

Operációs rendszerek

ELTE IK.

Dr. Illés Zoltán

zoltan.illes@elte.hu

Miről beszéltünk korábban...

- ▶ Operációs rendszerek kialakulása
- ▶ Op. Rendszer fogalmak, struktúrák
- ▶ Fájlok, könyvtárak, fájlrendszerek
- ▶ Folyamatok
 - Folyamatok kommunikációja
 - Kritikus szekciók, szemaforok.
- ▶ Klasszikus IPC problémák
- ▶ Ütemezés

Mi következik ma...

- ▶ **Beviteli/ Kiviteli eszközök vezérlése**
 - I/O eszköztípusok
 - Eszközök elérése
 - Megszakítás
 - DMA
 - I/O portok
- ▶ **I/O programok alapelvei**
- ▶ **Erőforrás elérés problémái**
 - Holtpontok
 - Alapelvek
 - Felismerés, megelőzés, elkerülés

Input-Output eszközök

▶ Blokkos eszközök

- Adott méretű blokkban tároljuk az információt.
- Blokkméret 512 byte– 32768 byte között.
- Egymástól függetlenül írhatók vagy olvashatók.
- Blokkonként címezhető
- Ilyen eszköz: HDD, CD, szalagos egység, stb

▶ Karakteres eszközök

- Nem címezhető, csak jönnek–mennek sorban a „karakterek” (bájtok)

▶ Időzítő: kivétel, nem blokkos és nem karakteres

I/O eszközök modellje

- ▶ Bár mindig van kivétel (timer), az op.rendszer eszköz független szoftvermodellje erre a blokkos–karakteres modellre épül.
 - Pl. A fájlrendszer absztrakt blokkos eszközökkel foglalkozik.
 - Szalagegység is blokkos, az N. blokk olvasása parancsnál, előbb visszateker, majd előre.
 - Az eszközfüggő részt az eszközmeghajtók (device driver) jelentik. (DDK)

I/O eszközök sebessége

- ▶ Billentyűzet: 10 bájt/sec
- ▶ Egér: 100 bájt/sec
- ▶ 56k modem: 7kbájt/sec
- ▶ Szkenner: 400kbájt/sec
- ▶ 52xCD ROM: 8 MB/sec (1xCD, 150kb/sec)
- ▶ Firewire : 50 MB/sec
- ▶ USB2: 60 MB/sec
 - USB3 500 MB/sec (elméleti, 4.8GBPS)
- ▶ SATA: 200 MB/sec
- ▶ SCSI UW4: 320 MB/sec
- ▶ PCI sín: 528 MB/sec

Eszközvezérlők

- ▶ Az I/O eszközt a számítógéphez (rendszer busz-hoz) kapcsoló elem az eszközvezérlő, vagy adapter.
 - Video vezérlő
 - Soros-párhuzamos vezérlő
 - USB vezérlő
 - HDD vezérlő
 - IDE, SATA, SCSI
- ▶ Nagy gépek esetén az I/O speciális I/O gépekkel van megvalósítva.
- ▶ CPU-eszközvezérlő kommunikációja
 - Memórialeképezésű I/O, Megszakítás, DMA

I/O kapu, memórialeképezésű I/O

- ▶ A CPU külvilág felé kétféle adatcserét ismer
 - I/O kapuk írása olvasása
 - IN regiszter, port ; A port (8 vagy 16 bites szám) ; adatának regiszterbe olvasása
 - OUT port, regiszter ; A regiszter kiírása a portra.
 - Az I/O eszköz regiszterei, adatterülete a memória egy részén helyezkedik el.
- ▶ Létezhet olyan környezet, ahol
 - Csak I/O kapukat használnak. (IBM 360)
 - Csak memóriában vannak az I/O kapuk (PDP-11)
 - Memórialeképezésű I/O
 - Vegyes (Intel x86, Pentium)

Megszakítások I.

