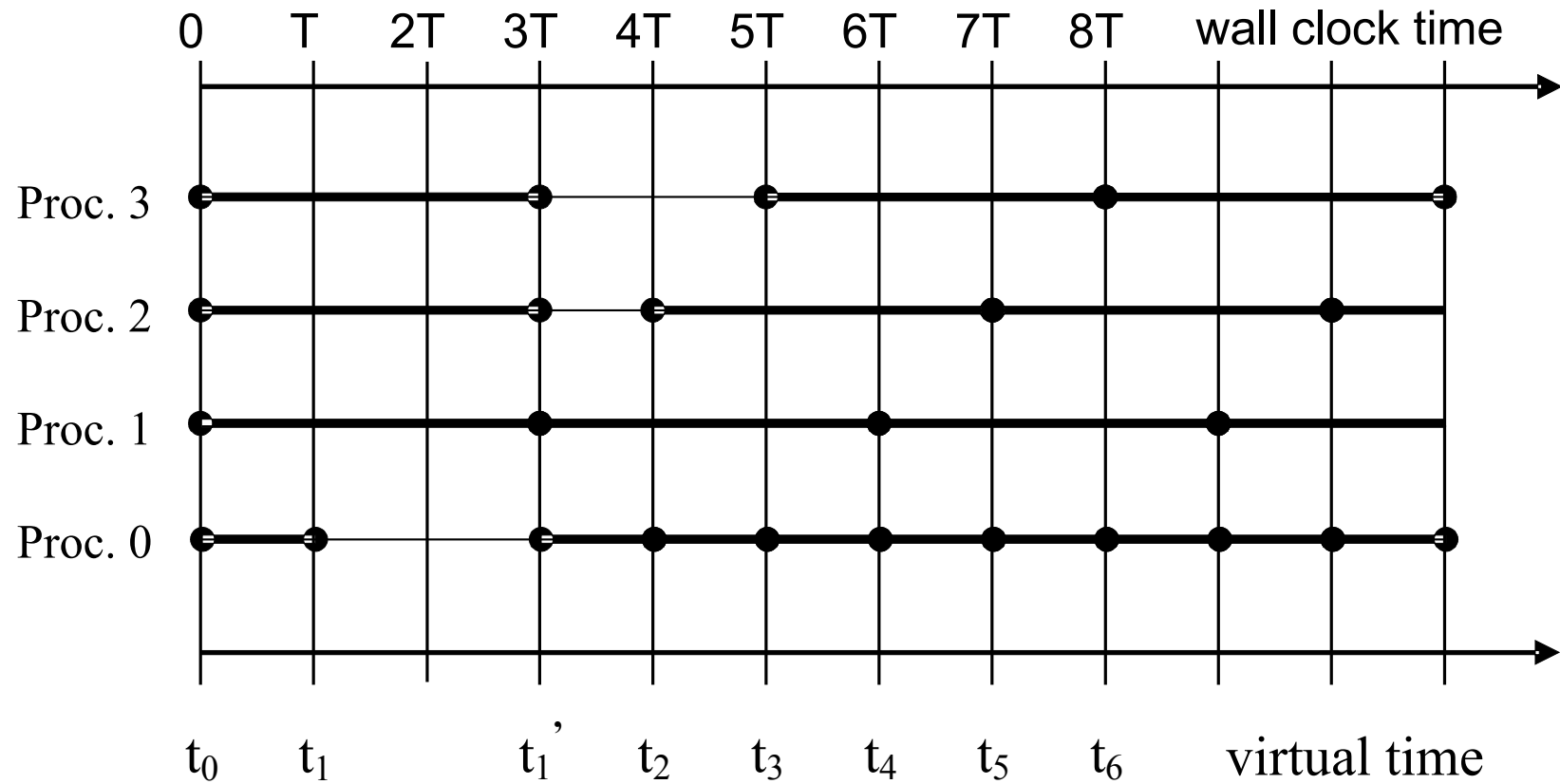
Példa: PCDT 3+1 processzoron



Figyelem!

Ugyanazt a TFA szegmenst hajtjuk végre párhuzamosan különböző t_i virtuális időpontokra!

