Tartalomjegyzék

Valaki tudna berakni egy helyes példát aszinkron/szinkron állapotgép szorzatra?

Ja, a 2. gyakorlat megoldásában benne van (6. feladat):
https://inf.mit.bme.hu/sites/default/files/materials/category/kateg%C3%B3ria/oktat%C3%A1s/bsc-t%C3%A1rgyak/rendszermodellez%C3%A9s/14/remo-gyak-2-megoldasok.pdf

Itt vannak a ZH feladatok, valaki legyen szives kidolgozni.

2015 - PótZH

1. nagyfeladat — Folyamat- és teljesítménymodellezés, statisztika (16 pont)

Üzleti tranzakcióink hibatűrő és hatékony adattárolását kell megoldanunk. A friss tranzakciók az alkalmazásszerver két merevlemezéből álló redundáns adattárába kerülnek, amely RAID-1 (tükrözés) szervezésű, mindkét lemezre azonos adattartalmat replikál. A nagyobb adatbiztonság érdekében a tranzakcióról egy helyi (on-site) és egy távoli (off-site) backup tárolórendszerbe elhelyezünk egy-egy másolatot. Mivel az alkalmazásszerver kapacitása véges, ezért a régebbi tranzakciókat egy idő után eltávolítjuk (archiváljuk), de kizárólag miután már készült róluk biztonsági mentés (amelyet archiváláskor sem törlünk).

Új tranzakció rögzítésekor a RAID tömb mindkét diszkjére (tetszőleges sorrendben, akár átlapolva) el kell végezni az írást; később el lesz készítve a két backup egymással párhuzamosan: végül megfelelő idő eltelte után archiválandó (mindkét lemezről egyszerre eltávolítandó) a tranzakció.

A visszaolvasási igények 95%-a viszonylag friss, még nem archivált tranzakciókra vonatkozik: ilyenkor mindig az alkalmazásszerverhez fordulunk, ahol a két lemez közül terheléstől függően bármelyik kiszolgálhatja az olvasási kérést. A már archivált tranzakciók olvasásakor ellenben az on-site backupot kell visszaolvasni (az off-site backup csak hiba esetén történő helyreállításra szolgál).

A szigorú jogi szabályozás miatt már rögzített tranzakciókat sosem módosítunk, mindig csak új tranzakciókat rögzítünk. Ezért például egy korábbi tranzakció sztornózása két fő lépésből áll: először az eredeti tranzakciót be kell olvasni, majd ezután a hatását semlegesítő sztornó utasítást új tranzakcióként kell eltárolni.

Egy új tranzakció rögzítése az alkalmazásszerver egyik lemezére átlagosan 10 ms-ig. visszaolvasása 2 ms erejéig teljesen lefoglalja a lemezt. Mindkét backupra 20 tranzakciót tudunk másodpercenként kiírni, és erezel full duplex módon (tehát az ivási sebességet nem befolyásolva) az on-site backupról másodpercenként 30, az off-sitt backupról másodpercenként 5 tranzakciót lehet visszaolvasni. Az archiválást mindig sok tranzakción egyszerre végezzük, és csak egy katalógusbejegyzés törléséből áll, így tranzakciónkénti műveletigénye elhanyagolható.

a) Készítsen folyamatmodellt a sztomózásról! Jelenjen meg egy főfolyamat, amelynek az írás ill. olvasás lépését a fenti leírás szerint alfolyamatként finomítjuk! (4p)

b) Jólstrukturáltak-e az elkészült folyamatok? Mi alapján állapítottuk ezt meg? (2p)

c) Hosszú távú átlagban legfeljebb hány új tranzakció lenne rögzíthető a rendszerben óránként, ha közben nem történne se visszaolvasás, se sztornózás? Rövid távon, ha a backupokkal elég később foglalkozni. mekkora a legnagyobb irási ráta, amit a rendszer még várakozási sorok növelése nélkül elbir? (3p)

d) Egy üzemszüneti napon új tranzakciók nem keletkeznek, de másodpercenként 200 tranzakció visszaolvasását igénylő üzleti elemzési folyamatot futtatunk. Mekkora az egyes erőforrások kihasználtsága, feltételezve, hogy ilyenkor is hasonló a visszaolvasott tranzakciók kora? Az elemzés hányszoros gyorsítását bírná még cl a tárolórendszer? (3p)

e) Ha a tranzakciókat mindig a rögzítésük után 400 nappal archiváljuk, és a tapasztalatok szerint a gyakorlatban körülbelül 60 millió tranzakciót szokott egyidejűleg tárolni az alkalmazásszerver, akkor átlagosan hány új tranzakció születik naponta? (2p)

f) A visszaolvasásra kért tranzakciók korának eloszlását tanulmányozzuk; a szórásuk 100 napnak bizonyult. Az eloszlás melyik kvantilise (percentilise) ismert az eddig közölt adatok alapján, és mi az értéke? (2p)

. nagyfeladat –Hibamodellezés és adatfolyamháló (20 pont)

Az előző feladatban említett üzleti tranzakciókat kezelő rendszert ezúttal szolgáltatásbiztonsági zempontból szeretnénk analizálni.

Ha vissza kell olvasni egy már backupba másolt, de nem túl régi (a RAID tömbből még nem rchivált) tranzakciót, a művelet sikeres lesz, amennyiben a RAID tömb legalább egyik lemezén, vagy az on-site és off-site backup tárhelyek egyikén elérhető az adat. Utóbbi akkor érhető el, ha a maga a távoli szolgáltatás él, és az internetkapcsolatunk nem hibásodott meg. Az on-site backup és a RAID tömb egyaránt csak akkor olvasható, ha a szerverterem klímája jól működik.

Az alkalmazásszerver RAID tömbjében használt merevlemezeink terhelése hosszú távon egyenletes, a gyári hibáikat az előzetes tesztelések kizárták, az elöregedést pedig megelőző cserével kerüljük el; ez alapján a meghibásodásaik modellezésére a szokásos közelítés jól alkalmazható. A katalógus szerint 2 000 óra MTTF-fel rendelkeznek, de az alkalmazásszerver a RAID-1 (tükrözéses) szervezésének köszönhetően egy lemez kiesését még adatvesztés és szolgáltatáskiesés nélkül tolerálja. Ha a tömb egyik merevlemeze meghibásodik, és a cserepéldány üzembe helyezése előtt a másik lemez is meghibásodik, akkor a RAID tömbben adatvesztés lép fel.

a) Rajzolja fel a „nem sikerül a tranzakció visszaolvasása" rendszerszintű hibajelenség hibafáját! (3p)

b) A hibafa redukció módszerét alkalmazva azonosítsa az egyszeres hibapontokat! (2p)

c) Adja meg képlettel (a számszerű értékeket behelyettesítve) egyetlen merevlemez megbízhatósági függvényét és meghibásodási rátáját az idő függvényében! Vegye figyelembe a fent ajánlott közelítést. (2p)

d) Ha soha nem cserélnénk a meghibásodott lemezeket, várhatóan mennyi idő alatt történne adatvesztés a RAID-1 tömbben (a backup másolatoktól eltekintve)? (2p)

e) Valójában persze cseréljük a meghibásodott lemezeket. A RAID-1 tömb egyetlen elemének mi a készenléti tényezője (aszimptotikus rendelkezésre állása), ha az oda csatlakoztatott merevlemezt a meghibásodásakor pontosan 10 óra alatt tudjuk kicserélni (az egyszerűség kedvéért beleértve az összes szükséges műveletet, tehát a hiba detektálásán, izolációján, az új hardver beszerzésén, beszerelésén kívül az adatok redundáns másolatról történő replikációját is)? (2p)

f) Ha a RAID-1 tömb egyik lemeze épp meghibásodott, mekkora az adatvesztés valószínűsége a következő javítási ciklus alatt (képlettel megadva)? (2p)

g) A lemez szolgáltatásbiztonsági mérőszámain túl megállapítottuk, hogy a klíma hibavalószínűsége 10^{-5}, az off-site backup „négy kilences" rendelkezésreállású, a többi elemi meghibásodás pedig minden időpillanatban 0,001 valószínűséggel fordul elő. Számítsuk ki a rendszerszintű hibajelenség bekövetkeztének valószínűségét a redukált hibafa segítségével! A számítás során hol kell közelítéseket, feltételezéseket tenni? (3p)

h) Készítsen adatfolyamháló modellt az alkalmazásszerver *olvasási* funkcionalitásáról! A modellben jelenjen meg a RAID vezérlő és a két lemez önálló komponensként, melyeknek a belső működését (ezt elég a lemezek egyikénél részletezni) állapotgép írja le. A kívülről kapott olvasási kéréseket a vezérlő véletlenszerű választással delegálja az egyik diszknek. Amikor a diszktől visszakapja az eredményt, továbbítja azt az egész rendszer kimenetén. Egy diszk egyszerre csak egy kéréssel tud foglalkozni, a többi addig sorban áll a FIFO csatornában. A vezérlőben viszont nincs ilyen korlátozás, pillanat- továbbításával. (4p)

ZH 2014

Pót ZH:

Kiskérdések:

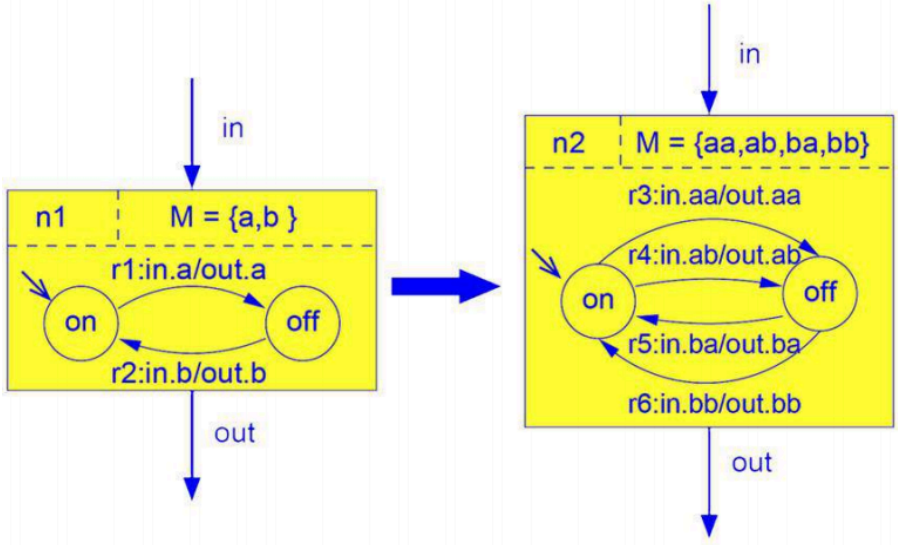
Modellfinomítás

1. Milyen modellfinomítási lehetőségeket ismer adatfolyamhálókban? Mutasson mindegyikre peldat!

- Tokenek halmazának finomítása: M'_n finomítottja M_n-nek
  - példa:

	n_1	$n_2 = \Re(n_1)$
Állapotok	on off	{on} {off}
Tokenek	a b	{aa, ab} {ba, bb}
Tüzelési szabályok	r1 r2	{r11, r12} {r21, r22}

- r1 = <on; in=a; off; out=a>
- r2 = <off; in=b; on; out=b>
- r11 = <on; in=aa; off; out=aa>
- r12 = <on; in=ab; off; out=ab>
- r21 = <off; in=ba; on; out=ba>
- r22 = <off; in=bb; on; out=bb>

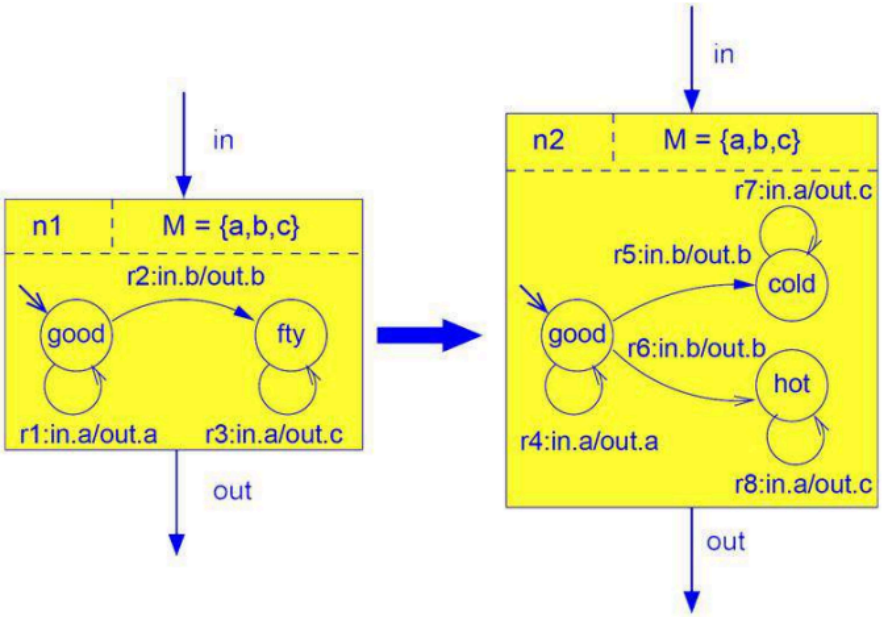


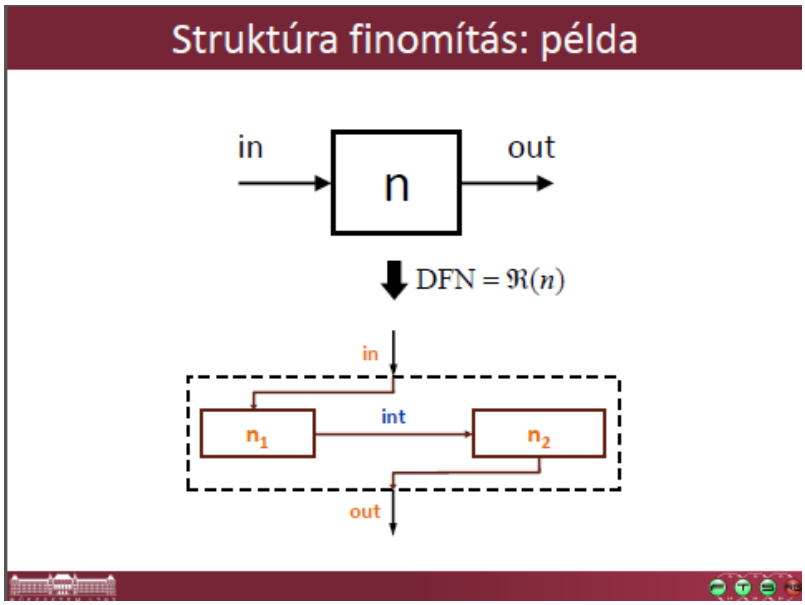
- Állapotok halmazának finomítása: S'_n finomítottja S_n-nek
  - példa:

	n_1	$n_2 = \Re(n_1)$
Állapotok	good	{good}
	fty	{hot, cold}
Tokenek	a	{a}
	b	{b}
	c	{c}
Tüzelési szabályok	r1	{r11}
	r2	{r21, r22}
	r3	{r31, r32}

- r1 = <good; in=a; good; out=a>
- r2 = <good; in=b; fty; out=b>
- r3 = <fty; in=a; fty; out=c>

- r11 = <good; in=a; good; out=a>
- r21 = <good; in=b; cold; out=b>
- r22 = <good; in=b; hot; out=b>
- r31 = <cold; in=a; cold; out=c>
- r32 = <hot; in=a; hot; out=c>



- Struktúrafinomítás
  - Csomópont helyettesítése adatfolyam alhálóval
  - Értelmezési tartomány finomítása
  - 

Meghibásodással kapcsolatos fogalmak

2. Mi a kapcsolat az alábbi fogalmak közt? Fault (meghibásodás), error (hiba), failure (hibajelenség). Adjon rájuk példát.

Hibajelenséget (**failure** - a specifikációnak nem megfelelő szolgáltatást (értékbeli/időzítésbeli, katasztrofális/„jóindulatú")) okoz a **hiba** (**error** - hibajelenséghez vezető rendszerállapot (lappangó→detektált)), aminek a **meghibásodás** (**fault**) az okozója.

Hatáslánc: meghibásodás (fault) → hiba (error) → hibajelenség (failure).

Példák:

- (hardver) elszáll egy merevlemez (fault), emiatt lehal a web szerver (error), a felhasználó nem éri el az oldalt (failure).
- (szoftver) megh.: programhiba, csökkentés helyett növel → hiba: vezérlés ráfut, változó értéke hibás lesz → hibajelenség: számítás végeredménye rossz.
- (hardver) megh.: kozmikus sugárzás egy bitet átbillent → hiba: hibás pozícióérték a memóriában → hibajel.: robotkar a falnak ütközik

Modellellenőrzés

3. Mit értünk modellellenőrzésen, milyen tipikus céljai vannak? Milyen módon adhatjuk meg a modellezett rendszerrel szemben a támasztott követelményeket?

(Modellellenőrzés: modellek kimerítő dinamikus analízise.) Azt ellenőrizzük, hogy a modell kielégíti-e támasztott követelményeket. Célja a modell verifikációja az eredeti rendszer követelményei alapján. Kérdése: jól építem-e a modellt? (validálás csak feltétel, ott a kérdés, hogy helyes rendszert építek-e) Pl. folyamatábrából kódváz - nem felejtettem ki egy ágat? stb..

Leírás: formalizmussal pl. temporális logikák (állapotokra vonatkozó állítások, stb…).