- ▶ Interrupt – fogalma az I/O eszközökhöz kötődik!
 - Ha egy I/O eszköz „adatközlésre kész”, ezt egy megszakításkéréssel jelzi!
- ▶ Szoftveres vs. Hardveres megszakítás
 - Szoftveres: gépi kódú utasítás (int x) végzi, multitask esetén „trap-nak” (csapda) hívják!
- ▶ Megszakításnak száma van!
 - Megszakítás vektor (valós mód, 0 címtől)
 - Megszakítás kiszolgáló rutin.
 - INTR, NMI

Megszakítások kezelése

- ▶ Általában az eszközöknek van állapotbitjük, jelezve, hogy az adat készen van.
 - Ezt lehet figyelni, nem az igazi.
 - Tevékeny várakozás ez is, nem hatékony, ritkán használt.
- ▶ Megszakítás kezelés folyamata (IRQ)
 - A HW eszköz jelzi a megszakítás igényt(INTR)
 - CPU egy következő utasítás végrehajtás előtt, a tevékenységét megszakítja! (Precíz, imprecíz)
 - A kért sorszámú kiszolgáló végrehajtása.
 - A kívánt adat beolvasása, a szorosan hozzátartozó tevékenység elvégzése.
 - Visszatérés a megszakítás előtti állapothoz.

Megszakítások prioritása

- ▶ INTR – maszkolható, NMI nem maszkolható
 - INTR megszakítás prioritások
 - Megszakítás közben érkező hasonló vagy alacsonyabb prioritású kérés várakozik!
 - NMI – csak egy kiszolgálás, legnagyobb prioritás
- ▶ PC világban 15 különböző interrupt
 - 2x8 csatornás vezérlő
 - Kézi adapter IRQ állítás korábban, jumper-rel, sw-el
 - BIOS automatikus megszakítás hozzárendelés (Plug&Play)

Közvetlen memória elérés (DMA)

► Direct Memory Access

- Tartalmaz: Memória cím regisztert, átvitel irány jelzésre, mennyiségre, vezérlésre regisztert
- Ezeket szabályos in, out portokon lehet elérni.

► Működés jellemző lépései:

1. CPU beállítja a DMA vezérlőt. (Regisztereket.)
2. A DMA a lemezvezérlőt kéri a megadott műveletre.
3. Miután a lemezvezérlő beolvasta a pufferébe, a rendszersínen keresztül a memóriába(ból) írja, olvassa az adatot.
4. Lemezvezérlő nyugtázza, hogy kész a kérés teljesítése.
5. DMA megszakítással jelzi, befejezte a műveletet.

I/O szoftver célok I.

► Eszközfüggetlenség

- Ugyanaz a kód, parancs legyen képes pl. fájlt olvasni HDD-ről, CD-ről stb.
- Egységes névhasználat, a név mint paraméter jelenti (eltakarja) a valódi eszközt.
- Logikai csatolás (mount)
 - Unix: Floppy csatolva a /home/fdd könyvtárhoz
 - Windows: Floppy csatolva az a: névhez

► Hibakezelés

- Hardver közeli szinten kell(ene) kezelni
- Az esetek többségében itt elvégezhető, magasabb szintre csak „fatális” hiba esetén kerüljön a kezelés.

I/O szoftver célok II.

- ▶ Szinkron (blokkolós), aszinkron (megszakításos) átviteli mód támogatása.
 - Szinkron módban könnyebb írni a felhasználói programokat.
 - Kell támogatnia az op. Rendszernek az aszinkron mód használatot is.
- ▶ Pufferezés
 - Mindenki használja, billentyű puffer stb.
- ▶ Eszközök megosztott, vagy egyedi használat
 - Lemezt egyszerre több folyamat is tudja használni.
 - Egyedi (monopol) használatú pl. a szalagegység, CD író.
- ▶ DDK– Device Driver Kit

I/O szoftverrendszer felépítése

- ▶ Réteges szerkezet
 - Tipikusan 4 rétegbe van szervezve.
- ▶ Hardver eszköz
 1. Megszakítás kezelő réteg
 - Legalsó kernel szinten kezelt.
 - Szemafor blokkolással védve a kritikus (egész?) rész.
 2. Eszközmeghajtó programok
 3. Eszköz független operációs rendszer program
 4. Felhasználói I/O eszközt használó program

