Hibatűrés

Majzik István
Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék
http://www.mit.bme.hu/

Hibatűrés különféle hibák esetén

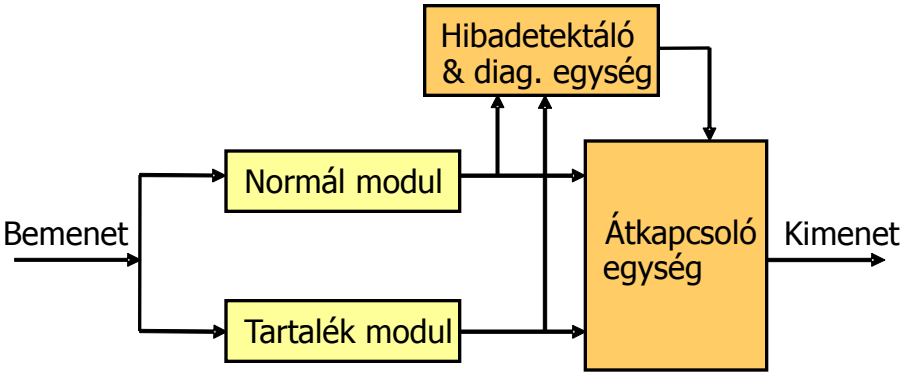
- Hardver tervezési hibák (< 1%):
  - Tipikusan nem számítanak rá (jól tesztelt komponensek)
  - Eltérő tervezésű hardver lenne szükséges
- Hardver állandósult működési hibák (~10%):
  - Hardver redundancia (pl. tartalék processzor)
- Hardver időleges működési hibák (~70-80%):
  - Szoftver redundancia (pl. állapotmentés és helyreállítás)
  - Idő redundancia (pl. utasítás újravégrehajtás)
  - Információ redundancia (pl. hibajavító kódolás)
- Szoftver tervezési hibák (~20-30%):
  - Szoftver redundancia (pl. eltérő tervezésű modulok)

Hibatűrés állandósult hardver hibák esetén

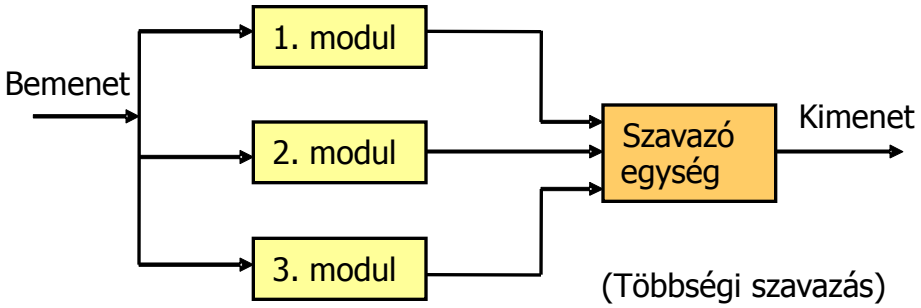
Hardver redundancia

Többszörözés:

- Kettőzés:
  - Hibadetektálás összehasonlítással
  - Hibatűrés: Diagnosztikai támogatás és átkapcsolás



- TMR: Triple-modular redundancy
  - Hiba maszkolása többségi szavazással
  - Szavazó kritikus elem (de egyszerű)

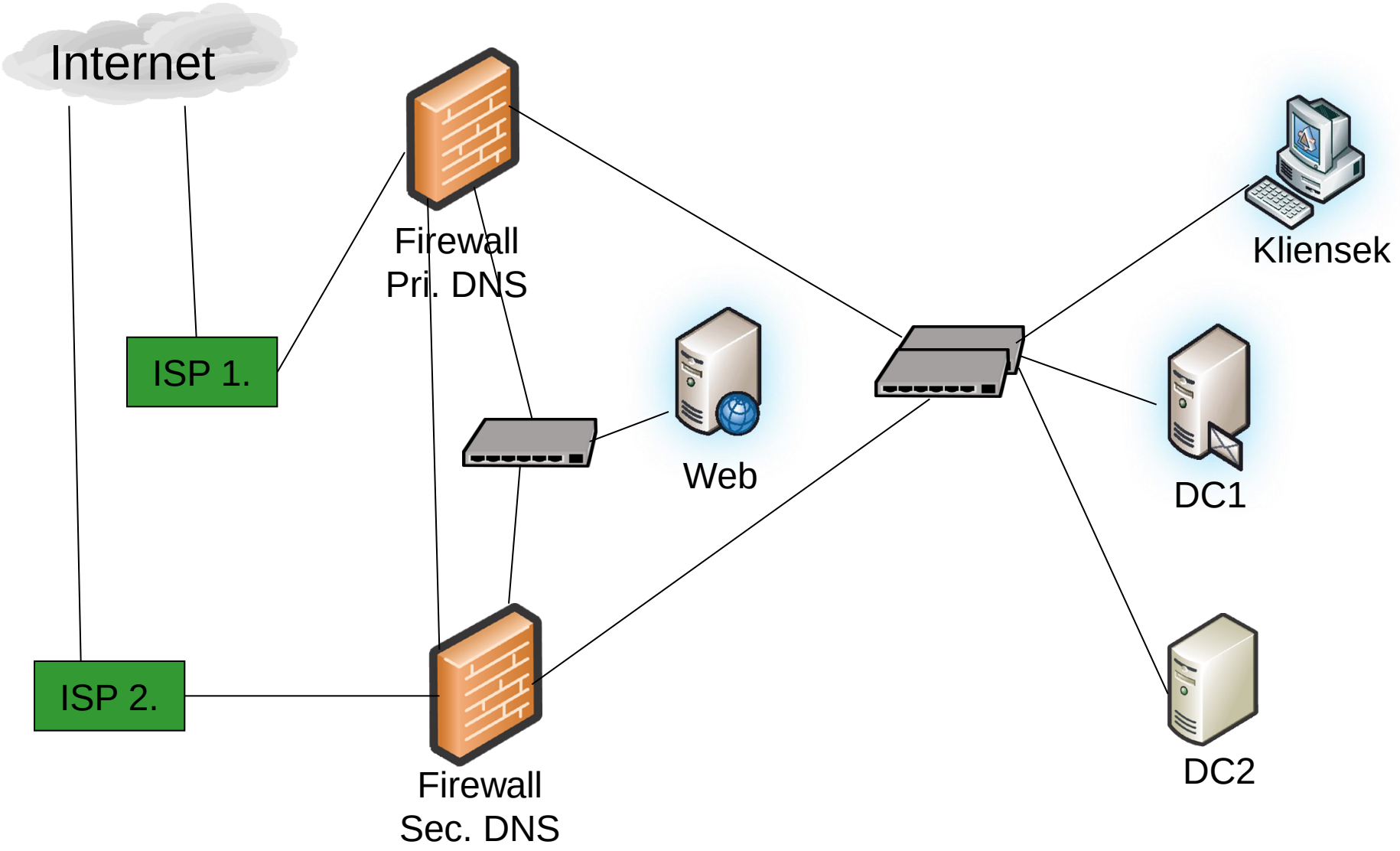


- NMR: N-modular redundancy
  - Hiba maszkolása többségi szavazással
  - Missziós idő túlélése nagyobb esélyű, utána javítás jöhet
  - Repülőgép fedélzeti eszközök: 4MR, 5MR