MODELLELLENŐRZÉS TOVÁBB

Metamodell

4. Mit nevezünk metamodellnek? Mi a kapcsolat modell es metamodell közt? Mutassa be egy példán keresztül.

Metamodell = egy modellezési nyelv modellje.

pl az UML metamodellje a MOF; adatbázistábla → (metamodellje) relációs adatbázisséma; XML-dokumentum → (metamodellje) XML-séma (vagy DTD)

5. Milyen elemekbol all egy felhasznaloi viselkedesmodell, milyen kozelito feltetelezessel el a felhasznaloi dontes leirasanal? Adjon peldat egy informatikai meroszamra, mely a modellbol meghatarozhato!

Customer Behavior Model Statechart (CBMS): UML-állapotdiagram (az szerepel rajta, amiket a felhasználó csinálhat, pl egy weboldal esetén: browse, login, register, search, pay etc.). Hasonló információt hordoz, mint CBMG (Customer Behavior Model Graph), de mindezt szabványos módon jeleníti meg.

Állapotok: az oldal funkciói (állapotai). Lehetséges átmenetek: rendszer modellje alapján

Átmenetekhez valószínűség tartozik (dinamikus viselkedés)

Mérőszámok:

- Revenue Throughput: az oldal által elért átlagos bevétel
- Potential Loss Throughput: mennyibe kerül a szolgáltatás kiesése egy adott időszakban
- Visit Ratio: átlagosan hányszor vesznek igénybe egy adott szolgáltatást (egy session alatt)
- Buy to Visit Ratio: átlagosan hányszor vásárolnak egy session alatt (tényleges eladási tanzakció)
- Average Session Length: egy session átlagosan hány szolgáltatást vesz igénybe (nem időt mér)
- mindezen mérőszámok változása, ha a bemenő paraméterek megváltoznak (pl. egyes átmenetek valószínűségei)

TODO!!!: "**milyen kozelito feltetelezessel el a felhasznaloi dontes leirasanal?**" ??

Előreláncoló rendszer

6. Mit értünk előreláncoló rendszer alatt? Milyen lehetősegek vannak üzleti szabályok kiértékelesénél a konfliktusok feloldasárá?

tények vannak rögzítve, illetve szabályok. ha a szabályok illeszkednek az éppen aktuális tényekre, akkor aktiválódnak, ez egy konfliktus-halmaz. Ebből kell kiválasztani egy szabályt konfliktus rezolúciós stratégiával (pl mélységi keresés, legjobb először, stb) amit alkalmazunk. A szabályok tényeket törölhetnek, ill szülhetnek.

(Legalábbis ez volt MI-ből nem? Nyugadtan írja át aki ezt látta konkrétan valamelyik diában)

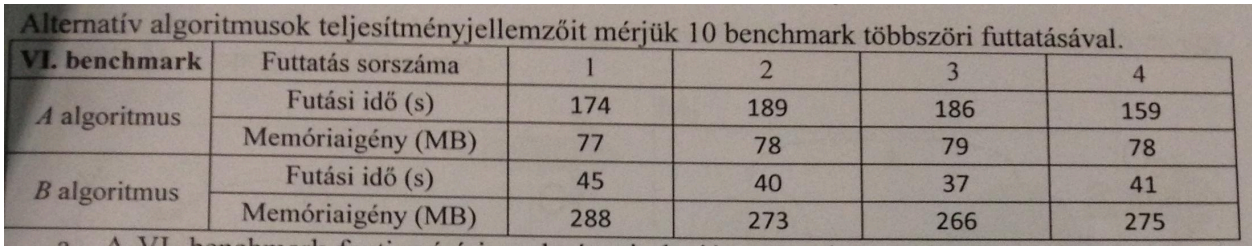
→ itt van (előreláncoló szót amúgy egybeírják, csak ők rosszul írták ;)):

Szabályalapú üzleti logikánál a szabályalapú rendszereken belül a következtető gépek (inference engines) következtető mechanizmusa lehet előreláncoló/produkciós (tiszta logikai; vagy épp üzleti szabálymotor) vagy hátraláncoló (Prolog, stb.); →

Előre láncoló (induktív/produkciós, adatvezérelt)

- logikai következtetés („Ha egy szerv gyulladt, akkor fájdalmat okozhat") vagy üzleti szabályok (ha → akkor jellegű: „Ha az ügyfél sokat roamingol, ajánljunk más tarifát"; „ha az ügyfél 30 év alatti, emeljük 35%-kal az ajánlatot")
- A tényekből újabb tényeket képez (produkciós szabály)
- Egy következmény teljesítheti egy szabály feltételrészét
- Analógia: generatív nyelvtan, hatáselemzés
- Ilyenek például a üzleti szabályrendszerek
- Logikai következtetés (vs. üzleti szabály)
    - ha a feltétel érvénytelenné válik, a következmény is?
    - Üzleti szabályok produkciós rendszer szemszögből: Nem (feltétlen) logikai következtetés → Érvénytelenné váló feltétel, akció hatása mégis megmarad

1.) Adatelemzés és kísérlettervezés



Alternatív algoritmusok teljesítményjellemzőit mérjük 10 benchmark többszöri futtatásával.

VI. benchmark	Futtatás sorszáma	1	2	3	4
A algoritmus	Futási idő (s)	174	189	186	159
	Memóriaigény (MB)	77	78	79	78
B algoritmus	Futási idő (s)	45	40	37	41
	Memóriaigény (MB)	288	273	266	275

Emlékeztető: a σ szórású normális eloszlás 68%-os konfidenciaintervalluma 1σ, a 95%-os 2σ, a 99,7%-os 3σ sugarú. A (folytonos) egyenletes eloszlás szórása az értéktartomány szélességének GYÖK(12)-edrésze, az exponenciális eloszlás szórása pedig megegyezik a várható értékével.

Alternatív algoritmusok teljesítményjellemzőit mérjük 10 benchmark többszöri futtatásával.

VI. benchmark	**Futtatás sorszáma**	**1**	**2**	**3**	**4**
A algoritmus	**Futási idő (s)**	**174**	**189**	**186**	**159**
	Memóriaigény (MB)	**77**	**78**	**79**	**78**
B algoritmus	**Futási idő (s)**	**45**	**40**	**37**	**41**
	Memóriaigény (MB)	**288**	**273**	**266**	**275**

a. A VI. benchmark fenti mérési eredményei alapján mennyi az A algoritmus futási idejének tapasztalati átlaga és korrigált tapasztalati szórása? (2p)

tap. átlag: m = (174 + 189 + 186 + 159)/4 = 708/4 = 177s

Futtatás sorszáma	1	2	3	4
x_i [s]	174	189	186	159
$(x_i - m)$ [s]	-3	12	9	18
$(x_i - m)^2$ [s^2]	9	144	81	324

tapasztalati átlagtól való eltérések négyzetének összege: x = 9+144+81+324 = 558 s^2
minták száma: t = 4
korrigált szórás (ha nem lenne korrigált,nem kéne levonni 1-et):
s* = sqrt(x/(t-1)) = sqrt(558/3) = sqrt(186) = 13,638s
(valaki validálja légyszíves) - szerintem jó. - szerintem is így kell, így volt gyakon

b. Összesen 100 futtatásig folytatva a VI. benchmark mérését, az A algoritmus memóriaigényét átlagosan 77.82 MB-nak tapasztaltuk, 2.3 MB szórással. Mit állíthatunk a memóriaigény várható értékéről, ha 1000 ilyen kísérletből legfeljebb 3-ban szabad téves konklúziót mondanunk? (3p)

tap.átlag: m = 77.82 mb
tap.szórás: s = 2,3 mb
minták szama: t=1000
szóras: szigma = σ = s / sqrt(t) = 0,23 mb
1000 - 3 = 997 esetben helyesen kell “tippelnünk”, ez 99,7% bizonyosság (konfidencia).
Ennel a konfidenciaértéknel azt mondhatjuk, hogy a memóriaigény várható értéke a tap. átlag 3szigma (3*0.23=0.69-et kell kivonni meg hozzáadni) sugarú intervallumába esik. Azaz hogy a [77.13 mb, 78.51 mb] intervallumba esik.
(valaki validálja légyszíves) - szerintem jó.

c. Milyen értéket vehet fel két valószínűségi változó korrelációja? Adjon hozzávetőleges becslést a B algoritmus futási ideje és memóriaigénye közötti korreláció értékére! (2p)

WTF? :D
A korreláció -1 és +1 közé esik, és egyenlőség akkor és csak akkor áll fenn, ha a két változó lineáris kapcsolatban áll egymással.

“A statisztikában nem állnak rendelkezésre az elméleti értékek, így a tapasztalati korrelációt a következőképpen számítják:

$$r_{XY} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y}$$

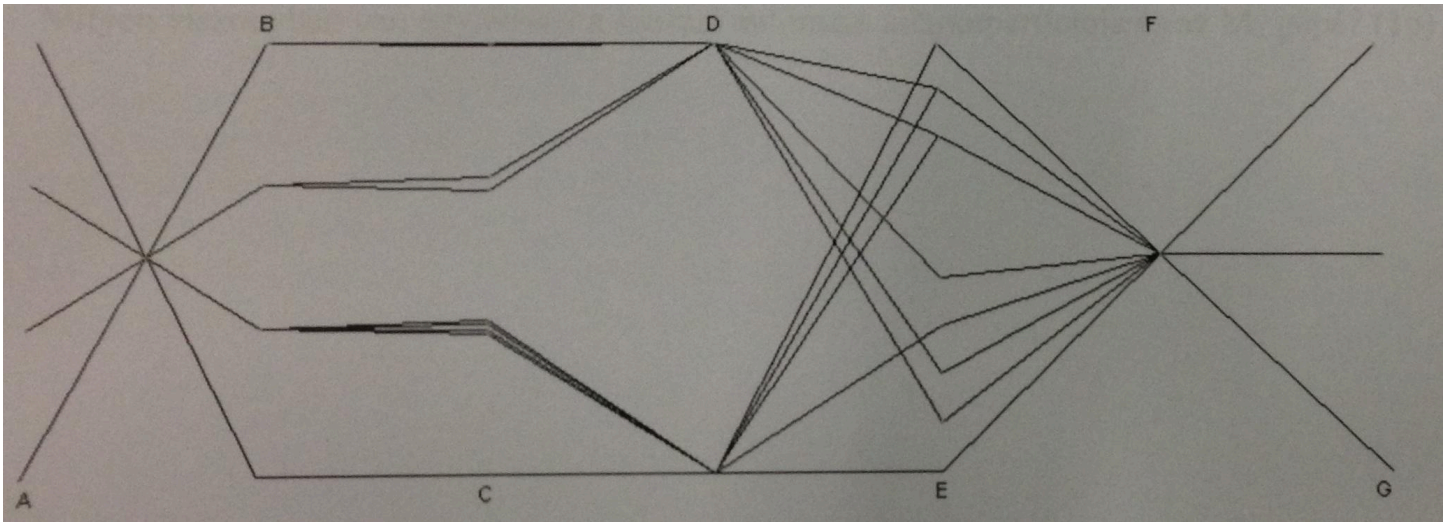
ahol a felülvonásos betűk a tapasztalati várható értéket, s_x, s_y a tapasztalati korrigált szórást jelölik.”

B algoritmus	Futási idő (s)	45	40	37	41
	Memóriaigény (MB)	288	273	266	275

ÉS AZTÁN?? Ki kéne számolni a tapasztalati **korrigált** szórást a memóriaigényre is ezek szerint?? És hogy történne a megfelelő behelyettesítés a képletbe?

TODO!!!

d. Az alábbi ábrán 7 algoritmus (A-G) átlagos memóriaigényét hasonlítjuk össze a 10 benchmark segítségével. Milyen viszonyban áll egymással az A és B algoritmus a különböző benchmarkokkal mért memóriaigény szempontjából? Mi az F algoritmus jellegzetessége? Hogy hívják ezt a diagramot? (3p)

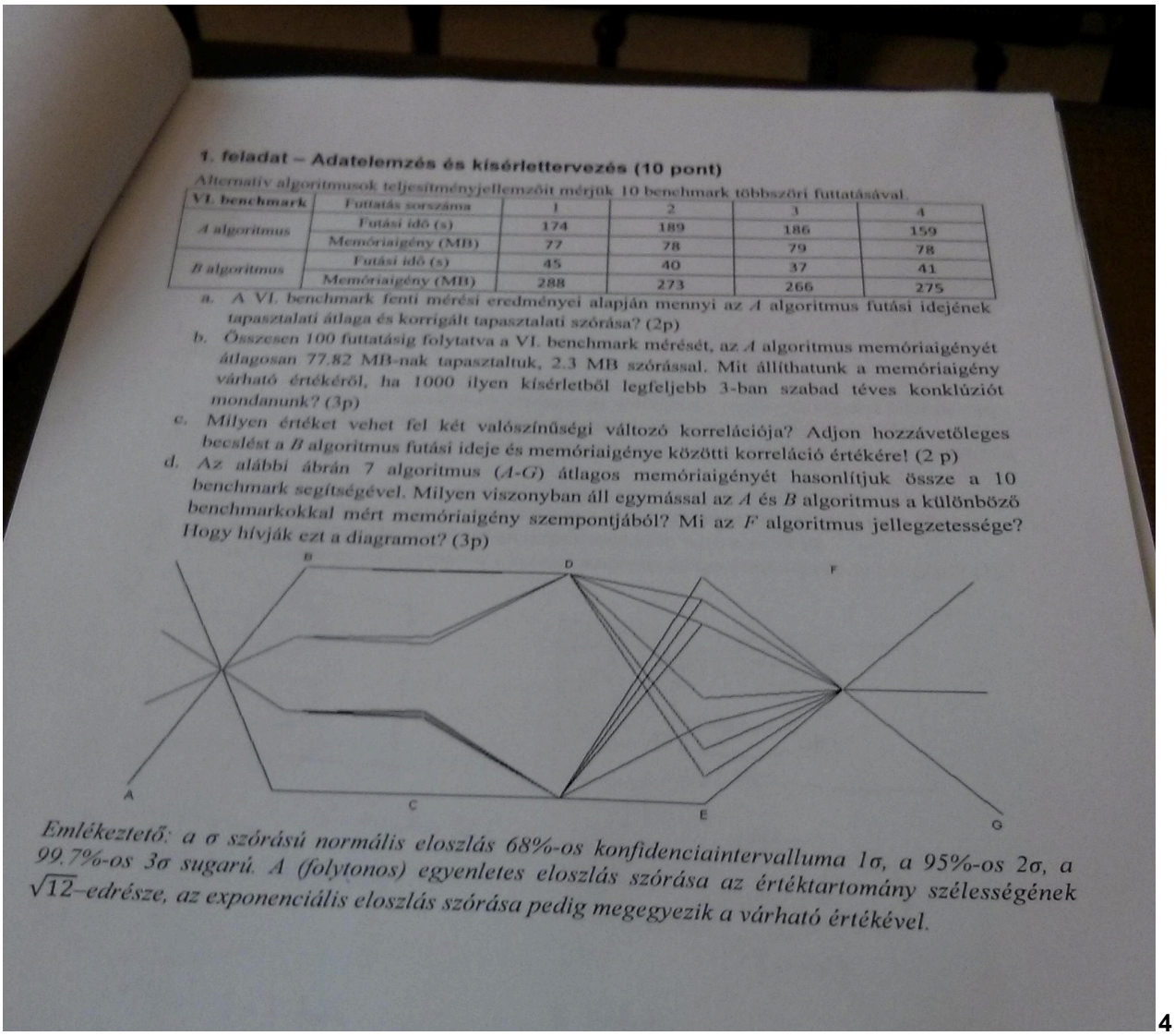


A és B között negatív korreláció van, F konstans? **(valaki validálja légyszíves)**

F-re nem jó az, hogy az F algoritmus jellegzetessége az, hogy minden egyes mérésnél azonos memóriaigényt mértünk? → **A és B fordítottan arányos, F konstans** (megoldókulcsból szedtem)

Párhuzamos koordináta diagram

DIAGRAM KEP



1. feladat – Adatelemzés és kísérlettervezés (10 pont)

Alternatív algoritmusok teljesítményjellemzőit mérjük 10 benchmark többszöri futtatásával.