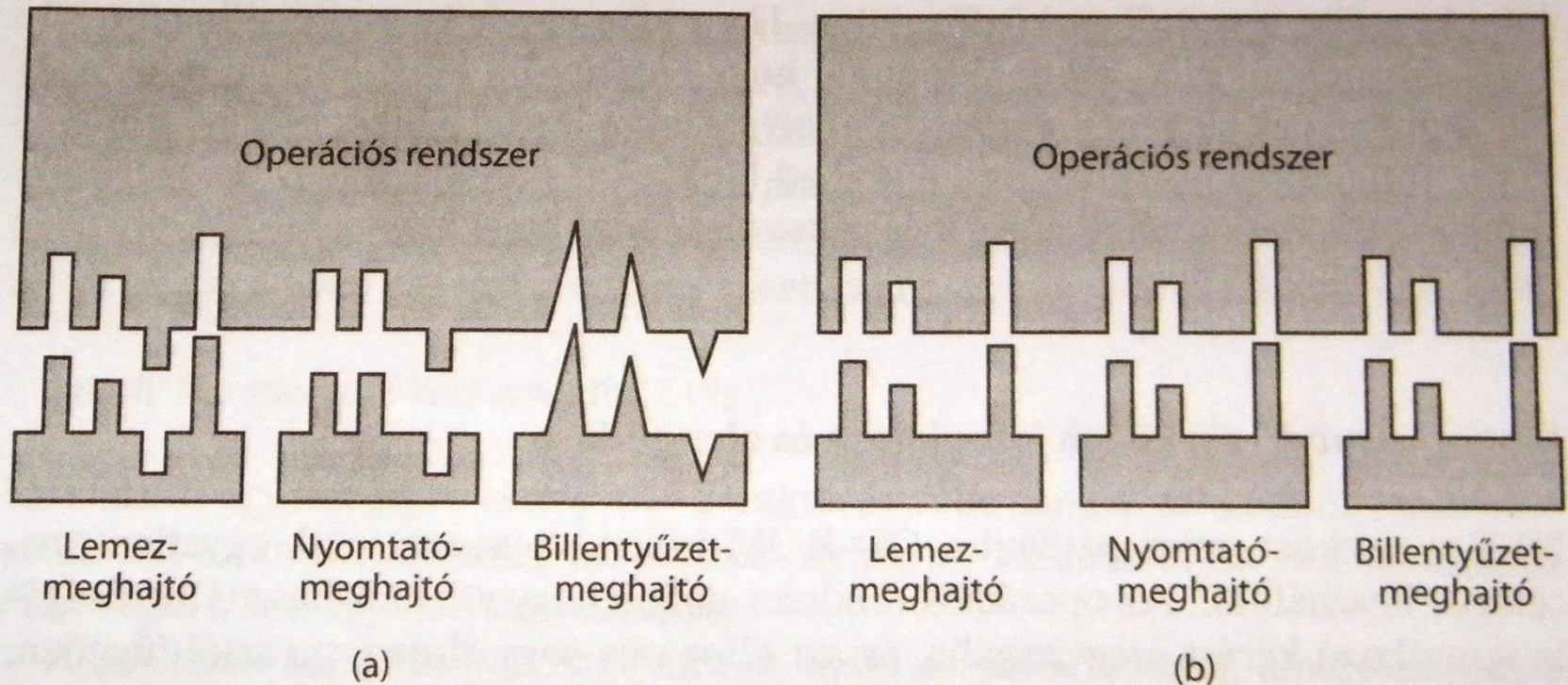
Eszközmeghajtó programok

- ▶ Eszköz specifikus kód – eszközmeghajtó program (driver)
- ▶ Ez pontosan ismeri az eszköz jellemzőit
 - Egérmegható megmondja az elmozdulást, melyik gombot nyomták meg.
 - Lemezmeghajtó ismeri a fejek mozgatását, beállítását adott sávhoz, szektorhoz.
 - Stb.
- ▶ Feladata a felette lévő szintről érkező absztrakt kérések kiszolgálása
 - Egyszerűbb eszközök, egy időben egy kérés
 - Intelligensebbek, kérések sorát fogadhatja (scsi)
- ▶ Kezeli az eszközt I/O portokon, megszakítás kezelésén keresztül.
- ▶ Blokkos – karakteres eszközök

Eszköz független I/O program feladata I.

- ▶ Egységes, szabványos kapcsolódási felület biztosítása.
 - Azonos elnevezések, hívási konvenciók.
 - I/O eszközök szimbolikus neveinek valós meghajtókhoz kapcsolása.
 - Főeszköz szám : a meghajtóra utaló azonosító
 - Mellékeszköz szám: + paraméter, pl. írás-olvasás jelzésre
 - I/O eszköz védelme
 - Fájrendszer szerű jogosultságok alkalmazása.