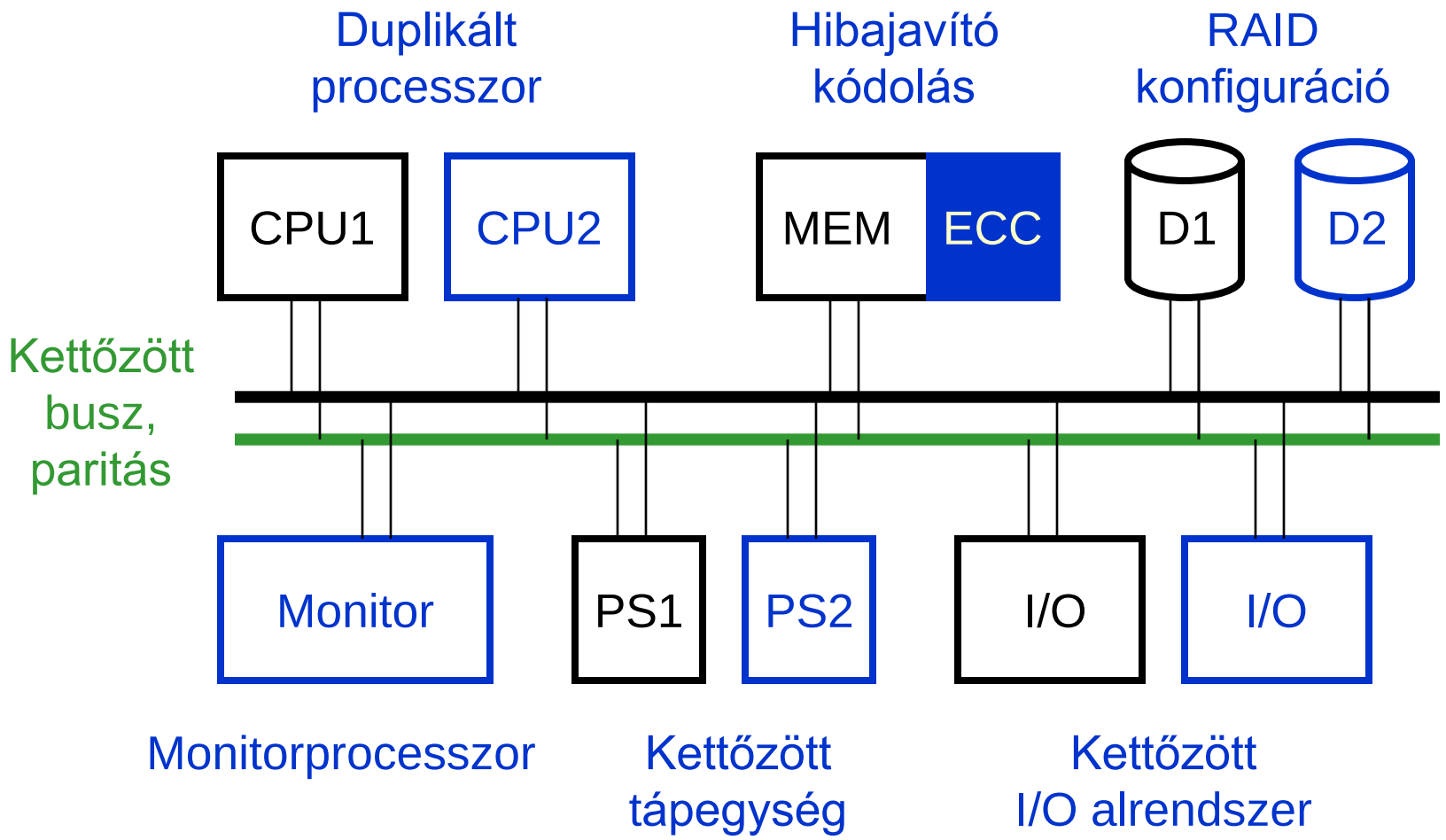
A redundancia szintje

- Eszköz (szerver) szint: Lazán csatolt
  - Nagy rendelkezésre állású szerver fürtök (feladatátvételi fürtök)
    pl. HA Linux
  - Szoftver támogatás: állapotszinkronizálás, feladatátvétel
- Kártya szint:
  - Futásidőbeli átkonfigurálás “hot-swap”
    pl. compactPCI, HDD
  - Szoftver támogatás: konfigurációkezelés
- Alkatrész szint: Szorosan csatolt
  - Alkatrész szintű többszörözés
    pl. TMR, önellenőrző áramkörök