VI. benchmark	Futtatás sorszáma	1	2	3	4
A algoritmus	Futási idő (s)	174	189	186	159
	Memóriaigény (MB)	77	78	79	78
B algoritmus	Futási idő (s)	45	40	37	41
	Memóriaigény (MB)	288	273	266	275

a. A VI. benchmark fenti mérési eredményei alapján mennyi az *A* algoritmus futási idejének tapasztalati átlaga és korrigált tapasztalati szórása? (2p)

b. Összesen 100 futtatásig folytatva a VI. benchmark mérését, az *A* algoritmus memóriaigényét átlagosan 77.82 MB-nak tapasztaltuk, 2.3 MB szórással. Mit állíthatunk a memóriaigény várható értékéről, ha 1000 ilyen kísérletből legfeljebb 3-ban szabad téves konklúziót mondanunk? (3p)

c. Milyen értéket vehet fel két valószínűségi változó korrelációja? Adjon hozzávetőleges becslést a *B* algoritmus futási ideje és memóriaigénye közötti korreláció értékére! (2 p)

d. Az alábbi ábrán 7 algoritmus (*A*-*G*) átlagos memóriaigényét hasonlítjuk össze a 10 benchmark segítségével. Milyen viszonyban áll egymással az *A* és *B* algoritmus a különböző benchmarkokkal mért memóriaigény szempontjából? Mi az *F* algoritmus jellegzetessége? Hogy hívják ezt a diagramot? (3p)

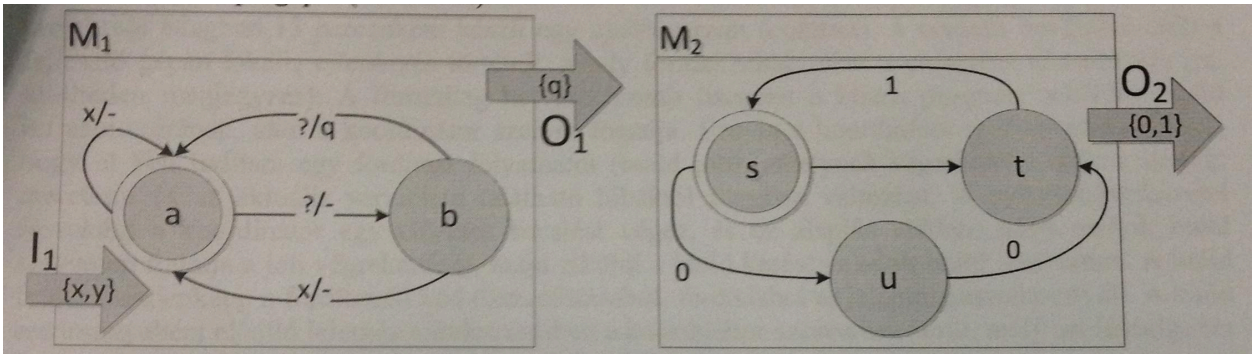
Emlékeztető: a σ szórású normális eloszlás 68%-os konfidenciaintervalluma 1σ, a 95%-os 2σ, a 99.7%-os 3σ sugarú. A (folytonos) egyenletes eloszlás szórása az értéktartomány szélességének $\sqrt{12}$-edrésze, az exponenciális eloszlás szórása pedig megegyezik a várható értékével.

4

2. Állapot alapú modellezés (12p)

(kapcsolódó gyak, mo.)

Az alábbi ábra mutatja az egy bemenetű és egy kimenetű M1, valamit a bemenet nélküli es egy kimenetű M2 állapotgépet (automatát).



ALLAPOTGEP

a. az M1 automata két szabályának elveszett az előfeltétele, ezeket kérdőjellel jelöltük meg. Töltsd ki úgy a

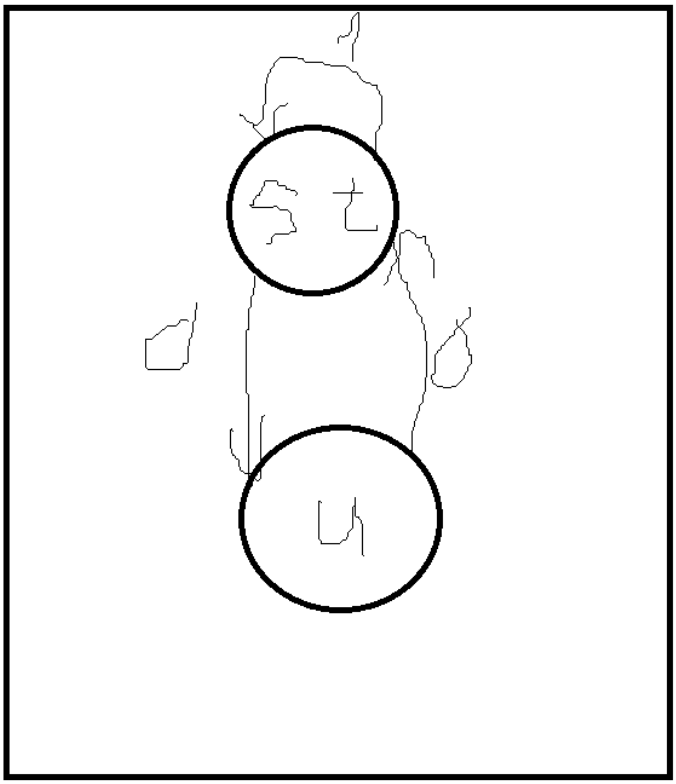
hiányzó inputokat, hogy determinisztikus legyen az automata. 1p

(determinisztikus: végignézünk minden állapotot és a kimenő élekre ellenőrizzük, hogy mindegyikből csak egy megy-e ki egy adott bemenetre; nemdeterminisztikus, ha -- van a bemeneten (ami ezt jelenti: bemeneten „nem olvasunk semmit” (bármikor megtörténhet az állapotátmenet, pl. --/z: valami állapotban spontán adhat ki z kimenetet))

y/q és y/-

b. Absztraháld az M2 automatát az ‘s’ es ‘t’ állapotok összevonásával! Hány választási lehetőség van? 3p

1 féle, az absztrakció mindig egyértelmű.
Az absztrakciót definiáló fv. szerint mindig csak egyféleképp lehet csinálni ({s,t} → x, legalábbis gondolom erre gondol).
Az s és a t állapotok összeolvadnak, az 1, 0 állapotátmeneti szabályok (élek) is összeolvadnak.



Tessék ;)

TODO!!! Rajz???

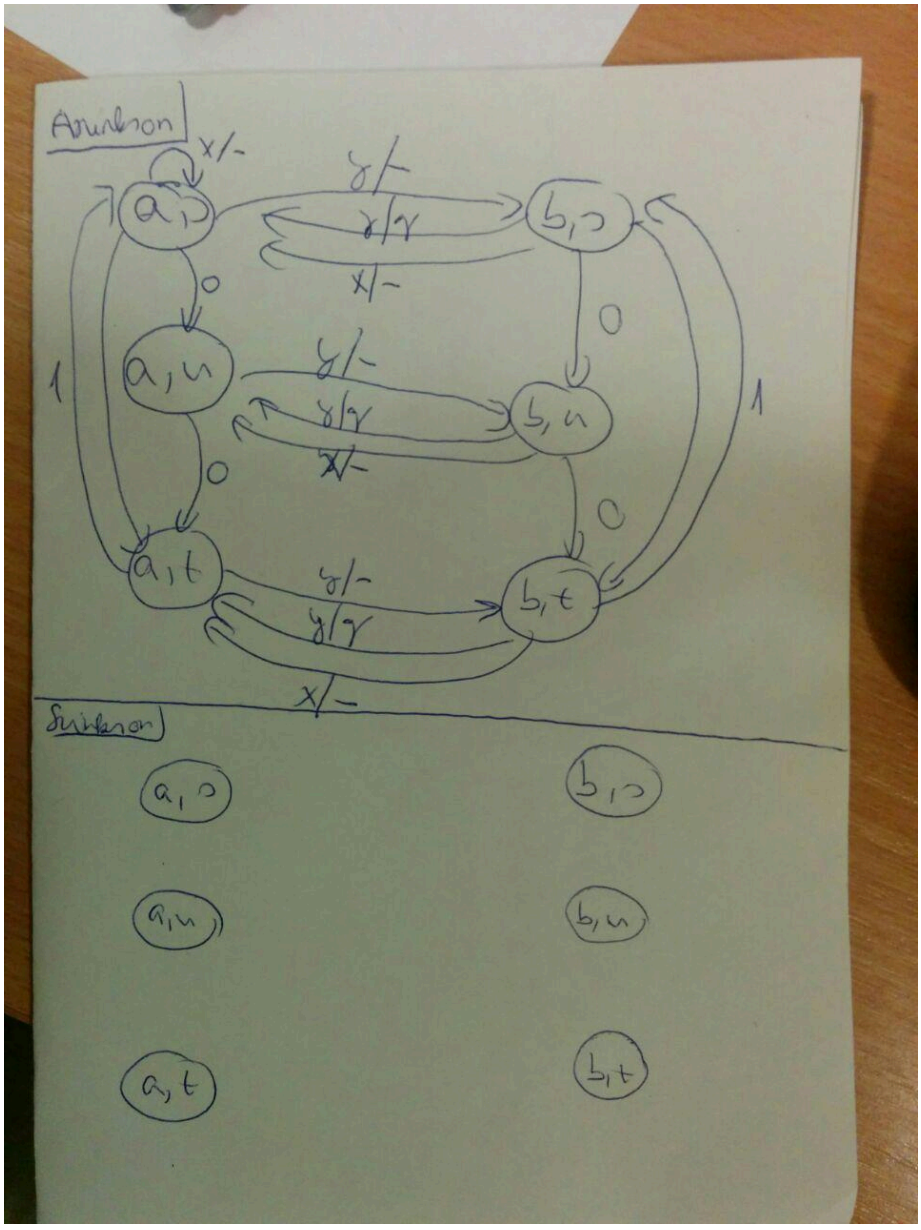
c. Finomítsd az M1 automatát a q token (m,n,o) esetekre bontásával! Hány választási lehetőség van a finomítás során? Melyik zárható ki, ha a deteminizmust meg akarjuk tartani? 3p

2^3 féleképpen kombinálhatom őket, de az hogy -/q nem opció, ezért (2^3)-1
Konkrétan ezek a lehetőségek ott, ahol q van a kimeneten (b → a): csak m átmenetet veszünk fel, csak n átmenetet veszünk fel, csak o átmenetet veszünk fel, vagy mindhárom átmenetet felvesszük (párhuzamos élekként), vagy kettőt-kettőt (m és n, m és o, n és o) == 7 lehetőség (nincs benne az, hogy egyik sincs benne, mert az nem jó).

TODO: Rajz???

d. Készitsd el az eredeti M1 es M2 állapotgépek aszinkron szorzatát! 4p

“Aszinkron szorzat: minden állapotátmeneti él lemásolódik a szorzatállapotokra. Ezt a két állapotgépet egyszerűen elkészíthetjük, ha felvesszük az állapotok Descartes-szorzatát, és az átmeneteket úgy rajzoljuk be, hogy a M1 átmenetei függőlegesen, M2 átmenetei vízszintesen szerepelnek az ábrán“



e. Milyen viszonyban van egymással a szorzat automata állapotpartíciója és az M1 gépé? 1p
Az automata allapotparticioja az M1 allapotparticiojanak finomitasa. (jobbat nem tudok)

Példa a szinkron és aszinkron szorzatra, állapotgép-absztrakcióra, ill. -finomításra gyakanyagból (megoldás):

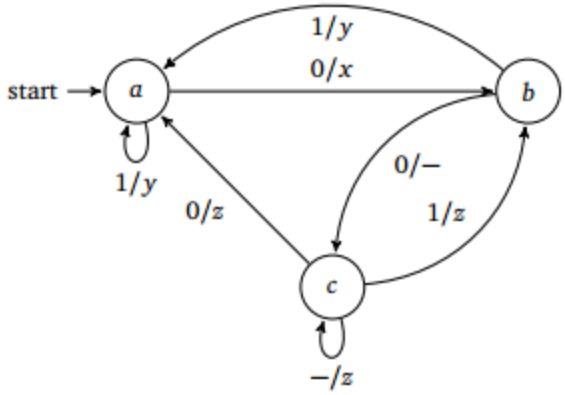
5. Tekintsük az M_1 szintetikus állapotgépet!

   a) Determinisztikus-e a viselkedésmodell? Hozzávehető-e, ill. elhagyható-e egyetlen állapotátmeneti szabály, hogy ez megváltozzon?
   b) Absztraháld az állapotgépet $\{a, b\} \to d$ állapotabsztrakcióval! Hány lehetőség van?
   c) Finomítsd az állapotgépet $a \to \{e, f\}$ állapotfinomítással! Hány lehetőség van?
   d) Absztraháld az állapotgépet $\{0, 1\} \to w$ tokenabsztrakcióval! Hány lehetőség van?
   e) Finomítsd az állapotgépet $z \to \{z_1, z_2\}$ tokenfinomítással! Hány lehetőség van?

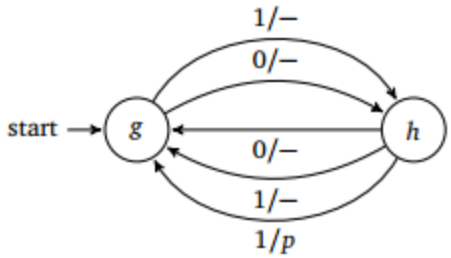
6. Készítsd el az azonos input csatornát olvasó M_1 és M_2 állapotgépek...

   a) ...szinkron szorzatát!
   b) ...aszinkron szorzatát!

M_1 állapotgép:



M_2 állapotgép:



5. feladat

a) Egy úgy dönthető el, ha végignézünk minden állapotot és a kimenő élekre ellenőrizzük, hogy mindegyikből csak egy megy-e ki egy adott bemenetre. Ennek megfelel az állapotgépünk, azonban van olyan állapotátmenet, amelynél — karakter van a bemeneten.

A — karakter (formális nyelvekben gyakran ε) jelentése: bemeneten „nem olvasunk semmit” (bármikor megtörténhet az állapotátmenet), ill. kimeneten „nem adunk ki semmit”. Ilyen van a c és az a állapotok között, ezért a modell nemdeterminisztikus, c állapotban spontán z-t adhat ki.

Fontos, hogy a — jelentése nem ugyanaz, mint a digitális technikában – ott „don't care”-t jelent, vagyis bemenet esetén olvasunk, de mindegy, hogy mit; kimenet esetén írunk, de mindegy, hogy mit.

Az állapotgép mindenképp el fog jutni c-be, így (prioritások nélkül) nem tudjuk determinisztikussá tenni állapotok/átmenetek felvételével.

b) Az absztrakciót definiáló fv. szerint mindig csak egyféleképp lehet csinálni. Az a és a b állapotok összeolvadnak, az $1/y$ állapotátmeneti szabályok (élek) is összeolvadnak.

c) A lehetőségek száma a következő. A kezdőállapot lehet e és f is (2). A három ki-, ill. bemenő állapotátmeneti szabály külön-külön háromféle lehet változhatnak (pl. c-ből 0-ra e-be, f-be vagy mindkettőbe menjen), ez $3^3 = 27$. Az $1/y$ hurokél 4 helyre is kerülhet, de legalább egy helyre kerülnie kell. Ezek behúzására összesen $2^4 - 1 = 15$-féle lehetőség van (mindegyik egyformán „jó”). Ez összesen $2 \times 27 \times 15 = 810$ lehetőség.

Ennek magyarázata, hogy a finomított modell több információt tartalmaz, mint az absztrakt.

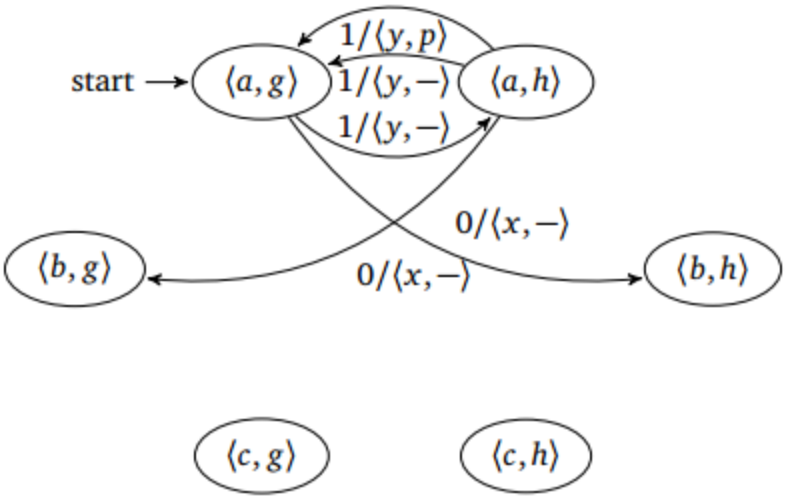
d) A b) részhez hasonlóan ismét csak egy lehetőségünk van.

e) Azon átegmenetek esetén, ahol z van a kimeneten ($c \to a, c \to c, c \to b$), háromféle lehetőségünk van: csak z_1-es átmenetet veszünk fel, csak z_2-es átmenetet veszünk fel vagy mindkét átmenetet felvesszük (párhuzamos élekként). Így összesen $3^3 = 27$ lehetőségünk van.

6. feladat

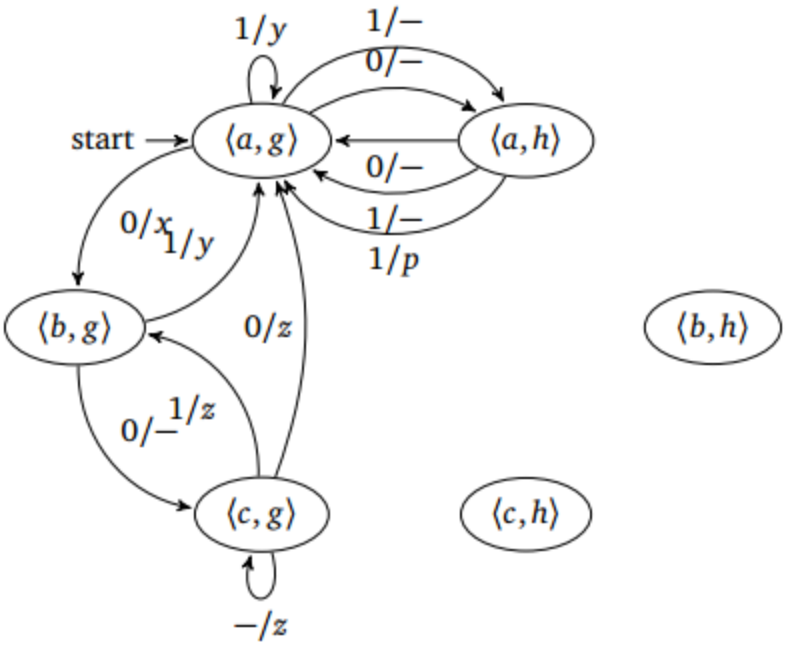
Két állapotgép partícióinak szorzata az állapothalmazok Descartes-szorzata. A gépek ugyanazt az i inputot olvassák mindketten.