Egységes felület illusztráció.



3.7. ábra. (a) Szabványos meghajtó interfész nélkül. (b) Szabványos meghajtó interfésszel

Eszköz független I/O program feladatai II.

- ▶ Pufferezés, hasonlóan az adapterekhez, általános elv, a felhasználó független az eszköz független I/O programtól. (Gyorsabban ír a pufferbe mint az „eszközbe”.)
- ▶ Hibakezelés
- ▶ Monopol módú eszközök lefoglalása, elengedése
- ▶ Eszköz független blokkméret kialakítása.
 - Lemezek logikai blokkmérete

Felhasználói I/O programok

- ▶ Könyvtári I/O eljárások 2 kategóriája
 - Továbbítja a paramétereket a rendszerhívás számára
 - `N=write(fd, buffer, db);`
 - Tényleges feladatot (is) végeznek, majd rendszerhívás
 - `printf(„%d almafa”, n);`
- ▶ Háttértárolás (spooling)
 - Monopol eszközök kezelési módja (nyomtató)
 - Speciális folyamatok kezelik a háttértár, spooling könyvtárakat. Démonok.

Monopol módú erőforrások használata

- ▶ Láttuk korábban, ezek az I/O eszközök egyik fontos csoportja.
- ▶ De ilyen a rendszer belső táblázat (pl folyamat tábla) kezelése is.
- ▶ Egy időben csak egy folyamat használhatja.
 - Pl: „Párhuzamosan” 2 fájlt nyomtatni nem az igazi.
- ▶ Nem csak monopol I/O eszköznél fordulhat elő versenyhelyzet
 - Egyszerű memória rekesznél is, de jellemzően a monopol I/O eszközökhöz kötődik.
- ▶ Tipikus helyzet: két folyamat ugyanarra vár.
 - Két udvarias ember a lift előtt...lift elmegy ők maradnak...

Holtpont (deadlock)

- ▶ Két vagy több folyamat egy erőforrás megszerzése során olyan helyzetbe kerül, hogy egymást blokkolják a további végrehajtásban.
 - Pontos definíció: Folyamatokból álló halmaz holtpontban van, ha minden folyamat olyan eseményre vár, amit csak a halmaz egy másik folyamata okozhat.
- ▶ Nem csak az I/O eszközökhöz kötődik
 - Párhuzamos rendszerek
 - Adatbázisok
 - Stb.

Holtpont feltételek

- ▶ Coffman E.G., szerint 4 feltétel szükséges a kialakuláshoz
 1. Kölcsönös kizárás feltétel. Minden erőforrás hozzá van rendelve 1 folyamathoz vagy szabad.
 2. Birtoklás és várakozás feltétel. Korábban kapott erőforrást birtokló folyamat kérhet újabbat.
 3. Megszakíthatatlanság feltétel. Nem lehet egy folyamattól elvenni az erőforrást, csak a folyamat engedheti el.
 4. Ciklikus várakozás feltétel. Két vagy több folyamatlánc kialakulása, amiben minden folyamat olyan erőforrásra vár, amit egy másik tart fogva.

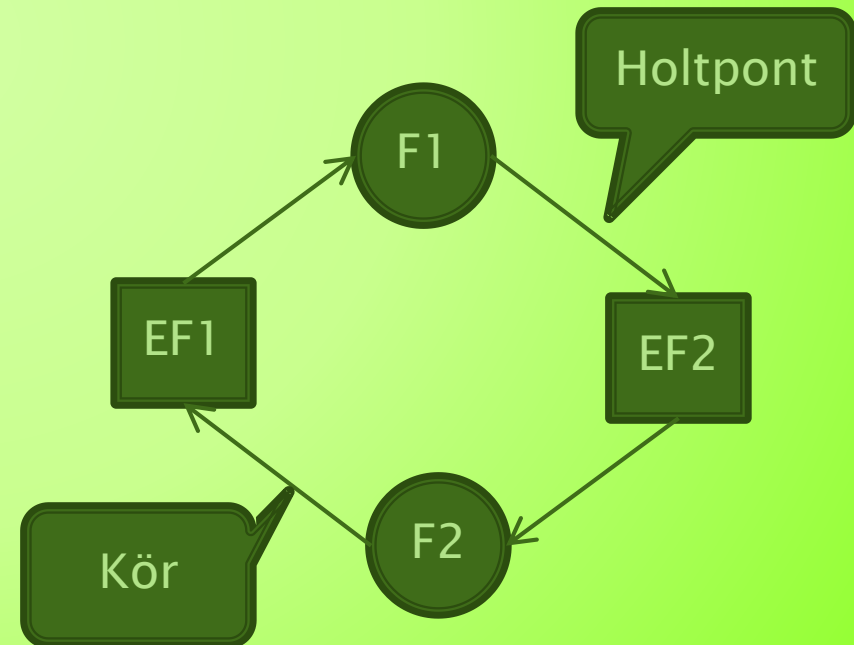
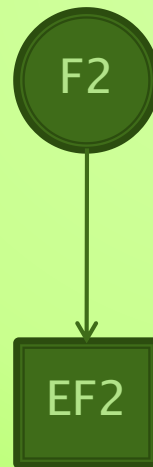
Holtpont gráfmodellje