Példa: Eszköz szintű redundancia…

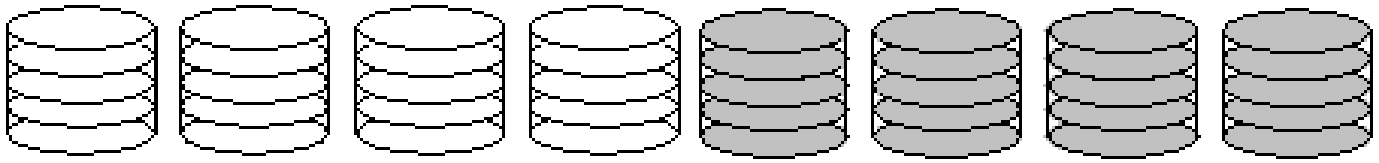


Példa: Kártya szintű redundancia…



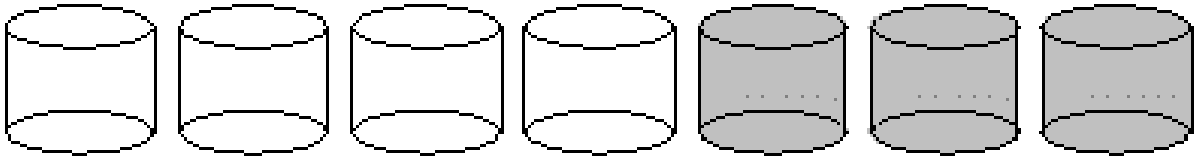
Példa: RAID

Tükrözött diszkek



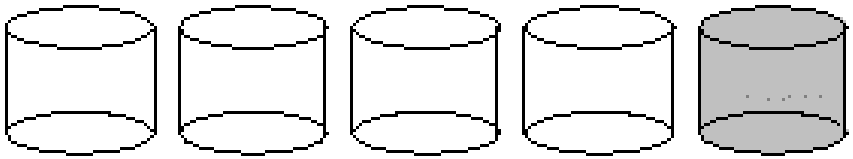
RAID-1

Hibajavító kódolás



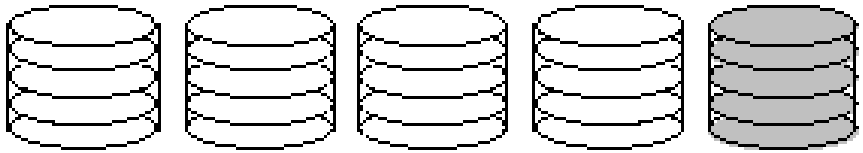
RAID-2

Azonosítható a hibás diszk: Paritás elég a javításhoz



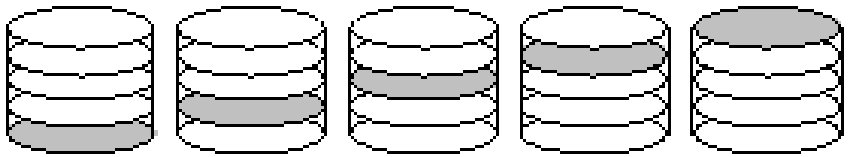
RAID-3

Konkurens hozzáférés a blokkokhoz



RAID-4

Paritásdiszk sem szűk kereszt-metszet



RAID-5

Példa: További RAID konfigurációk

- RAID-6
  - Kétféle elosztott paritás
  - Két diszk kiesése is tolerálható
  - Helyreállítás alatt is van redundancia
- RAID 0+1
  - Kombinált: Teljesítmény és hibatűrés ötvözése
  - Diszkek összefűzve (RAID-0), majd ez tükrözve (RAID-1)
  - Egy hiba esetén a tükrözés megszűnik
- RAID 1+0
  - Diszkek tükrözve (RAID-1), majd ez összefűzve (RAID-0)
- RAID 5+0
  - RAID-5 összefűzve
- RAID 5+1
  - RAID-5 tükrözve

Hibatűrés időleges hardver hibák esetén

Időleges hardver hibák kezelése

Megoldás: Szoftver alapú

- Ismételt végrehajtás esetén a hiba nem jelentkezik
- Hibahatások kiküszöbölése a fontos

A hiba kezelhető hibamentes állapot beállításával (és ismételt végrehajtással)

Feladatok (fázisok):

1. Hibadetektálás
2. Hibahatás felmérése
3. Helyreállítás
4. Hibaok (meghibásodás) kezelése

1. Hibadetektálás

- Alkalmazásfüggetlen mechanizmus:
  Pl. processzor, MMU, operációs rendszer szintjén
  - Illegális utasítás
  - Védelmi szintek, jogosultságok (pl. memória hozzáférés)

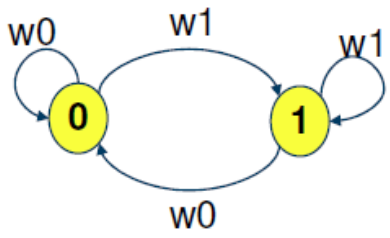
- Alkalmazásfüggő, ad-hoc módszer:
  - Időzítések ellenőrzése
  - Hihetőségvizsgálat
  - Visszahelyettesítés (algoritmus)
  - Struktúra ellenőrzés
  - Diagnosztikai ellenőrzés
  - …

Példa: SAFEDMI

- Indításkor:
  - Részletes öntesztelés
    - Processzor („core” utasítások alapján) + watchdog timer
    - Memória: teszt algoritmusok
- Működés közben:
  - Periodikus tesztelés
    - Kisebb erőforrásigényű technikákkal
  - On-line ellenőrzések:
    - Kommunikáció, konfigurálás: Hibadetektáló kódolás, adatok elfogadhatósági ellenőrzése
    - Vezérlési funkciók: Vezérlési gráf ellenőrzése (jelzőszámok)
    - Intenzív adatfeldolgozás: Duplikált végrehajtás és összehasonlítás

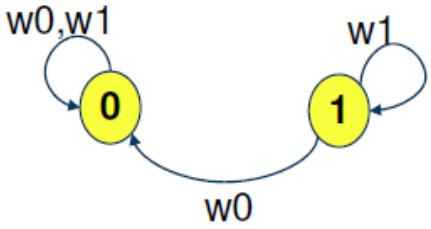
Példa: Memória tesztelése

Hibamentes cella:

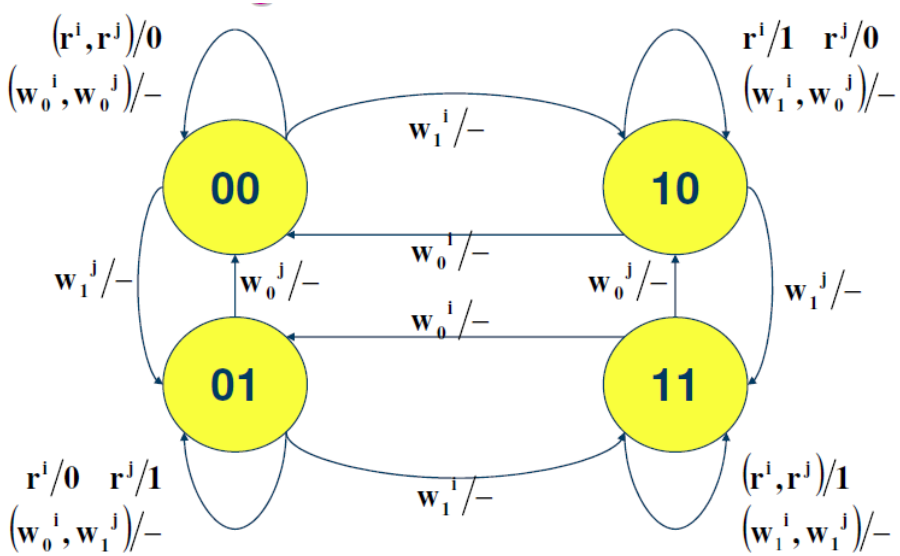


Leragadási hibák:

Tranzíciós hibák:



Le/összeragadás tesztjéhez:



„March” algoritmusok:

				1
			1	
		1		
	1			
1				

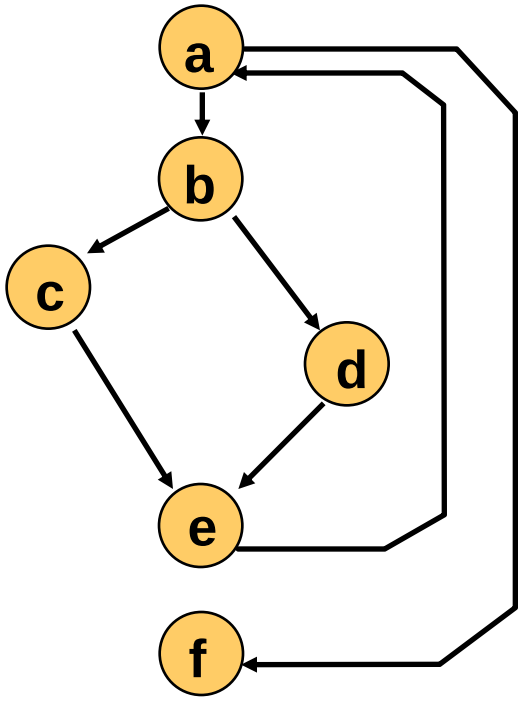
Példa: Szoftver önellenőrzés

- Végrehajtási útvonalak ellenőrzése
  - Referencia: Vezérlési gráf alapján

Forráskód:

```
a:  for (i=0; i<MAX; i++) {
b:      if (i==a) {
c:          n=n-i;
        } else {
d:          m=m-i;
        }
e:      printf("%d\n",n);
    }
f:  printf("Ready.")
```

Vezérlési gráf:



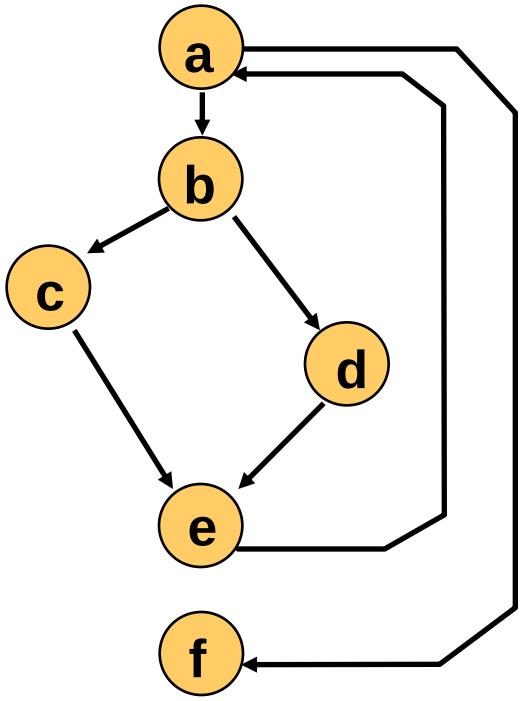
Példa: Szoftver önellenőrzés

- Végrehajtási útvonalak ellenőrzése
  - Referencia: Vezérlési gráf alapján
  - Aktuális futás: Jelzőszámok alapján ellenőrizhető

Forráskód:

```
a:  S(a); for (i=0; i<MAX; i++) {
b:      S(b); if (i==a) {
c:          S(c); n=n-i;
        } else {
d:          S(d); m=m-i;
        }
e:      S(e); printf("%d\n",n);
    }
f:  S(f); printf("Ready.")
```

Vezérlési gráf:



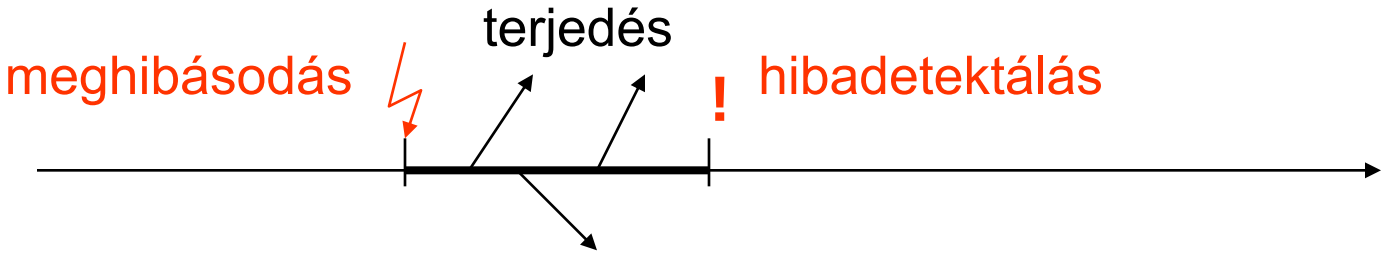
Járulékos feladatok

Keretrendszer hibadetektáláshoz (monitorozáshoz)

- Alkalmazásfüggő technikák beillesztése
- Detektálás ütemezése
- Hibajelzések kiadása
  - Ismételt hibajelzések kiszűrése
  - Függőségek kezelése (root cause analysis)
  - Naplózás
- Megszűnő hiba jelzése
- Hozzáférés hibajelzésekhez
  - Nézetek, szerepek
  - Nyugtázás

2. Hibahatások felmérése (kárfelmérés)

- Hibadetektálás késleltetési ideje alatt a hiba terjed
  - Pl. elosztott rendszer komponensei között, processzek között



- Hibaterjedés behatárolása
  - Interface (kimeneti, bemeneti) ellenőrzés
  - Erőforrásokhoz való hozzáférés korlátozása
  - Atomi jellegű műveletek kialakítása
- Hibaterjedés felmérése: interakciók követése
  - Interakciók naplózása
  - Diagnosztikai ellenőrzések

3. Helyreállítás

Egyszerű technikák:

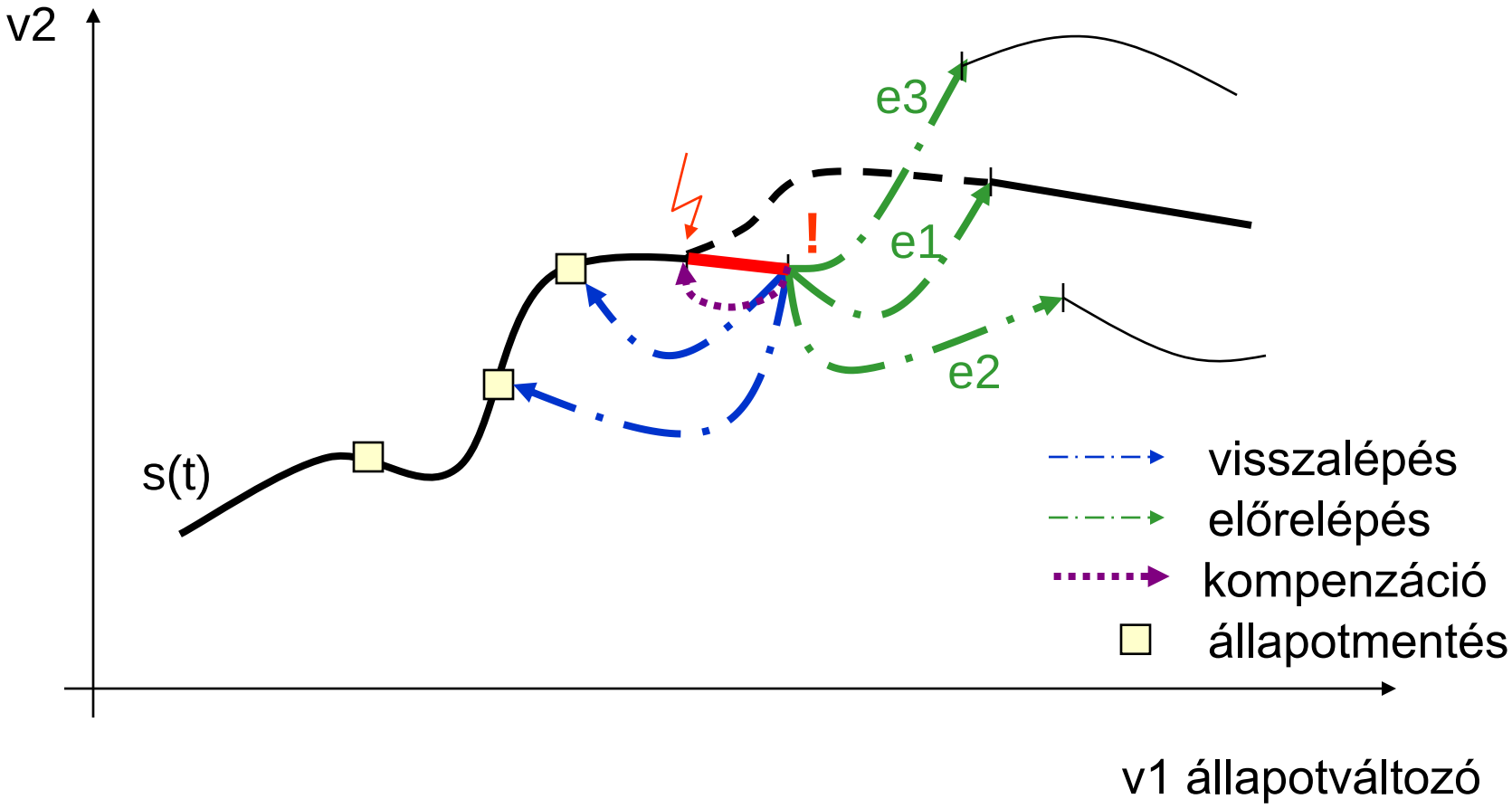
- Retry, restart, reboot; meleg reset, hideg reset; frame boundary

Összetett technikák:

- Előrelépő helyreállítás:
  - Hibamentes állapot beállítása szelektív korrekcióval
  - A korrekció a detektált hiba és a hibahatás függvénye
  - Előre figyelembe vett hibák esetén alkalmazható
- Visszalépő helyreállítás:
  - Korábbi hibamentes állapot beállítása
  - Hibától függetlenül megvalósítható
  - Állapotmentés és visszaállítás szükséges minden komponensre
- Kompenzáció:
  - Többlet információ alapján a hibahatás kompenzálható

A helyreállítási lehetőségek áttekintése

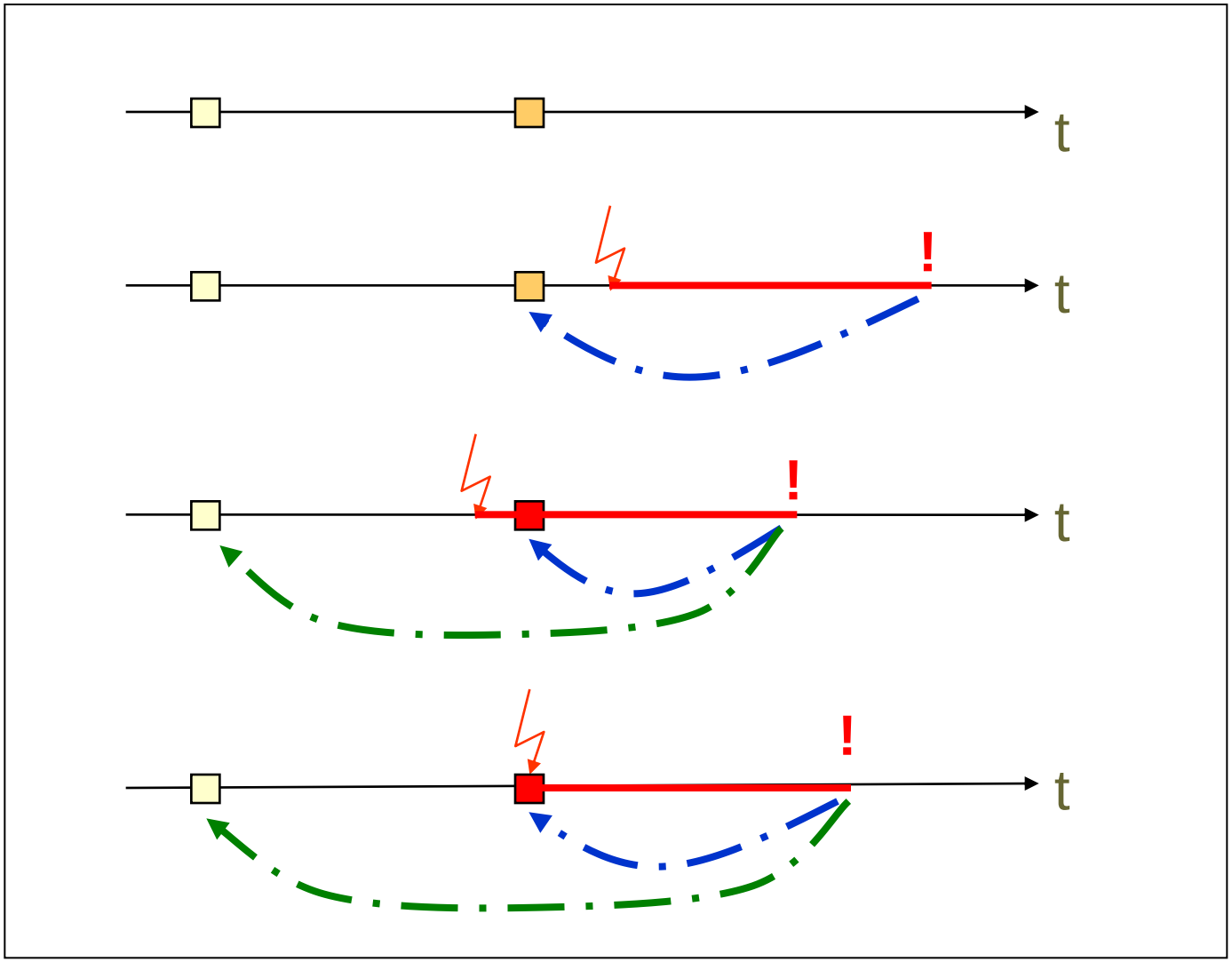
- Ábrázolás a rendszer állapotterében:



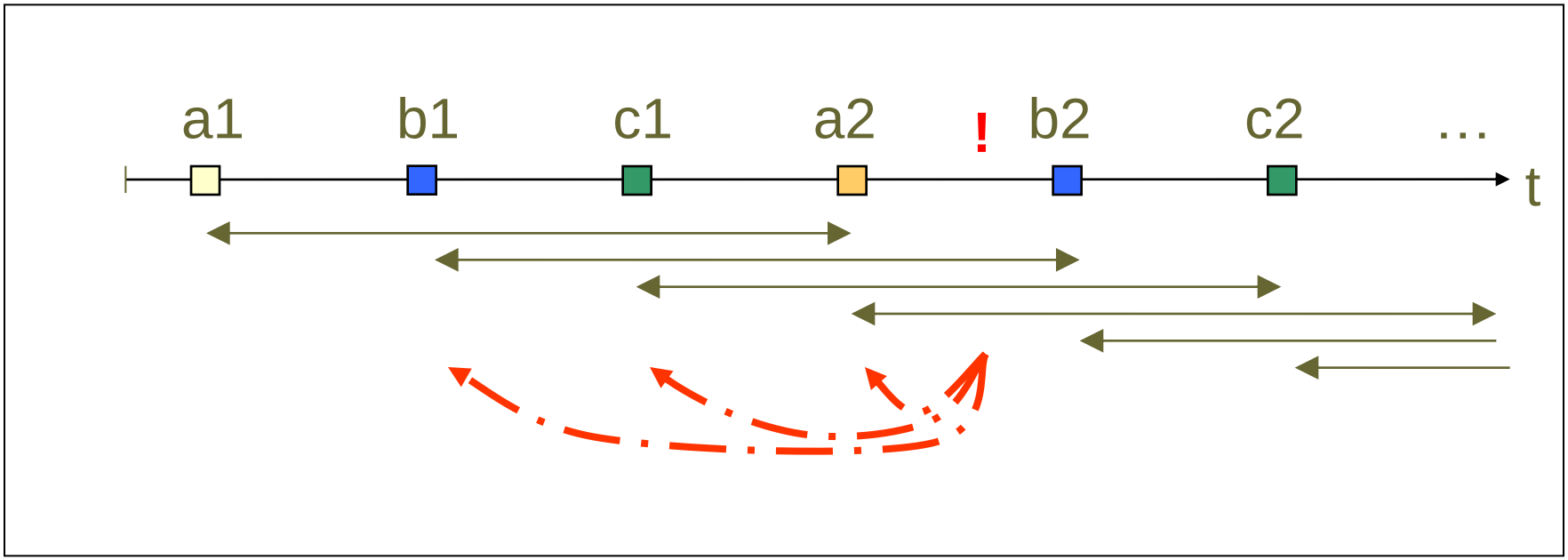
Visszalépő helyreállítás

- Állapotmentés alapján
  - Checkpoint: állapotmentés (időpontja)
  - Műveletek:
    - Állapotmentés: időközönként, üzenetek után
    - Visszaállítás: mentésből visszatöltés
    - Eldobás: adott számú checkpoint mentése után
  - Feltételezett hiba: időleges hardver hiba
  - Pl: “autosave”
- Műveletek visszavonása alapján
  - Műveletek naplózása szükséges
  - Feltételezett hiba: a téves vagy szándékolatlan művelet
  - Pl. többszintű “undo”
- Kombinált módszer

Visszalépő helyreállítás forgatókönyvei



Checkpoint tartományok

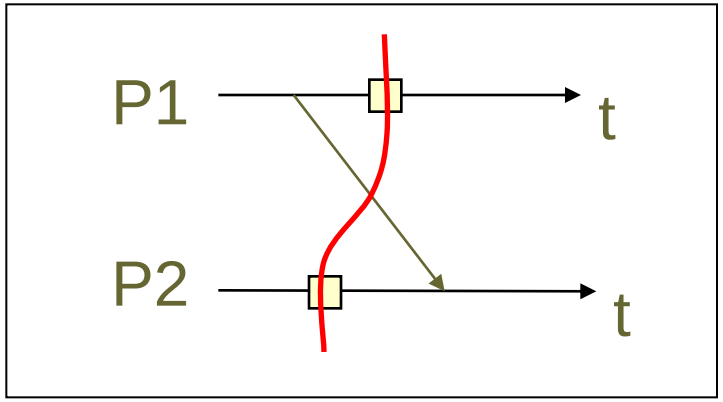


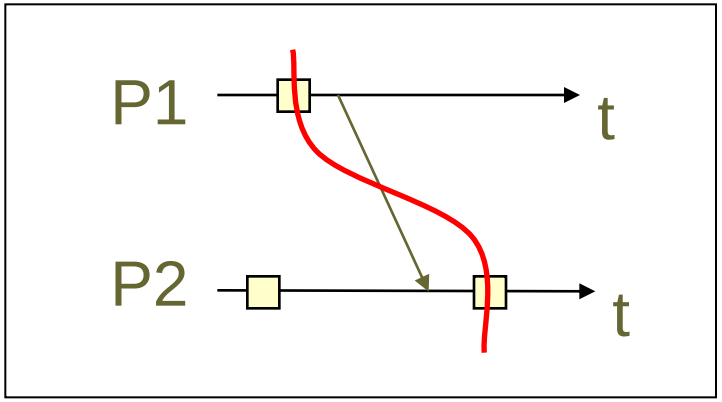
Optimalizálás szempontjai:

- Korlátos méretű tár az állapotmentéshez
- Visszalépéssel újra elvégzendő (elvesztett) számítások
- Hibadetektálás lappangási ideje

Helyreállítás elosztott rendszerekben

Üzenetek „metszik” a helyreállított állapotok által kijelölt vágatot:





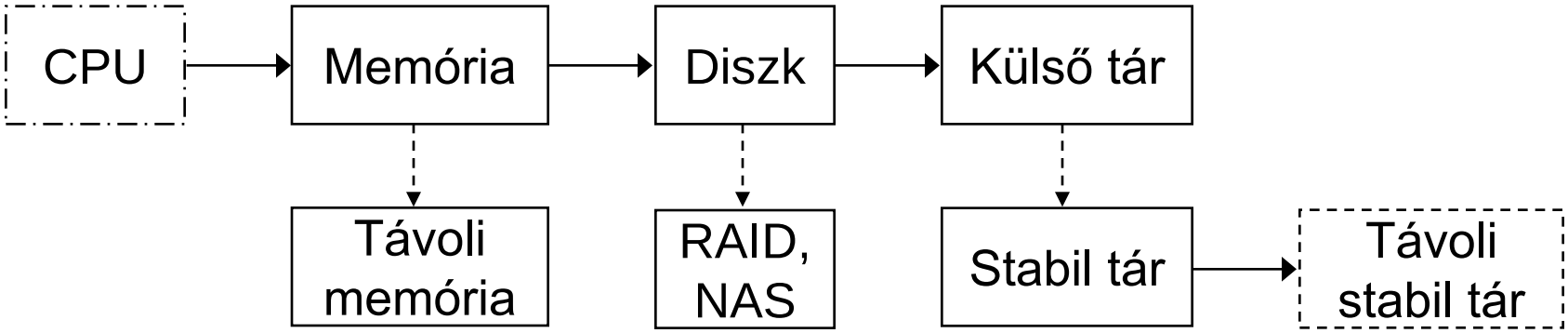
Függő üzenet:

- Előtte elküldött
- Utána feldolgozott

Inkonzisztens üzenet:

- Utána elküldött
- Előtte feldolgozott

Állapotmentések visszalépő helyreállításhoz



- Jellegzetes használati esetek:
  - Hosszú számítási idejű alkalmazások
    - Tudományos számítások, folyamatos működésű alkalmazások
  - „Commodity computing”: Skálázhatóság olcsó hardverekkel
    - Adat analízis alkalmazások: Pl. Google: Átlagban 5 számítógép kiesése egy-egy Map-Reduce job végrehajtása alatt
    - Nagyméretű infrastruktúrák: 4000 számítógépből álló hálózat: 6 óránként egy-egy diszk kiesése

Stabil tár koncepciója

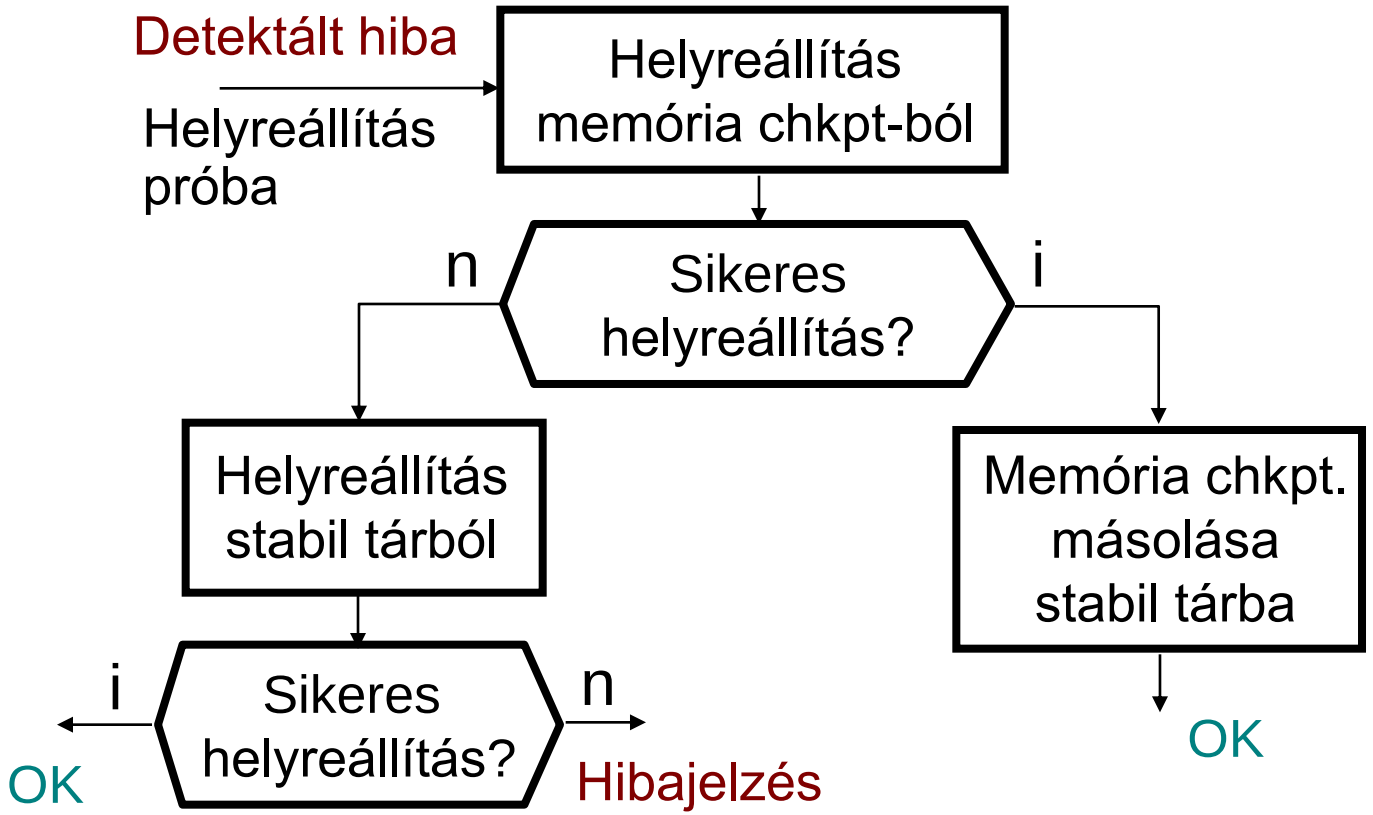
- Elvárt tulajdonságok
  - Perzisztens
  - Atomi hozzáférésű
  - Autonóm működésű
  - Hibajavítást alkalmazó
  - Információrejtést alkalmazó
- Többféle megvalósítás lehetséges