Szinkron szorzat: az állapotgépeket egyszerre léptetjük. Nagyon fontos, hogy minden állapotra megvizsgáljuk, hogy 0, 1, ill. — bemenetre milyen állapotba kerülnek és az átmenet során milyen kimenetet adnak. Az alábbi ábrán az $\langle a, g \rangle$ és az $\langle a, h \rangle$ állapotokat vizsgáltuk meg a lehetséges bemenetekre.



Felmerül a kérdés, hogy az M_1 állapotgép c állapot — bemenetű hurokéle bekerül-e a szinkron szorzatba. Nem, mert az M_2 állapotgépnek nincs olyan állapotátmenete, ami — bemenetre történne meg.

Aszinkron szorzat: minden állapotátmeneti él lemásolódik a szorzatállapotokra. Ezt két állapotgép egyszerűen elkészíthetjük, ha felvesszük az állapotok Descartes-szorzatát és az átmeneteket úgy rajzoljuk be, hogy a M_1 átmenetei függőlegesen, M_2 átmenetei vízszintesen szerepelnek az ábrán.



4. Teljesítményszámítás és terhelésmodellezés 14p

(kapcsolódó gyak, mo.)

Fejlesztőcégünk build infrastruktúrája 5 projekt automatikus fordításért felelős, és egy koordinátor-szerverből, valamint két további dedikált build szerverből áll. A fejlesztőgárda minden egyes projektből átlagban 15 percenként készít el egy újabb verziót (commit). A commit beküldése előtt a fejlesztő gépén lokális ellenőrzés történik, amely formai hibák miatt a commitot elutasíthatja (pl. kitöltetlen megjegyzés). A formailag helyes commit üzenetet a kliensprogram beküldi a build infrastruktúrának, ahol a koordinátor-szerver fogadja.
Ezután a koordinátor nyilvántartásba veszi, hogy el kell inditani egy fordítási folyamatot (build job), amelynek végrehajtási ideje a projekt méretétől es az aktuális verizóban található hibáktól függően változhat.
A nyilvántartásbavétel részeként a koordinátor egy előzetes becslést végez, és ez alapján eldönti, hogy melyik build szerveren futtatja a job végrehajtását, majd elküldi a build kérést az adott szervernek. A build futtatása mindenképp a fordítandó kód összeállításából, fordításából és jelentésgenerálásból áll. A build eredményeként előálló jelentés minden esetben a koordinátor szerverhez kerül, mely azt feldolgozza (pl. statisztikakészités céljából), és visszaküldi a fejlesztőnek.

A build szerverek átlagos kihasználtsága 62.5%, egyszerre egy-egy jobot tudnak feldolgozni, a többi várakozik. A commitok 25%-a formailag hibás. Formailag helyes commit beküldése esetén az infrastruktúrában a build szervereken történő várakozással együtt átlagosan 10 perc telik el egy commit kérés fogadása és a build eredménye (report visszaküldése) között. A koordinátor 10 kérés párhuzamos feldolgozására képes, egyszerre átlagosan 1 kéréssel foglalkozik.

a Készitse el a fenti rendszer (BPMN) folyamatmodelljét! Ügyeljen arra, hogy az egyes lépések közti adatáramlást jelezze! (4p)

TODO!!!

b. A folyamat mely részet lehet hierarchikus modellezés segítségével egyszerűsíteni? 1p

TODO!!! ???
hierarchikus modellezés: alfolyamatok!
pl. (gondolom, fixme): lokális ellenőrzés, nyilvántartásba vétel, build

c. Egyszerre átlagosan hány commit kérés feldolgozásával foglalkozik az infrastruktúra? Egyszerre átlagosan hány build job fut vagy várakozik a build szervereken (összesen)? 3p

(5 projekt - 25% $_{(1,25)}$ = 3,75 - > 3,75/15 = 0,25)
5 projekt, mindegyiknél 15 percenként egy-egy commit kérés, ezek 25%-a formailag hibás
→ 5 kérés 25%-a 1,25 kérés
→ 5-1,25 = 3,75 kérés 15 percenként
→ 3,75/15 = 0,25 kérés percenként (=X, átlagos átbocsátás)
Little-törvény (egyensúlyi állapotban igaz, amikor λ = X, vagyis egységnyi idő alatt ugyanannyi új felhasználói kérés érkezik a rendszerbe, mint amennyit ezalatt az idő alatt feldolgozott - a koordinátor 10 kérés párhuzamos feldolgozására képes):
X=0,25 kérés/perc
R=10 perc (*"az infrastruktúrában a build szervereken történő várakozással együtt átlagosan 10 perc telik el egy commit kérés fogadása és a build eredménye (report visszaküldése) között"*)
N=X*R (rendszerben tartózkodó kérések átlagos száma (átlapolódás mértéke) = átbocsátás * átlagos rendszerben töltött idő)
=>0,25*10 = 2,5 kérés feldolgozásával foglalkozik az infrastruktúra.

Egyszerre átlagosan hány build job fut vagy várakozik a build szervereken (összesen)?
TODO!!! ???
n=2 build szerver
kihaszn.: U_b=62.5% (egyszerre 1 job/build szerver)
U_i = N_i / n_i (N érték megadja, hogy átlagosan hány kérés tartózkodik a rendszerben, ez esetben az erőforráson belül. Az egyes erőforrásokból több példány is rendelkezésre állhat, ezeken belül feltételezzük, hogy nincs átlapolódás. Az erőforrás tehát egyszerre legfeljebb n_i kérést képes kiszolgálni, ahol n_i az i. erőforrásból elérhető példányok száma)
62.5% = N_i / 2
N_i = 1,25 kérés átlagosan egy build szerveren, összesen 2,5?????
TODO!!! ???

d. Mekkora egy build job átlagos futási ideje? 2p

TODO!!!

e. Miért várakoznak a kérések, ha egyszer az infrastruktúra nincs telítésben? Nevezzen meg legalább két lehetséges okot. 2p
szűk keresztmetszet???
TODO!!! ???

f. Ha egyes projektekre érkező commitok Zipf-törveny szerint oszlanak el, mekkora a legaktívabban fejlesztett projekt commitjainak gyakorisága? 2p
TODO!!! ???

Zipf törvénye - Képlet

$$R_i \sim \frac{1}{i^{\alpha}} \qquad f \sim \frac{1}{p}$$

- R_i – az i. szó rangja
- α – a korpuszra jellemző 1 közeli érték
- f (frequency)
  - gyakoriság
- p (popularity)
  - a szöveg ,,rangja'' (csökkenő sorrendben)

Zipf törvénye – Web dokumentumokra

$$P = \frac{k}{r}$$

- P – hivatkozások (elérések)
- r – rang (1 = leggyakoribb)
- k – pozitív konstans

ZH 2013.11.20 B csoport

Kiskérdések (6x4 pont)

Absztrakció

1) Mit értünk absztrakció alatt? Milyen halmazelméleti alapja van? Adjon példát az értékkészlet finomítására egy, a diszk telítettségét vizsgáló mutató kiértékelésének példáján keresztül.

Absztrakció során elvonatkoztatunk azon tulajdonságoktól, melyek a vizsgálat szempontjából lényegtelenek és csak kiemelt tulajdonságokat veszünk figyelembe.

Halmazelméleti alapja: diszjunkt részhalmazok hozzárendelése elemekhez. (Modellezés bevezető diasor 47. dia (de ez a halmaz*finomítás*, nem az absztrakció!!))

Finomítási példa: {telített, telítetlen} => {10%, 20%, 30%, …, 100%}

Modell ellenőrzés

2) Mit értünk modellellenőrzésen, milyen tipikus céljai vannak? Milyen módon adhatjuk meg a modellezett rendszerrel szemben támasztott követelményeket?

adott egy viselkedési modell és erre vonatkozó következmények, modellellenőrzés során azt vizsgáljuk, hogy a modell megfelel-e a vele szemben elvárt követelményeknek.

- Modellezés validálása*
  - Valóban jól modelleztük a rendszert?
  - Azt modelleztük, amit akartunk?
  - Szimuláció
  - Ellenőrző kritériumok igazolása
- Modell verifikációja
  - Az eredeti rendszer követelményei alapján
  - Ez a cél, a validálás csak feltétel

Előreláncoló rendszer

3) Mit értünk előreláncoló rendszer alatt? Milyen lehetőségek vannak üzleti szabályok kiértékelésénél a konfliktusok feloldására

Következtető mechanizmus, alapja: logikai következtetés vagy üzleti szabályok

Előre láncoló (induktív/produkciós, adatvezérelt)

- A tényekből újabb tényeket képez (produkciós szabály)
- Egy következmény teljesítheti egy szabály feltételrészét
- Analógia: generatív nyelvtan, hatáselemzés
- Ilyenek például a üzleti szabályrendszerek
- Logikai következtetés (vs. üzleti szabály)

Több egyidejűleg aktivált szabály -> Konfliktusban lévő szabályok feloldása

(Üzleti szabály alapú modellezés diasor 6., 8., 18. oldal)

Hátra láncoló rendszer

A csoport: Mit értünk hátraláncoló rendszer alatt?

Hátra láncoló (deduktív, igényvezérelt)

- Egy cél-állítást próbál visszavezetni alaptényekre
- Analógia: parser, diagnosztika

- Ilyen például a Prolog és számos szakértői rendszer

Little törvény

4) Hogyan lehet Little törvény segítségével meghatározni egy Web szerveren a szükséges szálak számát átlagos terhelés esetén? Adja meg milyen jellemzőket kell ismerni ehhez, melyik milyen fogalomnak felel meg!

Egyensúlyi állapotban (λ=X) igaz a Little-törvény:
N = X * R

λ [db/sec] - Érkezési ráta (A vizsgált rendszer határához egységnyi idő alatt érkező felhasználói kérések átlagos száma)
N [db] - Rendszerben lévő kérések átlagos száma (Az átlapolódás mértéke) - ez adja meg a szükséges szálak számát a webszerveren.
X [db/sec] - Átbocsátás (A vizsgált rendszert egységnyi idő alatt elhagyó feldolgozott felhasználói kérések átlagos száma)
R [sec] - Válaszidő (Round-trip time) (A felhasználói kérések által a rendszer határain belül töltött átlagos idő)

Párhuzamos koordináták

5) Hány dimenziót tud kezelni a párhuzamos koordináták módszere? Hogyan jelenik meg két változó közti negatív korreláció a diagramon? (rajzoljon egyszerű példát!)

N dimenziót tud kezelni
A negatív korreláció pedig szimplán úgy jelenik meg, hogy lesz a diagrammon egy “X”, tehát egymást keresztező vonalak.

Átlagszámítás - regresszió

6) Mi az elvi különbség az átlagszámításra építő és a regressziós módszerek közt? Melyik milyen távon tud előrejelezni? Mi a korrelációs együttható szerepe a regresszió esetében?

FIXME: Az átlagszámításra épülők a mért eredmények alapján becsülik meg a továbbiakat, a regressziós módszerek pedig a bemeneti attribútumok és egy függvény segítségével becsülik meg az értékeket.
A regressziósak előre tudják jelezni pl az exponenciális változásokat is, ezzel ellentétben az átlagszámításra alapulók ezt nem tudják figyelembe venni. (viszont kevés mért adat esetén pl a lineáris regresszió reltérhet a későbbi tényleges értékektől)

Mi a korrelációs együttható szerepe a regresszió esetében?
a változó becsült és tényleges értékének kapcsolata -> ??? NEM EZ A JÓ VÁLASZ RÁ ESETLEG, VALIDÁLÁST KÉREK
Terheléselőrejelzés 24. dia

Lineáris regresszió (folyt.)

- Korrelációs együttható (négyzete)
  - a változó becsült és tényleges értékének kapcsolata
  - 0 és 1 közti érték
  - 0: nincs kapcsolat
  - 1: függvényszerű kapcsolat

$$R^2 = \frac{\sum_{t=1}^{n}\left(F_t - \overline{Y}\right)^2}{\sum_{t=1}^{n}\left(Y_t - \overline{Y}\right)^2}$$

- Példa: E-mail szolgáltatás, 8 hétig mérjük a csúcsterhelést.

Hét	1	2	3	4	5	6	7	8
Max. terhelés (email/perc)	420	410	437	467	448	460	507	514

Hogyan közelíthető a terhelés változása?

Mekkora a korrelációs együttható?

(azelőtti dia:

Lineáris regresszió

- Egyszerű lin. függvény illesztése az adatokra
  - nem vár alapvető változást a rendszer viselkedésében

$$Y = a + bX$$

- Legkisebb négyzetek módszere
  - keressük azokat az *a,b* paramétereket, amelyekre

$$SSE = \sum_{t=1}^{n} \varepsilon_t^2 = \sum_{t=1}^{n}\left(Y_t - F_t\right)^2$$ minimális (Sum of Squared Errors)

- cél:

$$\sum_{t=1}^{n}\left(Y_t - F_t\right)^2 = \sum_{t=1}^{n}\left[Y_t - \left(a + bX_t\right)\right]^2$$

)

1. feladat - Teljesítményszámítás és terhelésmodellezés (10p)

Fejlesztőcégünk build infrastruktúrája 4 projekt automatikus fordításáért ill teszteléséért felelős, és 3 build szerverből valamint 1 koordinátor szerverből áll. A fejlesztőgárda minden egyes projektből átlagban 8 percenként küld be (commit) egy újabb verziót. Minden commit hatására egy koordinátor szerver nyilvántartásba veszi, hogy el kell indítani egy fordítási folyamatot (build job), amelynek végrehajtási ideje a projekt méretétől és az aktuális verzióban található hibáktól függően változhat; ha van tétlen build szerver, azonnal el is indítható a job. Egy build szerver egyszerre egy build jobot futtathat, de mihelyst felszabadul, átvesz egy várakozó jobot (ha van) a koordinátortól. A koordinátoron történő várakozással együtt átlagosan 6 perc telik el egy commit és a build eredménye között. A koordinátoron egy időben átlagosan 1.5 kérés várakozik.

a) Egyszerre átlagosan hány build job fut vagy várakozik az infrastruktúrában? (2p)

1,5 + 3 = **4,5** (Erre mondták megtekintésen, hogy **nem jó (!!!)** és el se fogadták.)
Mert 1,5 átlagosan mindig várakozik, és mivel a 3 build szerver mindig dolgozik, így ott mindig lesz 3 db.
(A 1,5 nem a koordinátorra várakozik? itt csak a build szerverre kérdez rá!)

Little törvény alapján
R = 6 perc
X = 4 projekt / 8 perc = 0,5 projekt / perc
N = X*R = 0,5 * 6 = **3** build job van egyszerre a rendszerben

b) mekkora egy build job átlagos futási ideje és a build szerverek kihasználtsága?(4p)
FIXME: build job átlagos futási ideje:
R_{bs} = ?
N_{bs}(build szerverek átlagos terhelése) =1,5 kérés ← *"A koordinátoron egy időben átlagosan 1.5 kérés várakozik"*, ezért ezt vettem a 3 build szerver átlagos terhelésének, de FIXME
N inkább jobban hangzik úgy, hogy rendszerben tartózkodó kérések átlagos száma (átlapolódás mértéke)
X_{bs}(bs áteresztő képessége)= 3 kérés / 6 perc = 1 kérés /120 s
$R_{bs}=N_{bs}/X_{bs}$=1,5/(1/120)=120*1,5= 180 sec

build szerverek kihasználtsága: U=X/Xmax= 3/3 -> 100% (a) rész, "a 3 build szerver mindig dolgozik")
??? ez így biztos jó??

c) Burstös- e a terhelés? (2p)
Igen, mert van benne várakozás.

d) Ha az egyes projektekre érkező commitok Zipf-törvény szerint oszlanak el, mekkora a legaktívabban fejlesztett projekt commitjainak gyakorisága? (2p)

4 projektünk van így: x + x/2 + x/3 + x/4

FIXME: Elvileg a legaktívabb 2szer gyakoribb mint a második, 3szor gyakoribb mint a harmadik stb. De ötletem nincs, hogy itt ez elfogadható-e válasznak.
(hát a diasorban található tanszéki oldal példánál a második alig kisebb mint az első, a harmadik pedig alig kisebb mint a második)

Én ebből gondoltam az eredeti választ:
A Zipf-törvény azt állítja, hogy egy természetes nyelv egyes részeiben, egy szó előfordulási gyakorisága fordítva arányos a frekvencia (előfordulási) táblában levő rangjával. Így, a leggyakoribb szó közel kétszer gyakoribb, mint a második leggyakoribb szó, és háromszor gyakoribb, mint a harmadik helyen lévő, stb.
Hasonló törvényszerűség (eloszlás) figyelhető meg, nem csupán a szövegtestekben, hanem más területeken is, mint például: különböző országokban a városok lakosságának eloszlásánál, vállalatok méreteinél, jövedelem eloszlásnál, stb.

A diáknál (ahogy én látom) nagyjából így következnek az értékek:
Eredeti (x) => x/2 => x/3 => x/4 => x/5 … Szóval passzolok melyik a helyes megoldás, de ahogy én látom a diákban ez helyes amit írtam.

Itt nem az f ~ 1/p képletet kérdezték, ahol p rang a leggyakoribb esetén 1?