- ▶ Holtpont feltételek modellezése irányított gráfokkal (Holt, 1972)
 - Folyamat– kör
 - Erőforrás– négyzet
 - Ha az erőforrások, folyamatok irányított gráfjában kört találunk, ez holtpontot jelent.

Erőforrás birtoklás



Erőforrás kérés



Holtpont kialakulása, példa

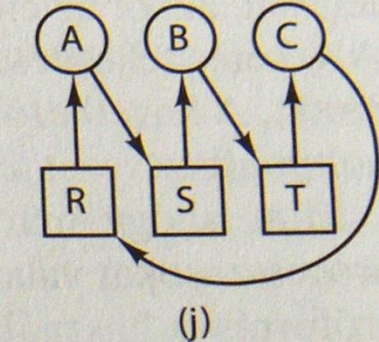
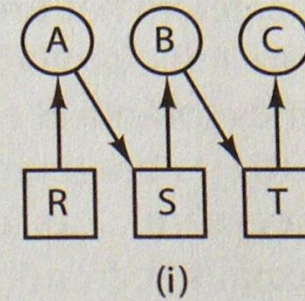
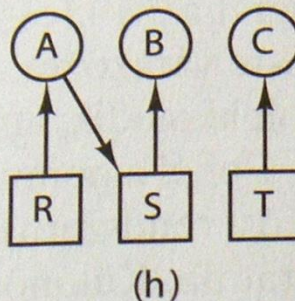
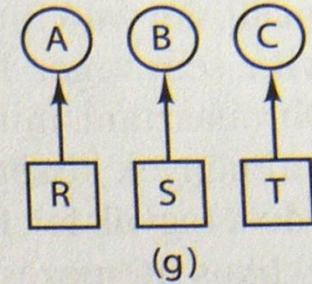
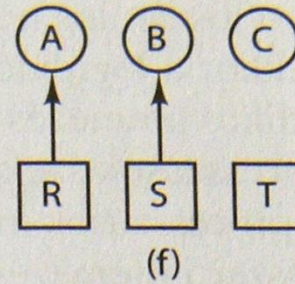
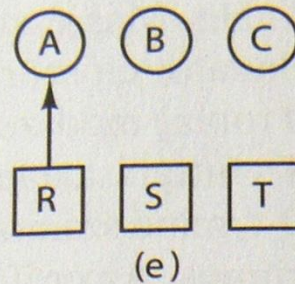
A
R kérése
S kérése
R elengedése
S elengedése
(a)

B
S kérése
T kérése
S elengedése
T elengedése
(b)

C
T kérése
R kérése
T elengedése
R elengedése
(c)

1. A kéri R-t
2. B kéri S-t
3. C kéri T-t
4. A kéri S-t
5. B kéri T-t
6. C kéri R-t

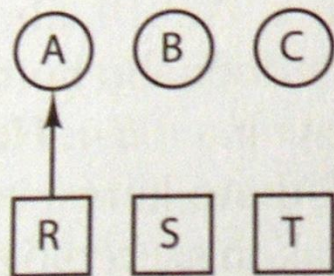
(d)