Kihívások visszalépő helyreállítás esetén

- Komponensek visszaléptethetősége
  - Pl. CPU állapot mentése és visszatöltése
  - Determinisztikus működés
- Környezettel való együttműködés kezelése
  - Bemenet: naplózás
  - Kimenet: késleltetés vagy ellensúlyozás
- Megbízhatóság és helyreállítási garancia
  - Hibadetektálás hibafedésének növelése, késleltetési idejének csökkentése
  - Mentés helyének megbízhatósága (replikák)
- Hatékonyság növelése
  - Konkurens állapotmentés
  - Másolatok használata (shadow update, recovery cache)
  - Többszintű állapotmentés („soft” és „hard” checkpoint)

Többszintű állapotmentés

- Alapelv:
  - Gyakori mentés gyors memóriába (memória chkpt.)
  - Ritka mentés perzisztens tárba (stabil tár)

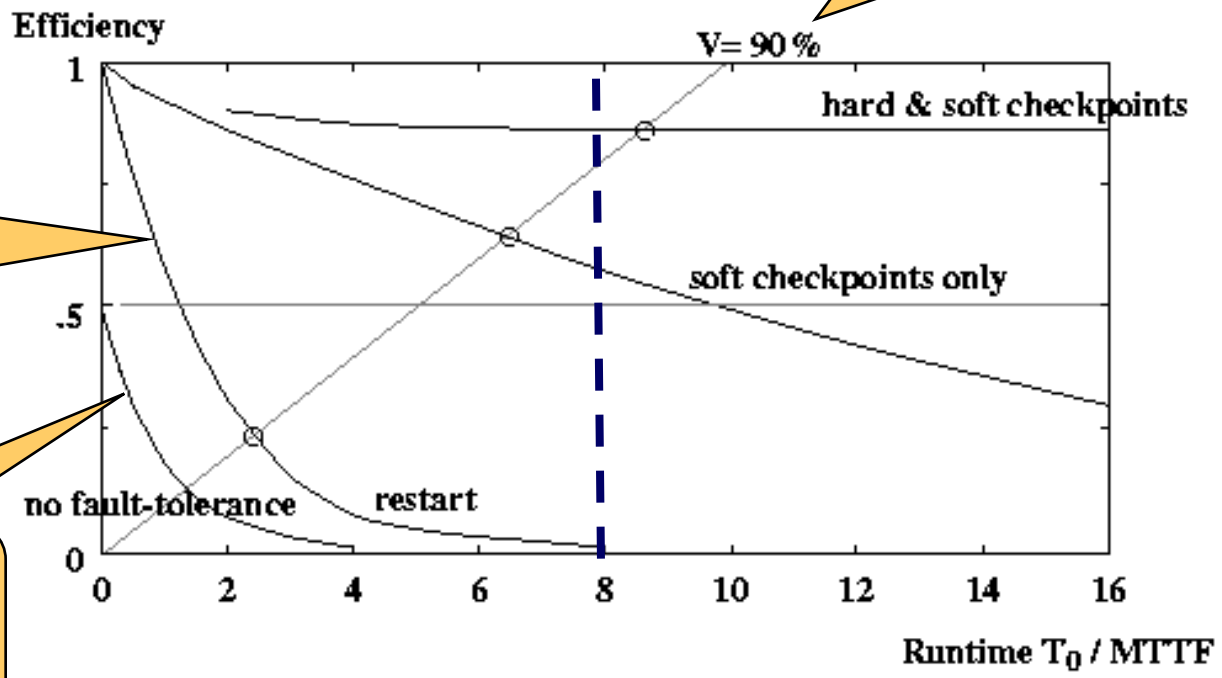


Mentések hatékonysága

Itt a hatékonyság:

$$\frac{\text{referencia (hibamentes) futási idő}}{\text{hibakezeléssel mért futási idő}}$$

V=90% vonal:
A hibadetektálás 99% hibafedése esetén 90%-os bizalom az eredmények helyességében a vonal feletti területen



99% hibafedésű hibadetektálás van

Újra és újra futtatás, míg ugyanaz az eredmény ki nem jön

Példák visszalépő helyreállítás megvalósítására (ld. gyakorlat)

- Felhasználói állapotmentés
  C nyelvű, alacsony szintű megoldással
- Állapotmentés az SA Forum köztesrétegben (HA middleware), C nyelvű implementáció
- Állapotok (objektumok) mentése J2SE alkalmazásokban
- Állapotmentés és helyreállítás adatbáziskezelő rendszerekben

4. Meghibásodás (hibaok) kezelése

- Időleges hibák:
  - Előre- vagy visszalépő helyreállítás elég
- Állandósult hibák:
  - Helyreállítás nem lesz sikeres (újra detektált hiba)
    - Ez egy mechanizmus lehet az állandósult hiba felismerésére
  - Hiba lokalizálás (diagnosztika) után beavatkozás kell:
    - Újrakonfigurálás:
      Hibás komponens feladatainak átvétele
    - Javítás, csere
    - Graceful degradation:
      Csökkentett funkcionalitású működés,
      de a kritikus funkciókat megtartva

Példa: SAFEDMI hibakezelése

- Működési módok:
  - Startup, Normal, Configuration
- Hibadetektálás esetén: Tranziens hiba feltételezése
  - Visszalépés Startup állapotba és részletes önteszt
- Állandósult hiba lehetőségének kezelése:
  - Visszalépés Suspect állapoton keresztül:
    Hibaszámlálás (adott idő alatt korlátozott számú visszalépés megengedett)
  - A hibaszám túllépése esetén Safe állapotba lépés (leállítás)