---
x + x/2 + x/3 + x/4 = 4 * 1/8 (1/8 magyarázata: *"átlagban 8 percenként küld be"* → tehát átlagosan egy perc alatt 1/8 kérés jön be)
Tehát:
x = 0,24
SZERINTEM:
1. projekt: tfh x alkalommal fordul elő, azaz 2x gyakrabban, mint a 2., 3x gyakrabban, mint a 3., és 4x gyakrabban mint a 4.
2. projekt: x/2 alkalom
3. projekt: x/3 alkalom
4. projekt: x/4 alkalom

Így a projektek a következőképpen alakulnak:
1. leggyakoribb projekt: 0,24 alkalom
2. leggyakoribb projekt: 0,12 alkalom
3. leggyakoribb projekt: 0,08 alkalom
4. leggyakoribb projekt: 0,06 alkalom
SZUMMA: 0,5 (ami az összes projektre vonatkozó commitek gyakorisága)
---

2. feladat - Modellezés adatfolyamhálóval, taxonómia építése (12p)

Modellezzük az I. feladatban említett cég build folyamatának egyszerű verzióját adatfolyamhálóval. az egyszerűség kedvéért számoljunk 2 build szerverrel, és egy felhasználóval, akitől a tetszőleges számú kérés érkezhet. Az egyes projekteket ebben a modellben nem kell megkülönböztetni.
a) A felhasználótól beérkező "commit" kérés hatására a koordinátor szerver elküldi a kérést egy szabad build szervernek, illetve "overload" üzenet mellett elutasítja, ha nincs szabad szerver. A build szerverek visszajeleznek egy kérés elfogadásáról, valamint a build lefutásáról. Utóbbiról a koordinátor szerver is értesíti a felhasználót.

TODO!! RAJZ?

b) Alakítsa át a modellt a következőképpen: csak egyetlen build szerver van, viszont néhány build "public build" (nightly build) minősítést kap (ez már a commit kérésből kiderül). Ezeknél a koordinátor szerver a sikeresen lefutott build publikusan elérhetőségét tartalmazó értesítést küld a vállalati portálnak (új csomópont). Public build kérést akkor sem utasít el a rendszer, ha épp nincs szabad build szerver: ehelyett várakozási sorba kerül a koordinátor szerveren. Tegyük fel, hogy egyszerre egy ilyen kérés tartózkodhat a rendszerben. (3p)

TODO!! RAJZ?

c) Készítsen olyan taxonómiát, amely leírja a fejlesztés során keletkező "termékeket". Ide érthetjük a követelmények szöveges vagy modellszerű dokumentációját, mely leegyszerűsített fejlesztési módszerünkben a fejlesztés bemenetét képezi. Kimenetként megjelenik a (kézzel vagy automatikus generálás útján előállított) kód, a felhasznált külső könyvtárak, a kész buildek, a követelményekből származtatott tesztesetek és az egyes egységeken valamint a teljes szoftveren értelmezett teszteredmények, valamint a kész szoftver dokumentációja.

Utóbbi tartalmazhat generált elemeket (pl. javadoc), szabad szöveges leírást és modelleket is, ezek a követelmény modellek finomításával állnak elő. Jelezze a taxonómiában felvett keresztélekkel az egyes elemek közti kapcsolatot (4p)

TODO!! Tudna valaki ide írni egy példa taxonómiát?

Valaki itt megtenné, hogy leírja itt mit vártak volna el a "kapcsolatok" között?
Én olyanokra gondolnék, hogy a fejlesztés **bemenete** a dokumentációk, a **kimenete** a kód, a buildek és társai. Hogy a kódot lehet többféleképpen **generálni**, követelményekből **származtatjuk** a teszteseteket.
Bármi hasonló kapcsolat, ami kiolvasható a szövegből, és felkerült már az ontológiára.

3. feladat - Szolgáltatásbiztonság és kísérlettervezés (14p)

Hajókon használt tengeráramlás mérő berendezést gyártunk, amely az Mi (i=1,2,...,6) modulokból áll. Ebből az M1 modul egy szonda, amelyik időnként eltömődik uszadékkal (ilyenkor tisztítani kell), ez azonban az alkatrész korától, napszakától és egyéb külső tényezőktől függetlenül teljesen véletlenszerűen, átlag 1.5 naponta egyszer következik be. Ismert, hogy az M2,M3,M4 modulok tetszőleges időpillanatban 99,9% míg az M5 és M6 modulok 99.999% valószínűséggel működnek helyesen.

a) Mekkora az M1 modul meghibásodási rátája az üzembe helyezés után 1, 2, ill. 3 nappal? (2p)
MTTF= 1,5 nap
meghibásodási ráta (λ) az MTTF-érték reciproka:
λ = 1/MTTF =2/3

r_i(t_i) - az R_i erőforrástípus megbízhatósági időfüggvénye (t_i azt adja meg, hogy mennyi ideig használta egy taszk az R_i erőforrástípus egy példányát) → $r_i(t_i) = e^{-lambda\_i * t}$

r(t=0)=1
r(t=1)=2/3
FIXME - ami volt zh gyak órán az alapján így kéne,de kérdés hogy jó e itt is így
r(t=2)= 2/3*2/3 ???????
r(t=3)= 2/3*2/3*2/3 ??????????????????

a válasz: igen, jó így

akkor most a kommentben lévő megoldás a jó, vagy ami itt van leírva??

Valaki meg tudná mondani, hogy mi a különbség a meghibásodási ráta és a megbízhatósági fgv között?

Megbízhatósági függvény : $r(t) = e^{-lambda*t}$
Meghibásodási ráta: lambda(t) = 1-r(t)

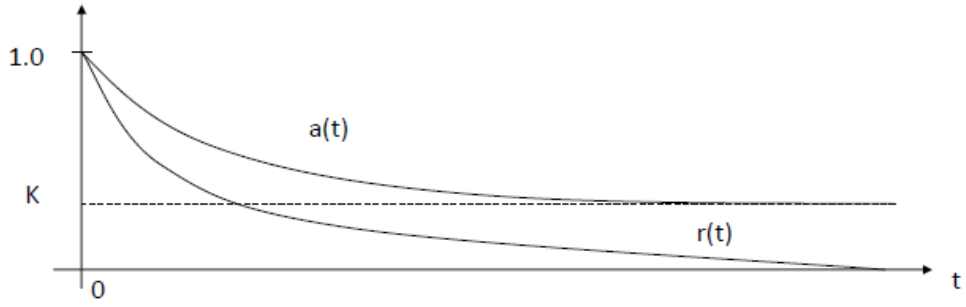
NA MOST AKKOR MI VAN??? Melyik a jó?? mindkettő.

b) ábrázolja diagramon és adja meg képlettel az M1 modul megbízhatósági függvényét!

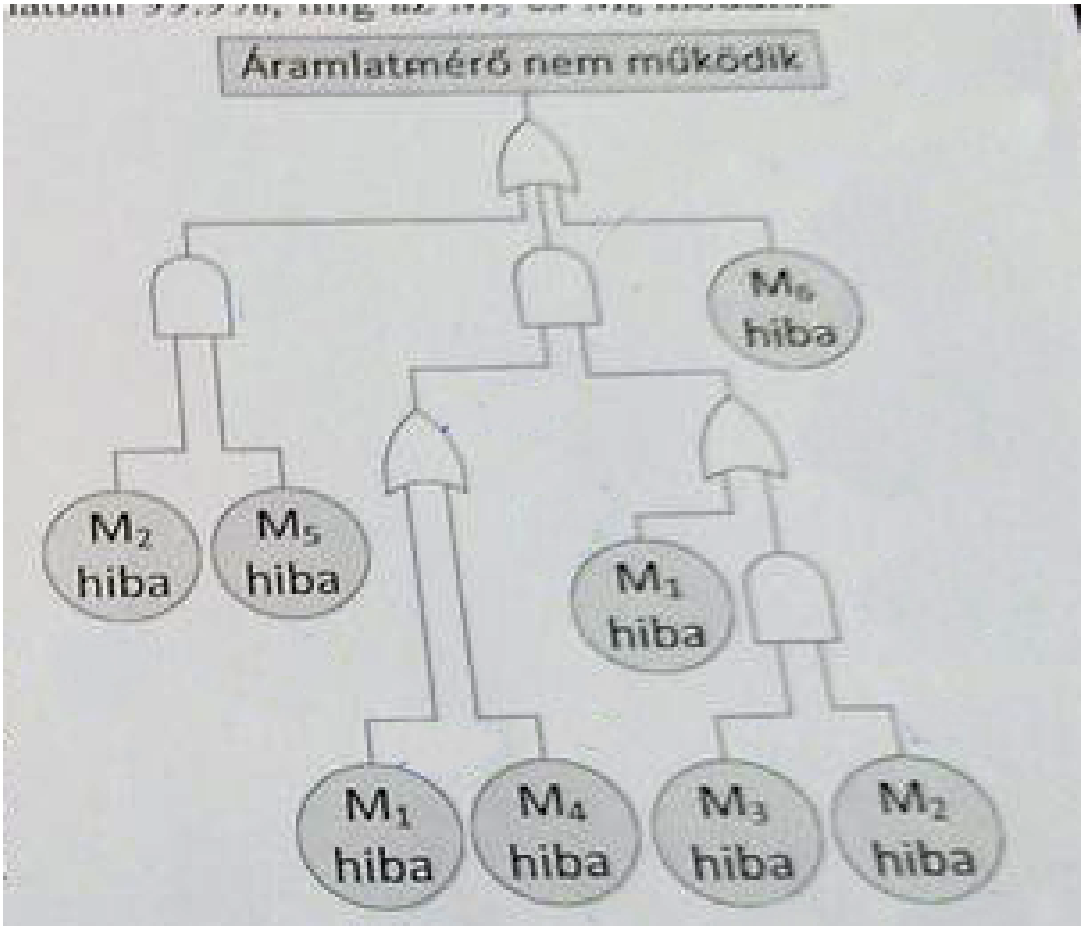
$r(t) = e^{-lambda*t}$

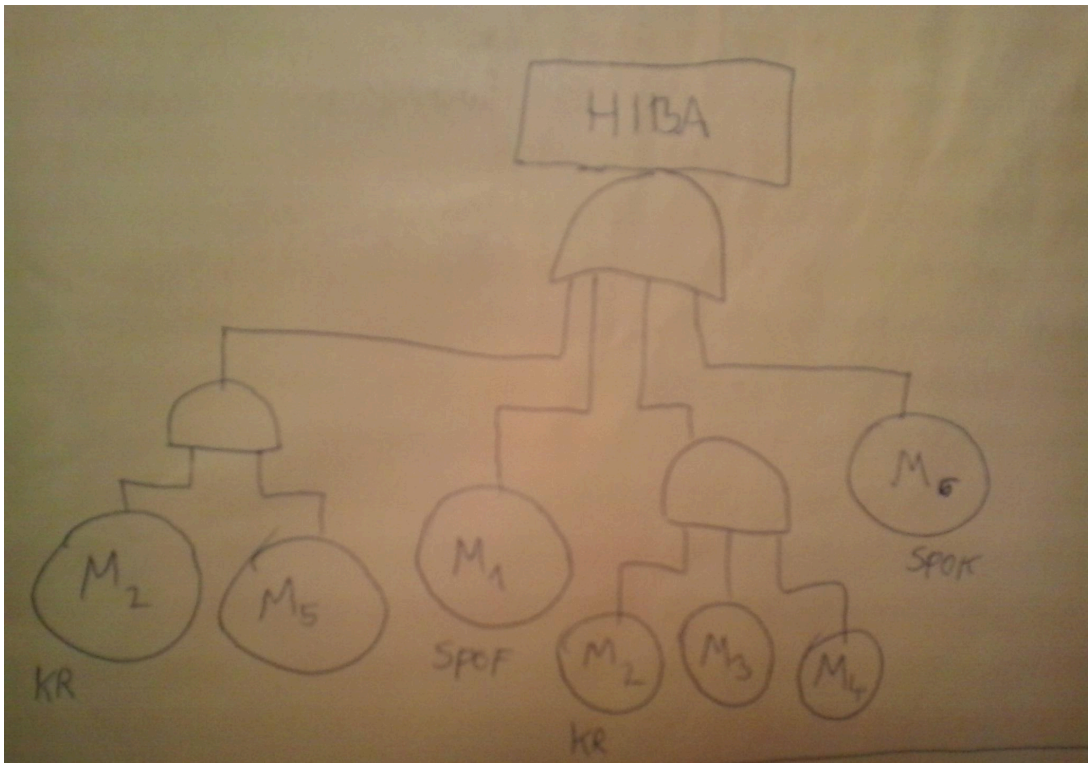
Valószínűség időfüggvények

- megbízhatóság: (reliability)
  r(t) = P{∀ t' < t: s(t') ∈ U} (nem hibásodhat meg)
- rendelkezésre állás: (availability)
  a(t) = P{ s(t) ∈ U } (közben meghibásodhat)
  - készenléti tényező: $K = \lim_{t\to\infty} a(t) = \frac{MUT}{MUT+MDT}$



c) Az oldalsó ábrán látható hibafasz teremt összefüggést az áramlatmérő és moduljainak meghibásodásai közt. Hibafasz-redukció alkalmazásával azonosítsuk az egyszeres hibapontokat és kritikus eseményeket! (3p)





d) A cél, hogy a teljes mérőberendezés rendelkezésreállása "két kilences" legyen. A redukált hibafasz alapján állapítson meg követelményt M1 rendelkezésreállására! Milyen közelítések, feltételezések szükségesek a számításhoz? (3p)

Egész rendszer: P_0 = 99% -> 10^{-2}
M2,M3,M4: P_1 = 99.9% -> 10^{-3}
M5,M6: P_2 = 99.999% -> 10^{-5}
M1: P_3 = ?
$P_0 = P_1P_2+P_3+P_1P_1P_1+P_2$
$P_3 = P_0-P_1P_2-P_1P_1P_1-P_2$

$P_3 = 10^{-2}-10^{-8}-10^{-9}-10^{-3} = 9{,}999999*10^{-3} \cong 10*10^{-3} = 10^{-2}$

Itt látszik, hogy a 10^{-2} lesz a legnagyobb, a többi meg mindig kicsit elvesz, tehát közelítéssel mondhatjuk, hogy 10^{-2} az értéke a P_3-nak

e) ennek a követelménynek a teljesítéséhez legfeljebb mekkora átlagos javítási (tisztítási) idő engedhető meg az M1 szondára? (1p)

egy évben 3.65 nap javítási idő. (1%)
az a kérdés, hogy ÁTLAGOSAN (tehát 1 alkalomra levetítve) mekkora jav. idő engedhető meg.
A fent kiszámolt 3,65 nap az az egy évben összesen megengedett idő javításra.
1,5 naponta romlik el → 365/1,5=243,33 javítás évente -> egy javításra (3,65nap/243,33) = 0,015 nap = 0,015*24*60 perc = **21,6 perc jut átlagosan**

f) **a flotta legutóbbi 400 szondatisztításának ideje átlagosan ezen célérték fele volt, de sajnos a tapasztalati szórás a tapasztalati átlag tízszeresére rúgott az outlierek miatt. Mennyire biztosan állíthatjuk, hogy a várható tisztítási idő megfelel a követelménynek? (3p)**

EZT MINDENKI CSAK SAJÁT FELELŐSSÉGRE HASZNÁLJA (nem biztos, hogy így kell)!!!!

Előző feladatban, egy évnél számítva: 3.65 nap (=87.6 óra) kell, de ennek a fele volt, tehát 43.8 óra.
x = 43.8 óra
s = 10x = 438 óra
σ = s/sqrt(t) = 438 / sqrt(400) = 21.9

43.8 ± 2 * 21.9 = 87 óra, ami pont a megengedett.