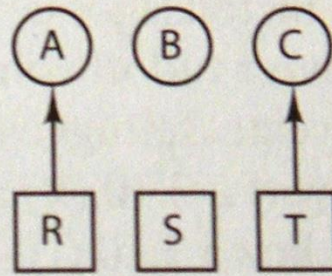
Holtpont elkerülése, példa

1. A kéri R-t
 2. C kéri T-t
 3. A kéri S-t
 4. C kéri R-t
 5. A elengedi R-t
 6. A elengedi S-t
- nincs holtpont

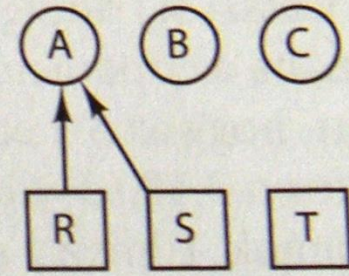
(k)



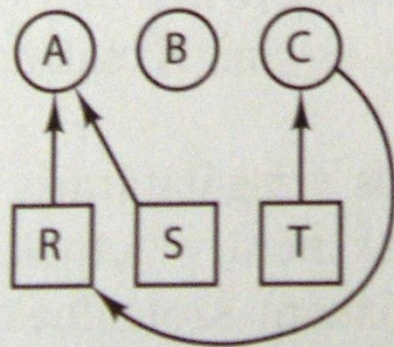
(l)



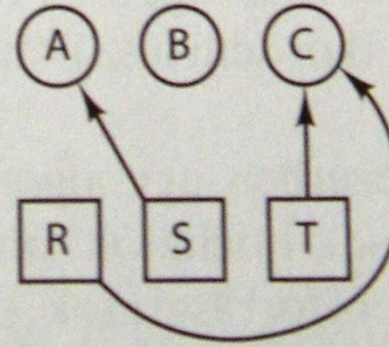
(m)



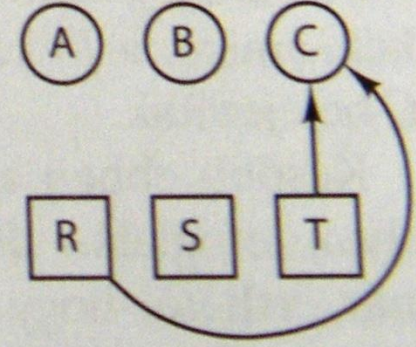
(n)



(o)



(p)



(q)

Holtpont stratégiák

1. A probléma figyelmen kívül hagyása.
 - Nem törődünk vele, nagy valószínűséggel Ő sem talál meg bennünket, ha mégis ...
2. Felismerés és helyreállítás.
 - Engedjük a holtpontot megjelenni (kör), ezt észrevesszük és cselekszünk.
3. Megelőzés. A 4 szükséges feltétel egyikének meghiusítása.
4. Dinamikus elkerülés. Erőforrások foglalása csak „óvatosan”.

1. Probléma figyelmen kívül hagyása

- ▶ Ezt a módszert gyakran strucc algoritmus néven is ismerjük.
- ▶ Kérdés, mit is jelent ez, és milyen gyakori probléma?
- ▶ Vizsgálatok szerint a holtpont probléma és az egyéb (fordító, op.rendszer, hw, sw hiba) összeomlások aránya 1:250.
- ▶ A Unix, Windows világ is ezt a „módszert” használja.
 - Túl nagy az ár a várható haszonért cserébe.

2. Felismerés, helyreállítás

- ▶ Folyamatosan figyeljük az erőforrás igényeket, elengedéseket.
- ▶ Kezeljük az erőforrás gráfot folyamatosan.
 - Ha kör keletkezik, akkor egy körbeli folyamatot megszüntetünk.
- ▶ Másik módszer, nem foglalkozunk az erőforrás gráffal, ha x (fél óra?) ideje blokkolt egy folyamat, egyszerűen megszüntetjük.
 - Nagygépes rendszereknél ismert módszer.

3. Megelőzés

- ▶ A Coffman féle 4 feltétel valamelyikére mindig él egy megszorítás.
 - Kölcsönös kizárás. Ha egyetlen erőforrás soha nincs kizárólag 1 folyamathoz rendelve, akkor nincs holtpont se!
 - De ez nehézkes, míg pl. nyomtató használatnál a nyomtató démon megoldja a problémát, de ugyanitt a nyomtató puffer egy lemezterület, itt már kialakulhat holtpont.
 - Ha nem lehet olyan helyzet, hogy erőforrásokat birtokló folyamat további erőforrásra várjon, akkor szintén nincs holtpont. Ezt kétféle módon érhetjük el.
 - Előre kell tudni egy folyamat összes erőforrásigényét.
 - Ha erőforrást akar egy folyamat, először engedje el az összes birtokoltat.