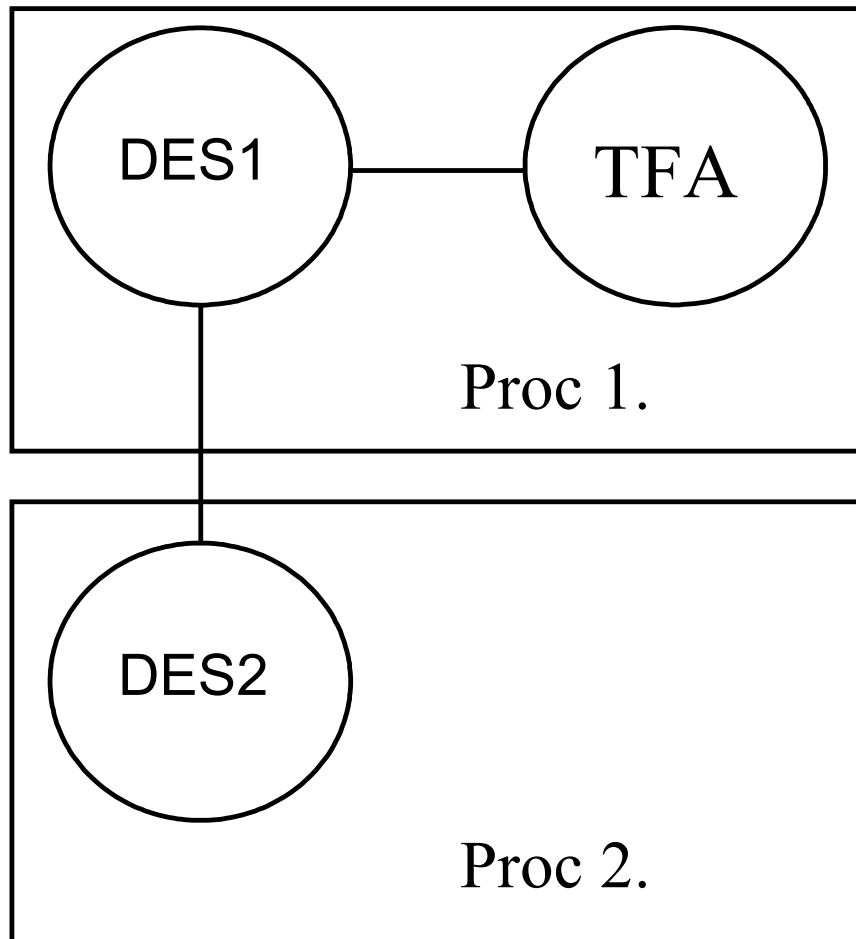
PCDT működése 3+1 processzoron



Az $n+1$ processzoros PCDT alkalmazhatósági kritériumai

- A t_i időpontbeli TFA eredményei jól jellemzik a TFA \rightarrow DES forgalmat a t_{i+1} virtuális időpontig. (ugyanaz, mint a nem párhuzamos esetre)
- A DES \rightarrow TFA forgalom vagy elhanyagolható, vagy t_i -kor t_{i+n} -ig előrejelezhető. (új kritérium)