Ez alapján 95% biztossággal mondhatjuk, hogy jó. (2*σ miatt)

Másik megoldás
A fenti feladat(e.) saját megoldása (21,6 perc) ezt számoltuk:
21,6 perces intervallumba kell essen:

t=400
R=21,6 perc/2 = 10,8 perc
d=108 perc

68% = d / gyök(t) = 108 / 20 = 5,4 (kb 5)
95% = 2d / gyök(t) = 216/20=**10,8**

Ami +- 10,8 intervallum (pont 21,6)
tehát 95% biztonsággal mondhatjuk hogy megfelel a követelményeknek.
(a százalékok azaz konfidencia-intervallumok adottak)

2. feladat (8 pont) Terheléses dolgok

„Közért"
Amikor a vásárló belép a közért területére, elvesz egy bevásárlókosarat, és fizetés után távozva teszi vissza a helyére. Minden pulthoz (kenyér, tejtermék, hús, stb.) sokan férnek hozzá, csak a zöldség-mérlegelés és a végén a kassza nem párhuzamosítható, ezeknél sor áll (minden mérleghez és kasszához külön-külön). Egy vásárló egy bevásárlás folyamán átlagosan 3,8-szor megy az elektronikus zöldségmérleghez megmérni a zacskóba kiválogatott zöldséget (egyenként 15 másodperc mérleghasználati idő), és persze a végén egyszer fizet (ami átlagosan 1,5 percig tart, sorban állást nem számítva).

a) Megfigyeltük, hogy csúcsidőben percenként 4 vevő lép be a közértbe, és a 50 bevásárlókosárból egyszerre átlagosan 37,3-at használnak. Átlagosan mennyi időt tölt bent egy vásárló csúcsidőben? (2 p)
N = 37,3 [k]
X0 = 4 [k/min]
ebből
R = N/X0 = 9,33 [mim]

b) Minimálisan hány mérleg és hány pénztár kell ahhoz, hogy a közért el tudja látni a forgalmat? (2 p)
Mérleg szolgáltatásigénye
$D_m = V_m * S_m$ = 3,8 [k/trz] * 15 [s/k] = 57 [s/trz]

ebből mérleg áteresztőképessége ⇒=
$X_m = 1 / D_m$ = (1/57 [k/s]) / (1/60 [s/min]) -átváltás 1/s-ből 1/min-be- = 60/57 [k/min] = 1,05 [k/min]

Majd ennek a teljes rendszerhez képest viszonya:

X_0/X_m = (4 [k/min]) / (1,05 [k/min]) = 3,8, tehát a mérleg áteresztése 3,8-szerese a boltének. vagy inkább: 3,8 mérlegnek akkora az áteresztése, mint a boltnak, tehát min. 4 mérleg kell

Hasonló számítással a kasszára:
X0/Xk = 6
Ez kifejtve:
Kassza szolgáltatásigénye
$D_k = V_k * S_k$ = 1 [k/trz] * 1,5 [min/k] = 1,5 [min/trz] // minden vásárló egyszer fizet, ezért V_k = 1 [k/trz]

ebből kassza áteresztőképessége ⇒
$X_k = 1 / D_k = 1 / (1{,}5$ [k/min]) = 2/3 [k/min]

X_0/X_k =(4 [k/min]) / (2/3 [k/min]) = 6,tehát 6 kassza kell, hogy a bolt áteresztését el tudja látni

c) Melyik erőforrás jelenti a szűk keresztmetszetet? (1 p)
~~*Az előző feladatban kiszámoltak szerint egyik erőforrás sem szűk keresztmetszet, de persze el is ronthattam, fixme… <- NEM*~~
~~*Szerintem itt annyi lenne, hogy mivel a mérlegek percenként 1.05 kérést (4X1.05 total), a kasszák percenként 2/3 kérést (6X(2/3) total) tudnak kiszolgálni, kijön, hogy mindkettő erőforrás 4.2 kérést tud kiszolgálni percenként (total) tehát bármelyik tekinthető szűk keresztmetszetnek (Percenként 4 vásárló lép be tehát mindet kiszolgáljuk ezekkel is.). Ha kicsit jobban belegondolunk akkor viszont ha 1et vennénk mindegyikből újként hozzá, akkor a kasszák lennének a szűk keresztmetszet. (Remélem jól gondolkodok.) <- érdekes*~~

~~Szerintem józan paraszti ésszel úgy lehet megfogni a kérdést, hogy nekünk 3,8 mérleg kellett meg 6 kassza, de ugya fél mérleget nem árulnak. Viszont ha már egy vásárlóval többen lépnek ugyanannyi idő alatt, hosszútávon torlódás / visszautasítás lesz (arról nem volt szó, hogy zárt vagy nílt a rendszer), mert több kassza kéne (hamarabb telítődik, mint a mérleg, de mindkettő közelebb van hozzá).~~

~~*Megj.: az előző kettőt azért hagytam ott, mert korábban 4 mérleg 7 pénztár megoldás volt, és ott valszeg a mérleg lesz a para, de ez durva becslés, ki kellene számolni!*~~

Kiadott segédletből:
“Ki kell tehát számolnunk az egyes erőforrások átbocsátásából a rendszer átbocsátásának elméleti maximumát, majd az így kapott értékek legkisebbikét kell kiválasztanunk, ez lesz a rendszer átbocsátóképessége, az értékhez tartozó erőforrás(ok) pedig a rendszer szűk keresztmetszete(i)”

1 db Kassza átbocsátás (Xk): 2/3 kérés/perc
1 db Mérleg átbocsátás (Xm): 1,05 kérés/perc

*6 kassza => 6 * 2/3 => 4 kérés / perc*
*4 mérleg => 4 * 1,05 => 4.2 kérés / perc*

Azaz a 6 kassza a szűk keresztmetszet.

d) Négyféle zöldséget árulunk valójában: paradicsom, paprika, uborka és répa. Zipf-eloszlással írható le, hogy egy zacskónyi lemérlegelt zöldség ezek melyike. A boltvezető terve, hogy a négy mérleget fixen egy-egy zöldséghez rendeli; így nem kell a megfelelő gombot keresgélni, és 10 másodpercre csökken az átlagos mérleghasználati idő. Jó-e az ötlet? (3 p)

Ahogyan számolni kellene:

Tehát Zipf-eloszlás, az összes gyümölcsvásárlás eloszlása gyümölcsökre osztva $\frac{k}{x}$azaz 3.8 = $\sum_1^4 k/x$

X a paradicsom
X/2 a paprikáé
X/3 az uborka
X/4 a répa **vásárlásának darabja.**
(ezek összege) = 3.8 db.

Így kijön k-ra 1,824

Paradicsom 1,824
paprika 0,912
uborka 0,608
répa 0,456

ekkor a mérleghasználata a paradicsomra: XbejövőEmberABoltbaPercenként * ParadicsomDarab * ParadicsomMérlegHasználatiIdeje = 4 (db/min) * 1,824 (db) * 1/6 (min) = 1,216 ami **121%-os kihasználtság, tehát nem jó ez, mert túl sokan használnák a paradicsom mérleget!**

Elméleti kérdések: 1

alternatív kidolgozás: https://wiki.sch.bme.hu/Rendszermodellez%C3%A9s_kisk%C3%A9rd%C3%A9sek,_2009.

Absztrakció, finomítás

1. Mi az absztrakció, mint művelet ellentéte? Mutasson rá egy példát!

Az absztrakció ellentéte a **finomítás**. Több információ, több tudás.
Példa: Értékkészletfinomítás:

- {hétköznap, hétvége} -> {Hé, Ke, Sze, Csü, Pé, Szo, Va}

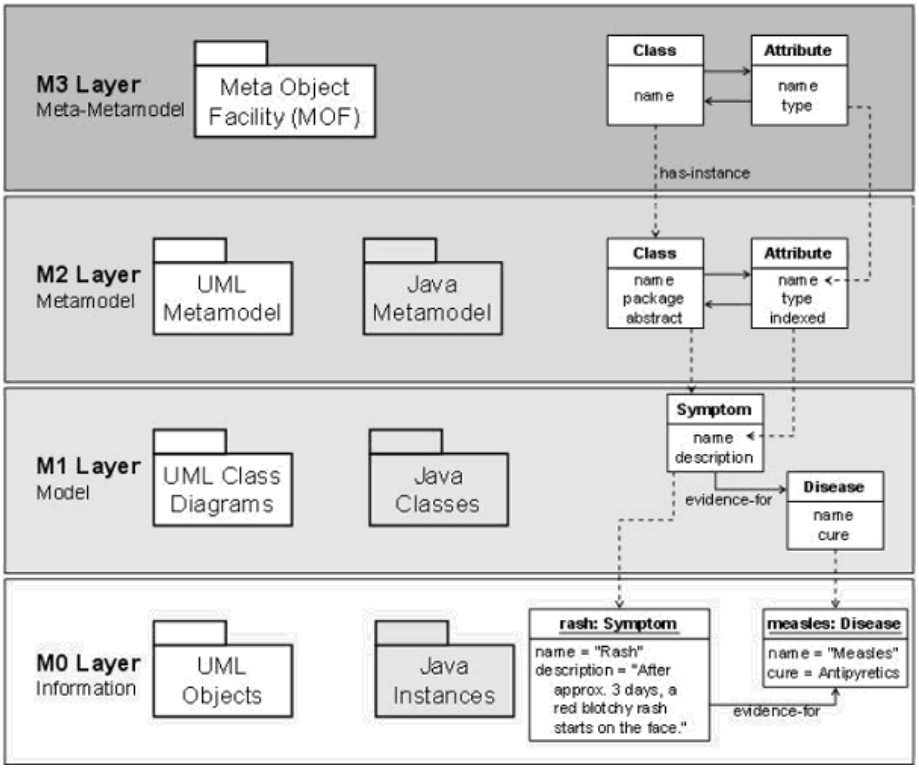
Felhasználói viselkedés gráfja

2. Milyen mérőszámok nyerhetők ki a felhasználói viselkedést leíró gráfból?

- *Revenue Throughput*: az oldal által elért átlagos bevétel
- *Potential Loss Throughput*: mennyibe kerül a szolgáltatás kiesése egy adott időszakban
- *Visit Ratio*: átlagosan hányszor vesznek igénybe egy adott szolgáltatást (egy session alatt)
- *Buy to Visit Ratio*: átlagosan hányszor vásárolnak egy session alatt o (tényleges eladási tanzakció)
- *Average Session Length*: egy session átlagosan hány szolgáltatást vesz igénybe (nem időt mér)
- mindezen mérőszámok változása, ha a bemenő paraméterek megváltoznak (pl. egyes átmenetek valószínűségei)

Idén nem volt UML, de aki akarja, tanulhatja...
3. Adjon meg két UML diagramot, amelyek közül az egyik a másik metamodellje!

Metamodell = egy modellezési nyelv modellje
UML objektum diagram -> osztálydiagram
Adatbázis tábla -> relációs adatbázisséma
XML dokumentum -> XML séma (vagy DTD)



Osztály diagram az objektum diagram metamodellje (fixme)
XML, DTD

Modellellenőrzés

4. Mit értünk modellellenőrzés alatt?

A modellellenőrzés során ellenőrizzük, hogy modell megfelel-e a formális követelményeknek. Pl.:

- Deadlockmentesség
- Biztonsági kritériumok
- Vezérelhetőség
- ...stb

(pl.: Dinamikus állapottér bejárással)

5. Adjon meg legalább két különböző jellemzőt, amely egy UML osztálydiagram asszociációján fel lehet tüntetve!

Role name, multiplicitás, navigálhatóság (mindegyikről van target és source)

6. Milyen elemkészlete van az UML aktivitás diagramoknak?

- rounded rectangles represent **actions**;
- diamonds represent **decisions**;
- bars represent the start (**split**) or end (**join**) of concurrent activities;
- a black circle represents the start (**initial state**) of the workflow;
- an encircled black circle represents the end (**final state**).

pl: http://en.wikipedia.org/wiki/File:Activity_conducting.svg

Vezérlési hiba üzleti folyamatokban

7. Milyen tipikus vezérlési hibákat ismer üzleti folyamatokban? Mutassa be ezeket egy-egy példán!

- Decision – Merge / Fork – Join párok összekeverése:
    - Fork – Merge: nincs szinkronizáció
    - Decision – Join: holtpont
- Ciklusok esetén, ha nincs a ciklusból kivezető él: livelock
- Többszörös összeköttetés: Nehezen átlátható, ha nem különböztetjük meg az adat és a vezérlés áramlást
- Kilépés különböző ágakból ("megkerüljük" a decision utáni merge-t):
    - Döntésnél holtpontot okozhat
    - Párhuzamosításnál megengedett
- Események kezelése: Eseményfeldolgozó logika vezérlésként lekódolva (többszörös start). Helytelen, mert az összes Start elem egyszerre aktív!
- Terminálás: Párhuzamos ágak helyett az egész folyamat leáll.
- Rossz interfész definíció! Nemdeterminizmus / deadlock

fállapotBPMN

8. Milyen főbb elemtípusokat ismer a BPMN szabvány?

- Esemény (kör/gyűrű)
    - Állapotváltozás
    - Ok-hatás
    - Eseménytípusok: Start, Intermediate, End
- Tevékenység (gömbölyített téglalap)
    - Atomi/összetett
    - Taszk/alfolyamat
- Átjáró (rombusz)
    - Szekvencia
    - konvergencia/divergencia
- Szekvencia (nyíl)
    - Tevékenységek sorrendje a folyamatban (nincs vezérlési folyamat a BPMN-ben)
- Üzenet (szaggatott nyíl)
    - Két független folyamat résztvevői közötti információcsere
- Asszociáció (pontozott nyíl)
    - Adat, szöveg stb. hozzárendelés

http://en.wikipedia.org/wiki/Business_Process_Model_and_Notation

Adatfolyamháló

9. Milyen főbb elemekből áll egy adatfolyamháló? Hogyan írhatjuk le a háló állapotát?

Adatfolyamháló (DFN): egy hármasból áll (N, C, S)

- N: csomópontok halmaza
- C: csatornák halmaza
    - I: bemenő csatornák
    - O: kimenő csatornák
    - IN: belső, csomópontok közötti csatornák
- S: Állapotok halmaza

Adatfolyam csomópontok: $n = (I_n, O_n, S_n, s_n^0, M_n, R_n)$

- I_n: bemenő csatornák halmaza
- O_n: kimenő csatornák halmaza

- S_n: csomópont állapotok halmaza
- s_n^0: csomópont kezdőállapota, sn0 Sn
- M_n: tokenek halmaza
- R_n: tüzelések halmaza

Tüzelések halmaza: Rn = (s_n,X_{in},s'$_n$,X_{out},pi)
- s_n: tüzelés előtti és utáni állapotok, s'$_n$
- X_{in}: bemenő leképzés, Xin : In Mn
- X_{out}: kimenő leképzés, Xout : On Mn
- pi: tüzelés prioritása

Modellfinomítás

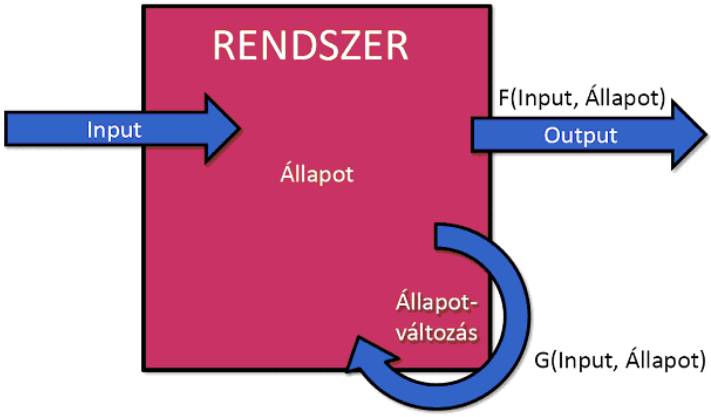
10. Milyen modellfinomítási lépések lehetségesek adatfolyam hálókban?

Értelmezési tartomány finomítás
- token halmaz finomítás,
- állapothalmaz finomítás,

Struktúra finomítás
- csomópont helyettesítése adatfolyam alhálóval

Klasszikus rendszerelmélet

11. Mi a klasszikus rendszerelméleti automata-modell?



Erőforrásigény

12. Hogyan lehet megadni a rendszer erőforrásigényét a funkciók statikus erőforrásigénye alapján? Miért „csal" ez az egyszerű képlet?

$$E[terhels_R] = \sum_{|F|}(gyakorisg_F * E[terhels_{F,R}])$$

Azért csal, mert nem additív a terhelés (nincs beleszámítva a taszkváltás, cache frissítés…) és lehetnek kiugró erőforrásigények.

Nyílt vs zárt modell

13. Teljesítménymodellezésnél mit nevezünk nyílt és mit zárt modellnek? Hogyan határozható meg a szűk keresztmetszet ezekben?

Teljesítménymodellezés.pdf 23.diától

Nyílt modell:
- Nincs explicit korlát a rendszerben lévő kérésekre

- $\text{Max}\{D_{i.\text{erőforrás}} = S_{i.\text{erőforrás}} * (V_{i.\text{erőforrás}} / N_{i.\text{erőforrás}})\}$

(Az i-edik erőforrásra: Di: egy tranzakció átlagos szolgáltatásigénye; Vi: a tranzakció átlagos erőforrásigénye; Si: egy használat átlagos erőforrásigénye; Ni (ezt a nyílt~zárt részről a diából): konkrét szerverek száma a i-edik erőforrásból (pl 3 darab DB szerver..))