3. Megelőzés (folyt.)

- ▶ A Coffman féle harmadik feltétel a megszakíthatatlanság. Ennek elkerülése eléggé nehéz.
 - Nyomtatás közben nem szerencsés a nyomtatót másnak adni.
- ▶ Negyedik feltétel a ciklikus várakozás már könnyebben megszüntethető.
 - Egyszerű mód: Minden folyamat egyszerre csak 1 erőforrást birtokolhat.
 - Másik módszer: Sorszámozzuk az erőforrásokat, és a folyamatok csak ezen sorrendben kérhetik az erőforrásokat.
 - Ez jó elkerülési mód, csak megfelelő sorrend nincs!

4. Dinamikus elkerülés

- ▶ Van olyan módszer amivel elkerülhetjük a holtponthoz?
- Igen, ha bizonyos információ (erőforrás) előre ismert.
- ▶ Bankár algoritmus (Dijkstra, 1965)
 - Mint a kisvárosi bankár hitelezési gyakorlata.
- ▶ Biztonságos állapotok, olyan helyzetek, melyekből létezik olyan kezdődő állapotsorozat, melynek eredményeként mindegyik folyamat megkapja a kívánt erőforrásokat és befejeződik!
- ▶ A bankár algoritmus minden kérés megjelenésekor azt nézi, hogy a kérés teljesítése biztonságos állapothoz vezet-e?
 - Ha igen jóváhagyja, ha nem a kérést elhalasztja.
 - Eredetileg 1 erőforrásra tervezett.

Bankár algoritmus mintaállapotok (Egy erőforrásra)

	Birtokol	Maximum
A	0	6
B	0	5
C	0	4
D	0	7

Szabad: 10

(a)

	Birtokol	Maximum
A	1	6
B	1	5
C	2	4
D	4	7

Szabad: 2

(b)

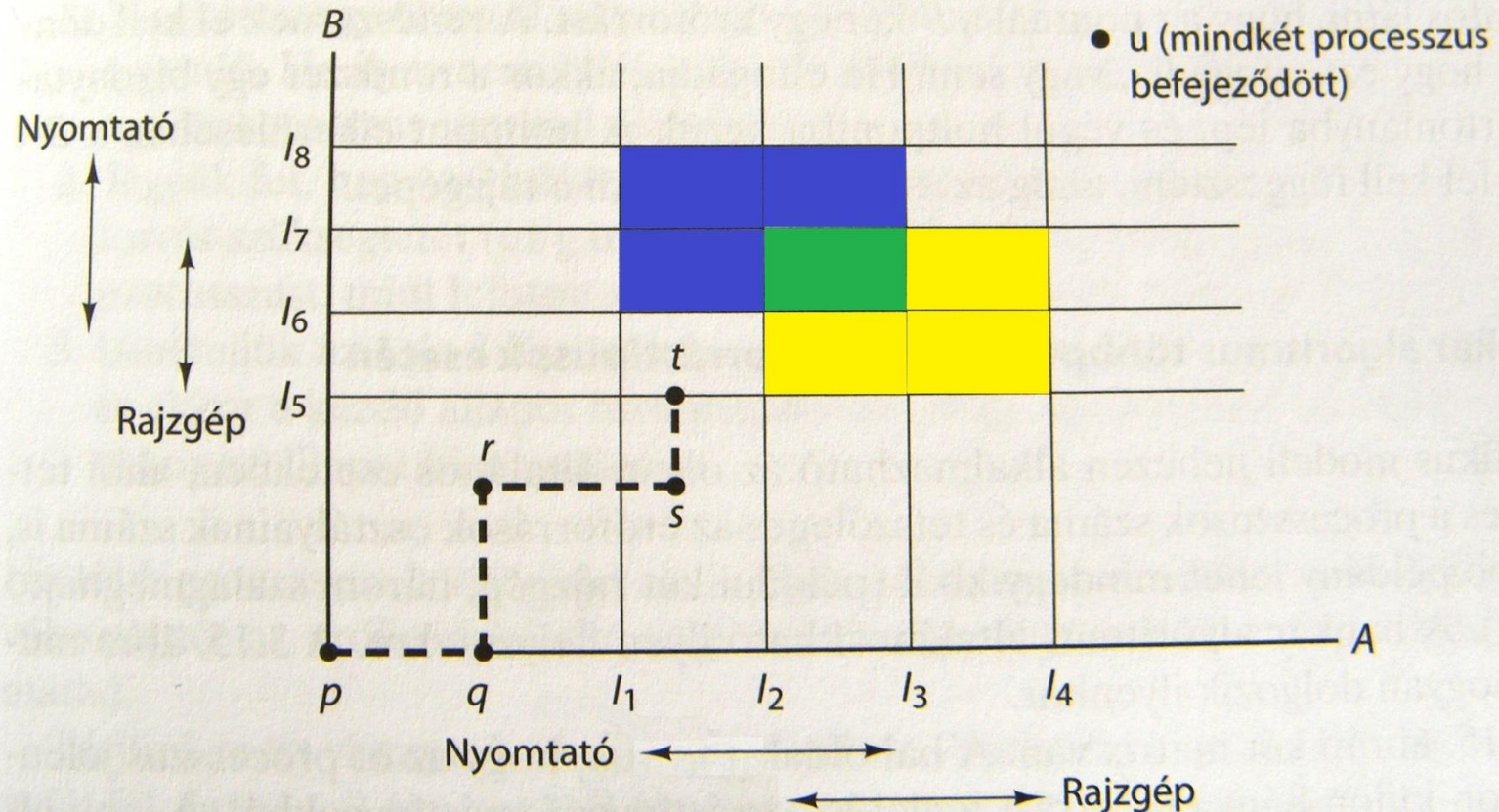
	Birtokol	Maximum
A	1	6
B	2	5
C	2	4
D	4	7