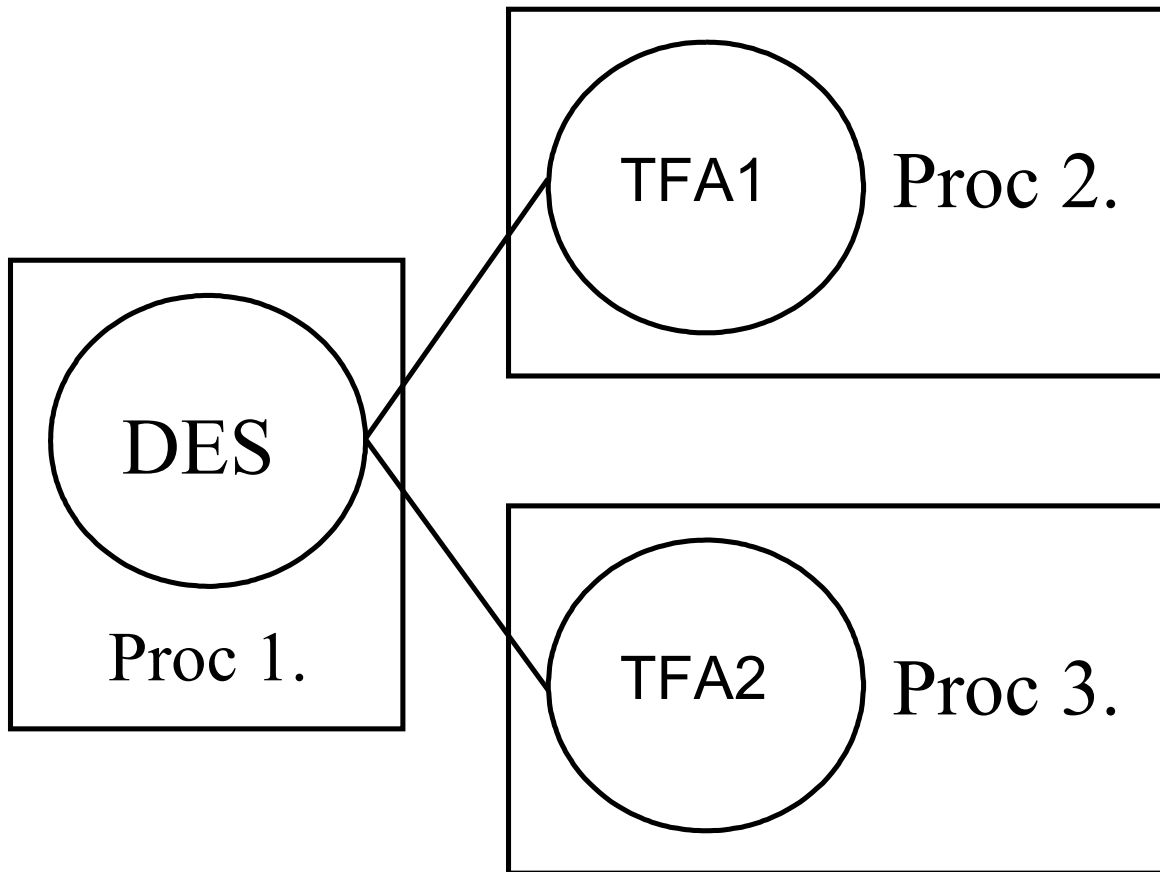
Több DES szegmens párhuzamos futtatása



Megjegyzés:

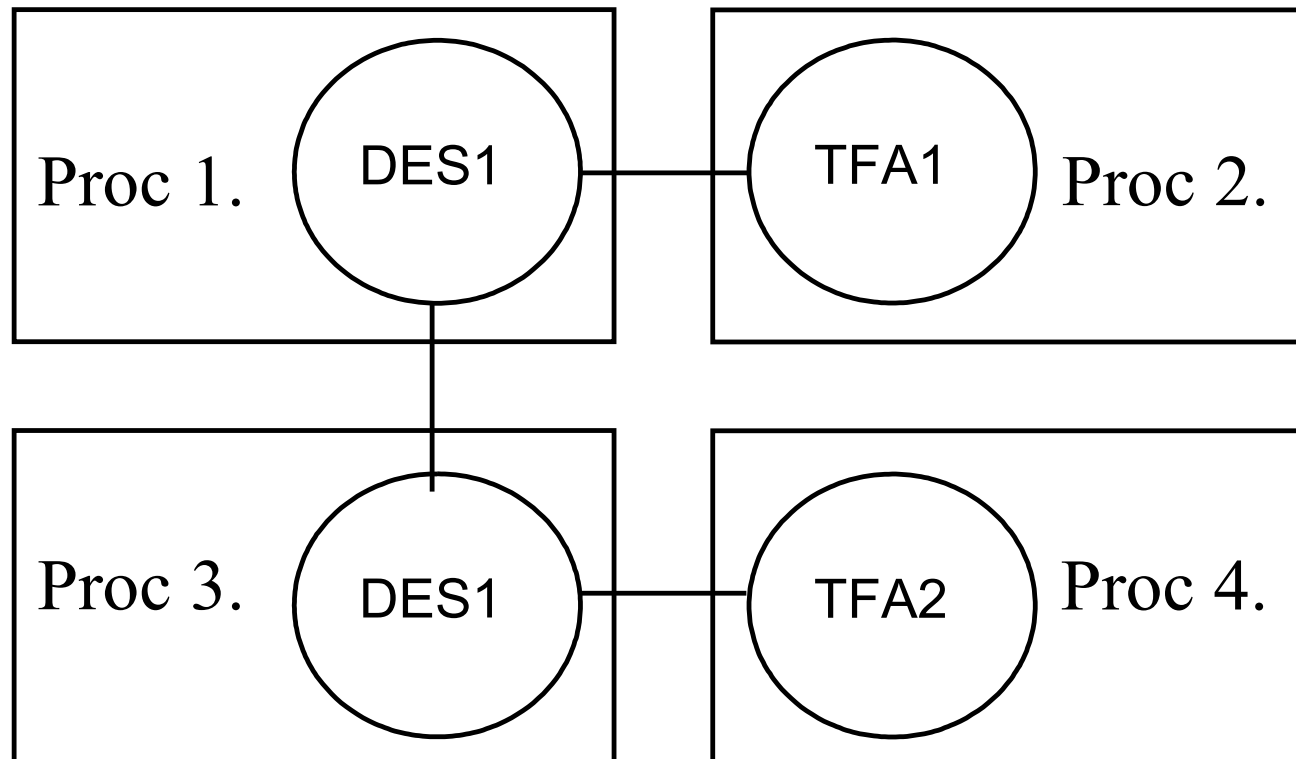
A kombinált (és együttműködő) DES és TFA rendszer DES szegmensét több (itt: két) részre bontva párhuzamosan futtatjuk több (2) processzoron (valamilyen PDES módszert alkalmazva)