Zárt modell: 0

- Felső korlát a kérések számára vonatkozóan
- $\min\{\text{maximális kérések száma} / D_{total}, 1 / D_{max}\}$

Little-törvény

14. Mit mond ki a Little-törvény (magyarázattal)?

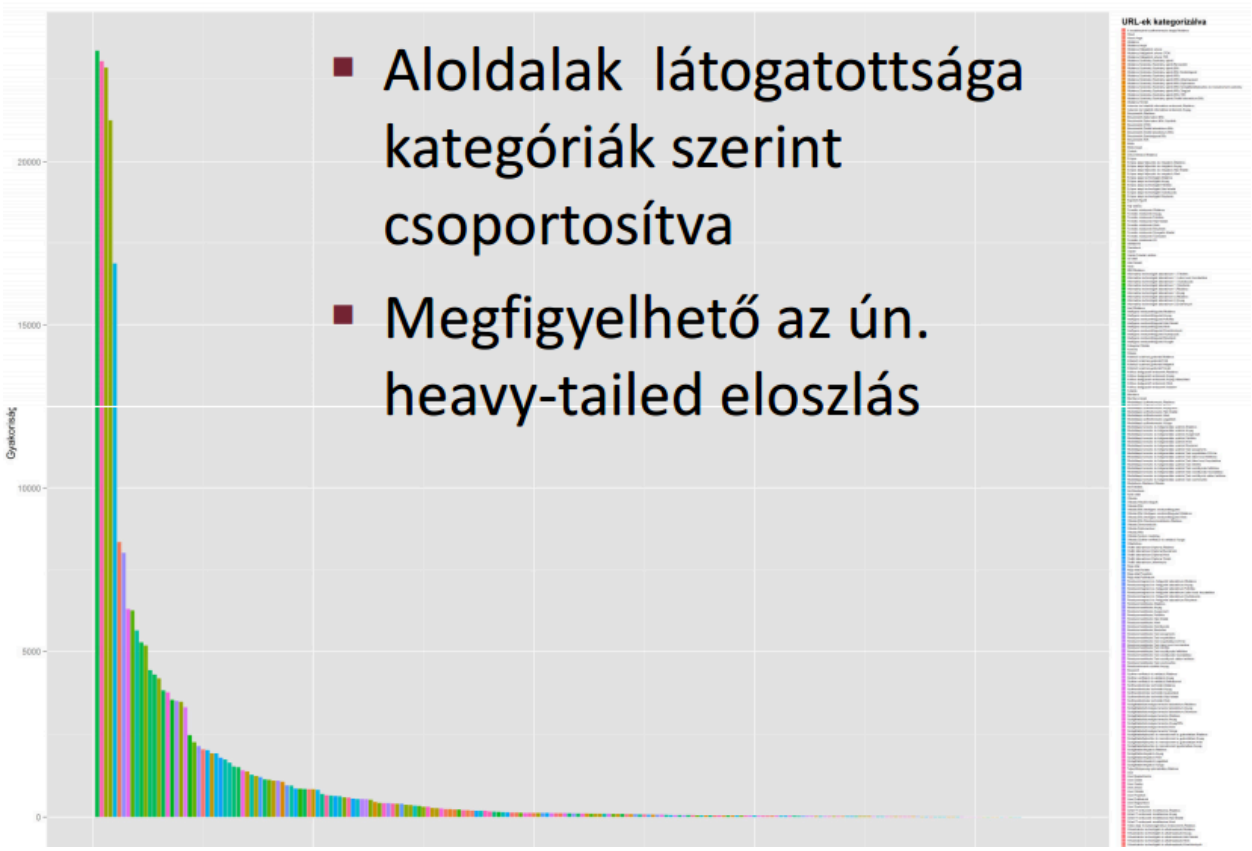
A rendszerben lévő kérések száma (N) megegyezik az érkezési ráta/átbocsátás (X) és a rendszerben töltött átlagos idő (R) szorzatával → N = X * R
Egyensúlyi állapotban igaz a Little-törvény (Ha $\lambda = X$, vagyis egységnyi idő alatt ugyanannyi új kérés érkezik a rendszerbe, mint amennyit ezalatt a rendszer feldolgozott)
(The long-term average number of customers in a stable system *L* is equal to the long-term average effective arrival rate, *λ*, multiplied by the (Palm-)average time a customer spends in the system, *W*; or expressed algebraically: *L* = *λW*.)

Valószínűségi eloszlás

15. Mit értünk az alatt, hogy egy valószínűségi eloszlás „heavy tailed”? Hogyan kapcsolódik ez Zipf törvényéhez?

„Heavy tailed” eloszlás: nem elhanyagolhatóak a ritka értékek sem (a legritkább esettel is számolni kell). (Pl. szavak előfordulása, weboldal aloldalainak látogatottsága.)

A Zipf-törvény értelmében a természetes szövegekben a szavak előfordulási gyakorisága is heavy-tailed eloszlást mutat. Később bebizonyosodott, hogy nem csak nyelvi szövegekre igaz, például weboldalunk oldalainak látogatottságára is igaz a Zipf-törvény (példa: kapacitástervezés ea/13.slide)



TPC-App

16. Mit mérhetünk a TPC-App benchmark segítségével? Milyen főbb részekből áll a TPC mérési konfiguráció? (Benchmarking diasor 30-31. dia)

Együttesen mérhetjük vele az alkalmazás- és a webszervert.

A TPC-App konfiguráció főbb részei:

- Remote Business Emulator (RBE)
- Üzleti funkciók megjelenítése
- Üzleti logika megvalósítása
- Külső szolgáltatók
- Adatbázisok
- System Under Test (SUT)

Konfidencia intervallum

17. Mit jelent a konfidencia intervallum fogalma?

(Kísérlettervezés diasor, 17.dia)

Ha tetszőleges eloszlású, d szórású vizsgált jellemzőről t db (>30) megfigyelést végzünk, a tapasztalati átlagról:

68% biztonsággal kijelenthető, hogy d/sqrt(t) sugarú intervallumba esik

95% biztonsággal kijelenthető, hogy 2d/sqrt(t) sugarú intervallumba esik

99.7% biztonsággal kijelenthető, hogy 3d/sqrt(t) sugarú intervallumba esik

`Wikipedia: A konfidencia-intervallum adott szignifikancia-szinten: a becsült változó alsó és felső korlátja. A konfidencia-intervallum intervallum értékű becslést ad egy paraméterre: valószínűleg ezek közé a korlátok közé esik.

Burstiness faktor

18. Mit értünk „burstiness faktor” alatt? Miért lényeges ez a jellemző?

λ_k : érkezési ráta a k. időszeletben (n darab időszelet)

λ : érkezési ráta az egész időszakra

b : burstiness factor, a zsúfolt időszeletek aránya

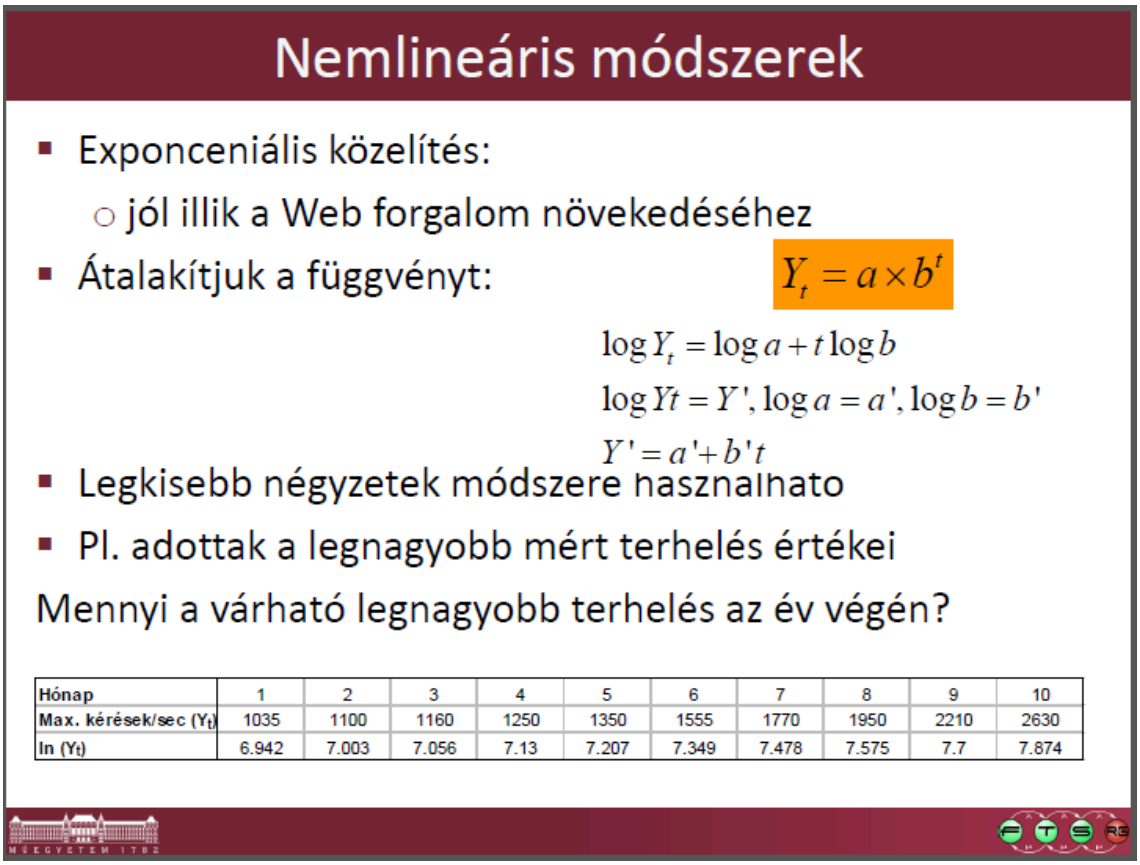
$$b = \frac{\sum_{i=1..n} \{0, \text{ ha } \lambda_i \le \lambda;\ 1, \text{ ha } \lambda_i > \lambda\}}{n}$$

Lényeges a jellemző, mivel szignifikánsan befolyásolhatja az áteresztőképességet.

Terhelés becslés

19. Milyen matematikai módszert alkalmazna egy webportál látogatottságának becslésére, ha adottak az előző hetek terhelési adatai, és tudjuk, hogy a portál felhasználóinak száma gyorsuló ütemben növekszik? Mi ennek a módszernek a lényege?

Exponenciális regresszió (ld. Terheléselőrejelzés jegyzet, 21. dia (27.dia 2014)), $Y_t = a * e^{bt}$



Hónap	1	2	3	4	5	6	7	8	9	10
Max. kérések/sec (Y_t)	1035	1100	1160	1250	1350	1555	1770	1950	2210	2630
ln (Y_t)	6.942	7.003	7.056	7.13	7.207	7.349	7.478	7.575	7.7	7.874

reliability availability

20. Mit jelent a megbízhatóság (reliability) és a rendelkezésreállás (availability)? Mi a kettő közt a különbség?

Megbízhatóság: (reliability) közben nem hibásodhat meg.

- Annak a valószínűsége, hogy amikor ránézek a rendszerre (tetszőleges időpontban), annak státusza UP úgy, hogy eddig még sosem volt DOWN.

Rendelkezésre állás: (availability) közben meghibásodhat.

- Annak a valószínűsége, hogy amikor ránézek a rendszerre (tetszőleges időpontban) annak státusza UP.

rendelkezésre állás

21. Mi a rendelkezésre állás kapcsolata a MUT, MDT metrikákkal?

(Hibamodellezés diasor, 12. dia)

MUT (Mean Up Time): hibamentes működési idő

- A várható értéke annak (időben), hogy a rendszer státusza UP. (vagyis várhatóan milyen hosszú ideig lesz UP)

MDT (Mean Down Time): hibás állapot ideje

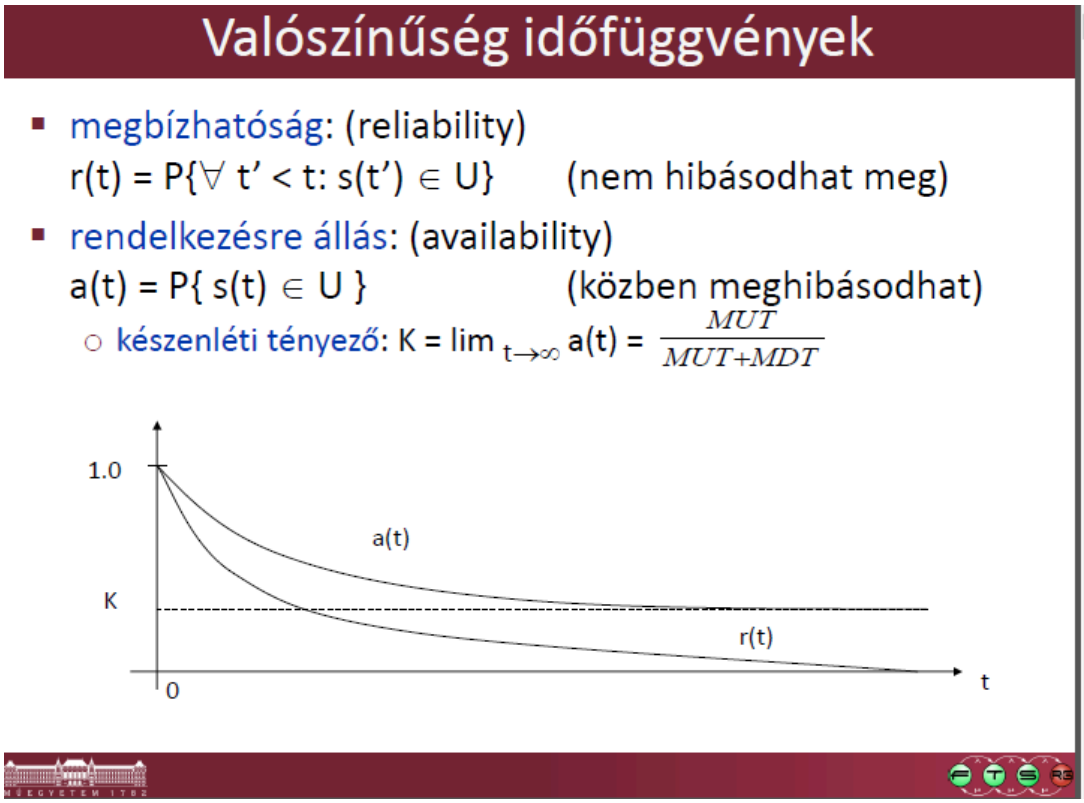
- A várható értéke annak (időben), hogy a rendszer státusza DOWN. (vagyis várhatóan milyen hosszú ideig lesz DOWN)

Hibák közötti idő: MTBF = MUT + MDT

- Ha bekövetkezik a hiba (DOWN-ra vált a rendszer), akkor várhatóan nem lesz újabb hiba a hibát követő DOWN szakaszig (MDT) plusz az ezután következő UP szakasz végéig (MUT).

Rendelkezésre állás esetén a készenléti tényező:

$K = \lim_{t \to \infty} a(t) = (MUT)/(MUT+MDT)$



BPEL

22. Mi a különbség BPEL folyamatokban a „fault handler" és "compensation handler" között?

Fault handlers define how the BPEL process service component responds when the web services return data other than what is normally expected.

A Compensation Handler is a container for the activities that perform compensation actions.

Régi wikiről:

Fault handling:

- Hiba típusokra, változókra definiálható
- Akkor hívódik meg, ha a fellépő hiba illeszkedik
- Mint Java-ban az exception handling
- Alulról felfele hívódik meg (scope-hierarchia)

Compensation handling:

- Tranzakció meghiúsul?
- Erőforrás zárolás?
- Kompenzáció
- Rekurzívan, fordított sorrendben
- Változó értékek megtartása

esemény szintű monitoring

23. Milyen főbb céljai vannak az üzleti esemény szintű monitorozásnak? Adjon példát rá, milyen eseményeket lenne érdemes monitorozni, ha a ZH javítás folyamatát szeretnénk ellenőrizni (feltéve, hogy a zh megírása és javítása is elektronikusan történik).

Futásidőbeli információk gyűjtése, az előírások teljesítésének ellenőrzése, a teljesítmény valós környezetben vizsgálható, biztonsági ellenőrzés.Az üzleti tevékenység monitorozásának célja valós idejű információk nyújtása az aktuális státuszokról és a különböző műveletek, folyamatok és tranzakciók eredményeiről. A legfőbb előnye, hogy egy vállalat jobban megalapozott üzleti döntéseket hozhat, gyorsítja a problémás területek megtalálását, és új helyzetekben teljes mértékben kihasználják a kínálkozó lehetőségeket.

Konfigurációmenedzsment

24. Mik a konfigurációmenedzsment főbb feladatai? Milyen szabványt használhat adatreprezentációra?

A konfigurációmenedzsment adatokat szolgáltat az aktuális konfigurációs részletekről, illetve nyilvántartja a konfiguráció változásait. Kezeli a hálózati- és rendszereszközök konfigurációit.

A készletnyilvántartás és topológia szolgáltatás, melynek feladata a hardver- és szoftverkészlet kezelése, és a rendszer konfigurációs térképének karbantartása. A megfelelő készletnyilvántartás érdekében célszerű az adatokat menedzsment információs bázis (MIB) segítségével tárolni.

A pontos készletnyilvántartáshoz szükséges a változások illetve változtatások megfelelő nyilvántartása. Továbbá a használhatóság érdekében szükséges a megfelelő elnevezés és címnk használata, azaz a névtér menedzsment.

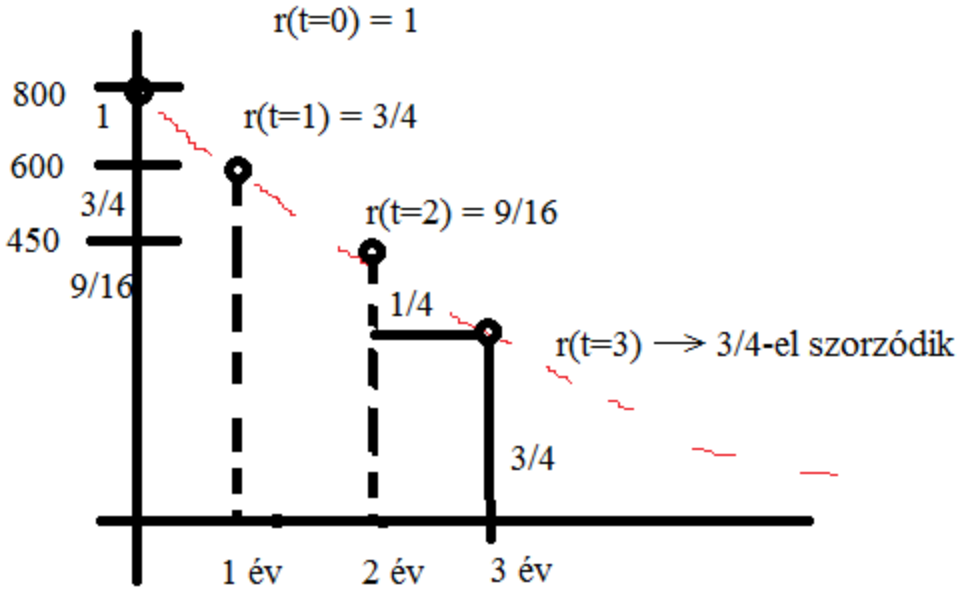
Kábelezési rendszer nyilvántartása és menedzselése. A funkció megvalósítására úgynevezett CMS (cable management system) termékek használhatók.