Szabad: 1

(c)

3.13. ábra. Három erőforrás-lefoglalási állapot. (a) Biztonságos. (b) Biztonságos.
(c) Bizonytalan

Két folyamat– erőforrás pályagörbe



3.14. ábra. Kétprocesszusú erőforrás-pályagörbék

Bankár algoritmus több erőforrás típus esetén

- ▶ Az 1 erőforrás elvet alkalmazzuk:
 - Jelölés: $F(i,j)$ az i . folyamat j . erőforrás aktuális foglalása
 - $M(i,j)$ az i . folyamat j . erőforrásra még fennálló igénye
 - $E(j)$, a rendelkezésre álló összes erőforrás.
 - $S(j)$, a rendelkezésre álló szabad erőforrás.
- 1. Keressünk i sort, hogy $M(i,j) \leq S(j)$, ha nincs ilyen akkor holtpont van, mert egy folyamat se tud végigfutni.
- 2. Az i . folyamat megkap mindent, lefut, majd az erőforrás foglalásait adjuk $S(j)$ -hez
- 3. Ismételjük 1,2 pontokat míg vagy befejeződnek, vagy holtpontra jutnak.

Több erőforrásos bankár példa

	Processzus	Szalagmeghajtó	Rajzgépek	Nyomtatók	CD-ROM-ok
A	3	0	1	1	
B	0	1	0	0	
C	1	1	1	0	
D	1	1	0	1	
E	0	0	0	0	

$F(i,j)$: Lefoglalt erőforrások

$E(j)$: (6342) összes erőforrás

	Processzus	Szalagmeghajtó	Rajzgépek	Nyomtatók	CD-ROM-ok
A	1	1	0	0	
B	0	1	1	2	
C	3	1	0	0	
D	0	0	1	0	
E	2	1	1	0	

$M(i,j)$: További erőforrásigények

$S(j)$: (1020) a még szabad erőforrások

Kaphat B 1 nyomtatót, az még biztonságos állapot lesz. (D be tud fejeződni, utána A,E,C,B.)

Bankár algoritmus összegzés

- ▶ A korábbi megelőzés is, meg ez az elkerülés is olyan információt kér (az erőforrás pontos igényeket, a folyamatok számát előre), ami nehezen megadható.
 - Folyamatok dinamikusan jönnek létre, erőforrások dinamikusan módosulnak.
- ▶ Ezért a gyakorlatban kevesen alkalmazzák.
- ▶ ...

Köszönöm a figyelmet!

zoltan.illes@elte.hu