Több független TFA szegmens párhuzamos futtatása



Megjegyzés:
TFA1 és
TFA2 között
nincs
forgalom!

Több DES és több **független** TFA szegmens párhuzamos futtatása



Megjegyzés:

TFA1 és TFA2 között **nincs forgalom!**

Referenciák - I.

- Fujimoto, R. M. 1990. Parallel Discrete Event Simulation. *Communications of the ACM* **33** no. 10, 31-53
- Lencse, G. 1997. "Efficient Simulation of Large Systems - Transient Behaviour and Accuracy" *Proceedings of the 1997 European Simulation Symposium (ESS'97)* (Passau, Germany, Oct. 19-23). SCS Europe, 660-665.
- Lencse, G. 1998a. "Efficient Parallel Simulation with the Statistical Synchronization Method" *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation (CNDS'98)* (San Diego, CA. Jan. 11-14). SCS International, 3-8.
- Lencse, G. 1998b. "Statistics Collection for the Statistical Synchronisation Method" *Proceedings of the 1998 European Simulation Symposium (ESS'98)* (Nottingham, UK. Oct. 26-28). SCS Europe, 46-51.
- Lencse, G. 1999a. "Applicability Criteria of the Statistical Synchronization Method" *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation (CNDS'99)* (San Francisco, CA. Jan. 17-20). SCS International, 159-164.

Referenciák - II.

- Lencse, G. 1999b. "Design Criterion for the Statistics Exchange Control Algorithm used in the Statistical Synchronization Method" *Proceedings of the Advanced Simulation Technologies Conference (ASTC 1999)* part of the 32nd Annual Simulation Symposium, April 11-15 1999, San Diego, CA, USA (pp. 138-144).
- Lencse Gábor 2000. "*Kommunikációs rendszerek hatékony szimulációjának egyes kérdései*" Ph.D. értekezés (magyar nyelven), BME Híradástechnikai Tanszék, nyilvános vita: 2001. május 9.
- Lencse, G. 2001. "Traffic-Flow Analysis for Fast Performance Estimation of Communication Systems" *Journal of Computing and Information Technology* **9**, No. 1, 15-27.
- Lencse, G. 2002. "Parallel Simulation with OMNeT++ using the Statistical Synchronization Method" *Proceedings of the 2nd International OMNeT++ Workshop* (Jan. 8-9, 2002, Technical University Berlin, Berlin, Germany) 24-32.

Referenciák - III.

- Lencse, G. 2004. "Combination and Interworking of Traffic-Flow Analysis and Event-Driven Discrete Event Simulation" *Proceedings of the 2004 European Simulation and Modelling Conference (ESM[®]'2004)* (Paris, France, Oct. 25-27.) EUROSIS-ETI, 89-93.
- Lencse, G. 2005. "Speeding up the Performance Analysis of Communication Systems" *Proceedings of the 2005 European Simulation and Modelling Conference (ESM[®]'2005)* (Porto, Portugal, Oct. 24-26.) EUROSIS-ETI, 329-333.
- Lencse, G; L. Muka 2006. "Convergence of the Key Algorithm of Traffic-Flow Analysis" *Journal of Computing and Information Technology*, **14**, No. 2, 133-140.
- Pongor, Gy. 1992. "Statistical Synchronization: a Different Approach of Parallel Discrete Event Simulation". *Proceedings of the 1992 European Simulation Symposium (ESS'92)* (Nov. 5-8, 1992, The Blockhaus, Dresden, Germany.) SCS Europe, 125-129.