ZH GYAK Rendszermodellezés gyakorlópéldák 2013:

1. Kvantitatív hibamodellezés

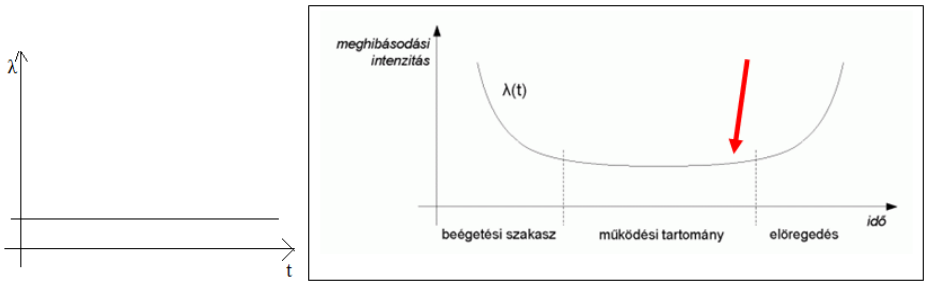
A szervereinkhez két évvel ezelőtt 800 darab új, egyforma merevlemezt szereztünk be egyazon gyártó egyazon sorozatából. Egy évvel ezelőtt már csak 600 működött az eredeti készletből (a többit újakkal pótoltuk, de ezt most nem vizsgáljuk), mostanra pedig újabb 150 eszköz mondta fel a szolgálatot.
A lemezek a vizsgált időszak alatt intenzíven, de egyenletesen és egyformán voltak terhelve. A gyártási hibás darabokat előzetes teszteléssel kiszűrtük, az eszközök elöregedése pedig két-három év alatt még nem következik be.

a. Mit mondhatunk a merevlemeztípus megbízhatósági függvényéről?



exponenciális

b. A második bekezdés alapján mi következik a merevlemez erőforrástípus viselkedésére? Hogyan támasztják ezt alá a mért adatok?



Alapesetben a fenti ábra a “kád görbét” mutatná. A “kád” baloldali lefelé futó éle a hibás darabok előzetes kiszűrése, míg a jobb oldali felfutó éle pedig azért tűnik el mert “2-3 év alatt még nem következik be az eszközök elöregedése”. (jobb oldalt az eredeti kádgörbe, annak a középső része kell)

Továbbá a “t” tengellyel párhuzamos egyenes az $r(t) = e^{-\lambda t}$, ami egy örökifjú tulajdonsággal bír, ezért is lesz párhuzamos.

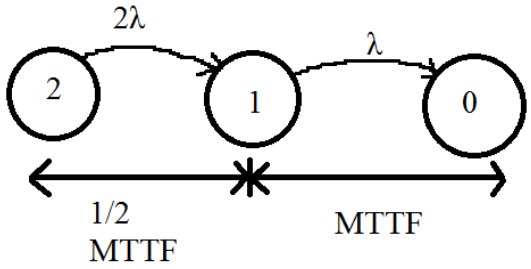
c. Milyen becslés adható arra, hogy a következő évben mennyi lemezt kell az eredeti készletből pótolni?
rajzon

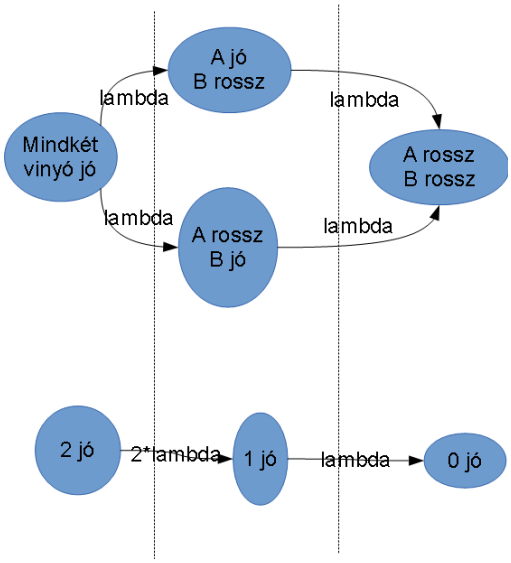
A többek által helyesnek ítélt megoldás:
Elvileg: Ha mindenképpen ragaszkodunk az exponenciális alakhoz, akkor a hibamodellezés dia 14. oldalán található
r(t)=e^(-lambda*t) képlettel modellezve, a lambda=ln(4/3) értékkel számolva kijön, hogy első év végére 3/4, második végére 9/16-od része, harmadik végére 27/64-ed része fog működni…
Tehát: 337db működik és 113db-ot kell pótolni a második évi működő darabszám tartásához.
(ami tulajdonképpen (t=0):1, (t=1): (¾)^1= ¾ , (t=2): (¾)^2= 9/16 , (t=3): (¾)^3=27/64 stb..)

d. Az adott típusból egyetlen merevlemez a használatbavételétől számítva várhatóan mennyi ideig működőképes? A számítás menete az érdekes, nem kell számszerű eredményt adni.
MTTF: 1/ **λ**
r(t=1év)=e^(- λ*1év) = 3/4

λ*1 év = -ln 3/4év
MTTF = 1/(ln4/3) év = 3.48 év.

e. Ha az egyik szerverben ezen lemezek közül 2 üzemel meleg tartalékban, várhatóan mennyi idő alatt fog mindkettő meghibásodni?





3/2 MTTF

f. Ha a meghibásodás után két napot vesz igénybe a csere, mekkora egy merevlemezhely készenléti tényezője?
k: készenléti tényező jele, ami annak a valószínűségét jelenti, hogy amikor ránézünk a rendszerre, működőképes-e.
k = MTTF / (MTTF + MTTR)
k= MTTF/(MTTF + 2nap)

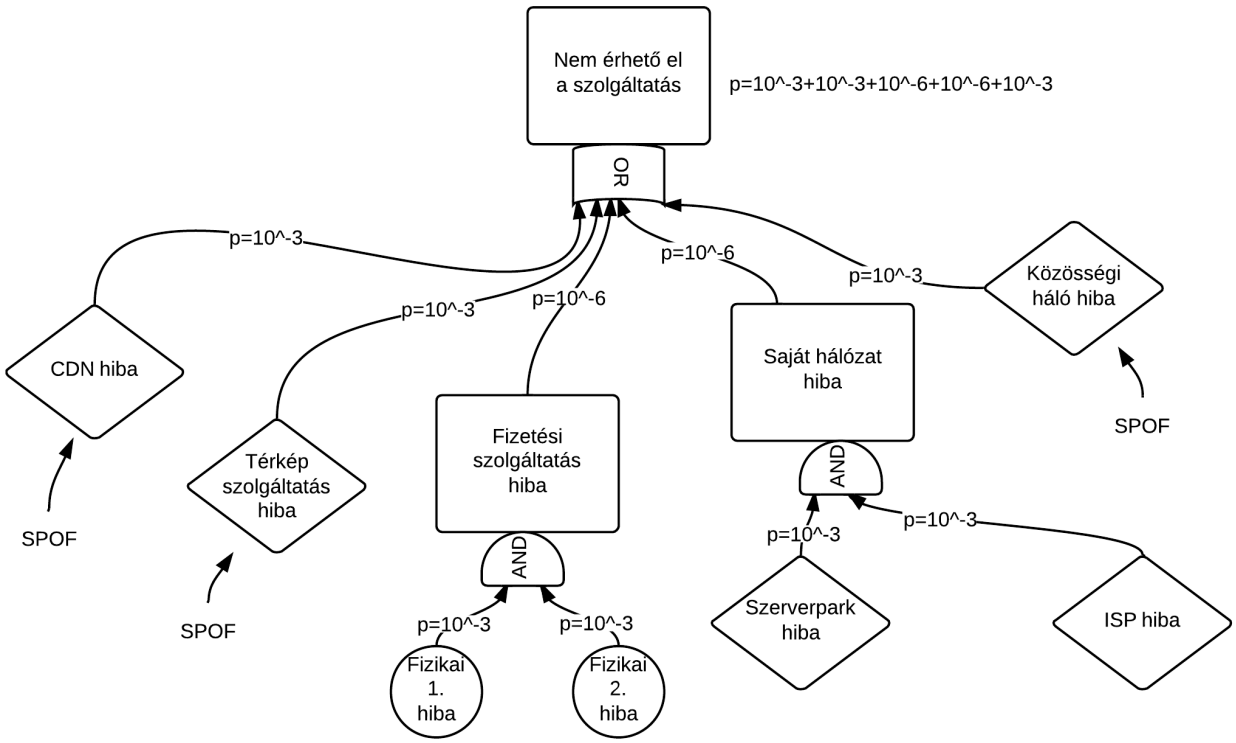
(MTTF: mean time to failure, vagyis mennyi időnként lesz hibás a rendszer átlagosan
MTTR: mean time to recovery- MTTF+javítás ideje, vagyis mennyi idő alatt romlik el és javítják ki a rendszert)

2. Kvalitatív hibamodellezés hibafával

Szolgáltatásunk működéséhez szükséges, hogy a látogatók a weboldalon kínált tartalmakat elérhessék a szerződött tartalomszolgáltatói hálózat (CDN) jóvoltából, vagy a saját infrastruktúránkból és az internetszolgáltató (ISP) is ; utóbbi akkor lehetséges, ha a szerverparkunk üzemképes állapotban van. További feltétel a külső térképszolgáltató és a közösségi hálózat működése. Végül fennakadást jelent az is, ha mindkét fizetési szolgáltatás szünetel.

a. Rajzolja le a „nem érhető el a szolgáltatás" rendszerszintű hibajelenség hibafáját!

HA MEG TUDOD MUTATNI MIÉRT NEM JÓ AZ ALÁBBI ÁBRA, AKKOR LE TUDOD RAJZOLNI A JÓT! (segítség az ábra alatt)



Itt szerintem ügyesen lehet De Morgannal játszani, vagyis:
CDN VAGY saját hálózat = tartalomelérés <-> CDN hiba ÉS saját hálózat hiba = tartalomelérés hiba
tehát téglalapba: tartalomelérés hiba, bele ÉS-sel két rombusz: CDN hiba és téglalappal Saját hálózat hiba, mert ez is függ két dologtól:
saját hálózat elérhető = szerverpark ÉS ISP <-> saját hálózat hiba = szerverpark hiba VAGY ISP hiba
ezutóbbi alapján a saját hálózat hiba téglalapba VAGY-gyal: szerverpark hiba vagy ISP hiba rombuszok
külön OR-ral megy a nagy buciba a közösségi háló hiba, illetve fizetés hiba téglalap
az utóbbi ellenkezőjét kapjuk megint a szövegből: fizetés = egyik mód VAGY másik mód, ami nekünk használható módon fizetés hiba = egyik mód hiba ÉS másik mód hiba TEHÁT a bal oldali két rombusztól és jobbról a második bucitól eltekintve az ábra egészen fasza.

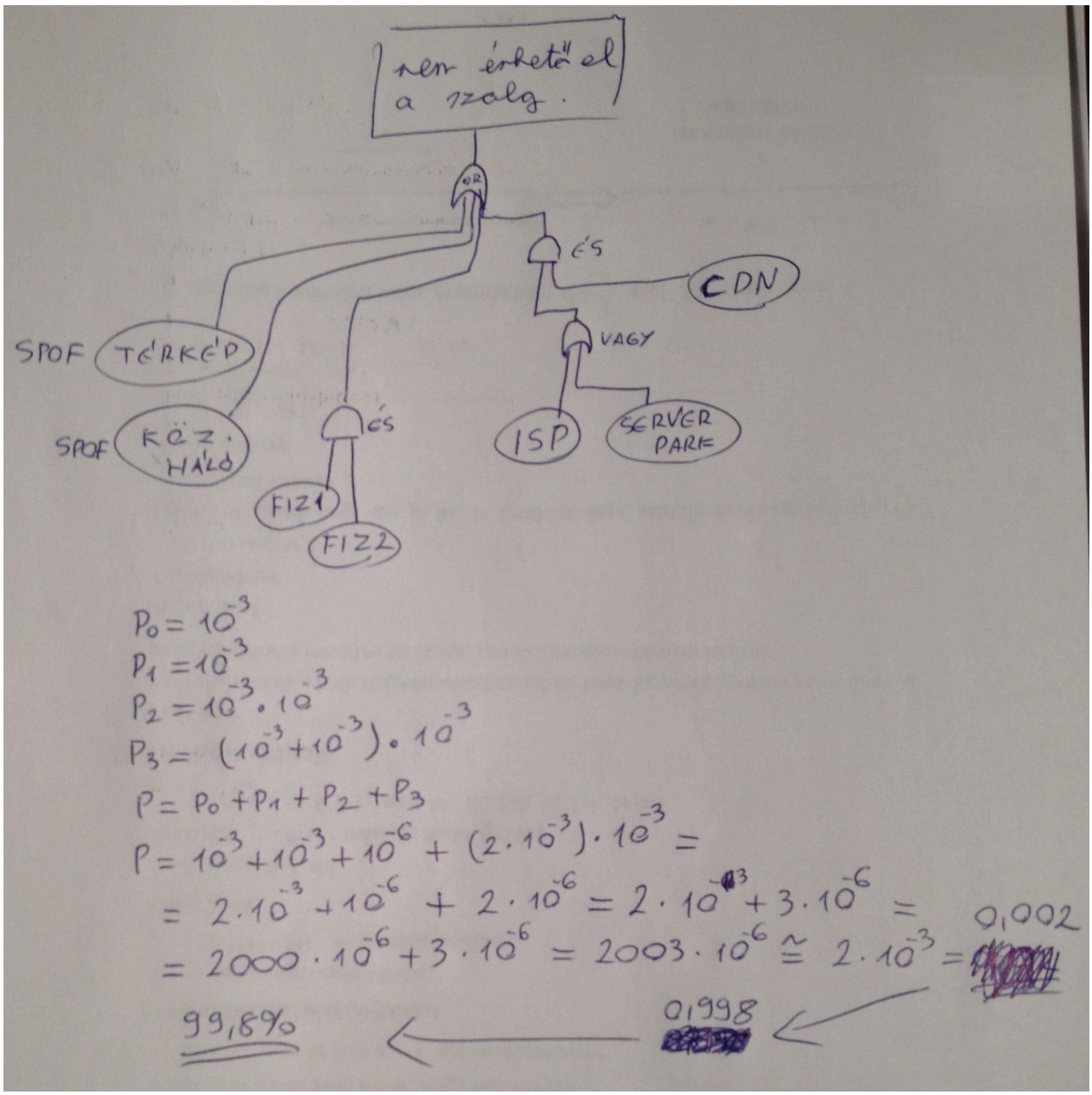
b. A hibafa redukció módszerrel azonosítsa a rendszer egyszeres hibapontjait és kritikus eseményeit!
SPOF: http://en.wikipedia.org/wiki/Single_point_of_failure
SPOF - ami már önmagában is hibát okoz
kritikus esemény - több vágatban is szerepel (vágat - AND kapuval összefogott elsődleges események)

ez fent leírtak alapján kritikus esemény nincs, és egyetlen SPOF a közösségi háló, mert a többi esetben ha egy rombusz kiesik még ott a többi

c. Az összes elemi meghibásodás tetszőleges időpillanatban p=10^-3 valószínűséggel áll fent. Mi következik a rendszerszintű rendelkezésre állásra? Milyen feltételezéseket és közelítéseket kellett tenni a számításhoz?
P(A ÉS B) = P(A)*P(B) (akkor ha p-k függetlenek)

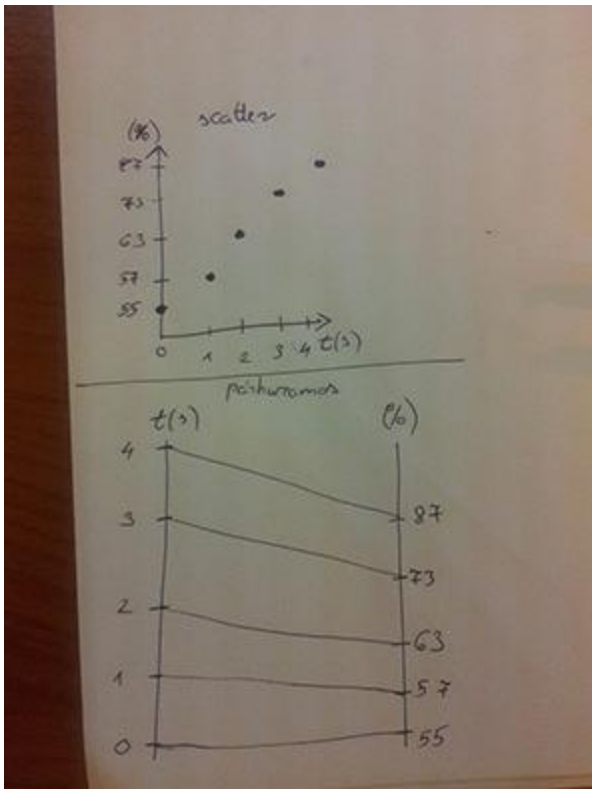
P(A VAGY B) = P(A) + P(B) - P(AB) <= P(A)+P(B) (p-k kicsik, felső becslés is elég)

Egy lehetséges jó megoldás (ábra + számolás)



3. Adatelemzés

Egy szerveren az alábbi kihasználtsági értékeket mértük a t={0, 1, 2, 3, 4} [s] időpontokban:
55%, 57%, 63%, 73%, 87%.
a. Ábrázoljuk a két adatsort pontfelhő (scatterplot) diagramon, valamint a párhuzamos koordináták módszerével!



b. Lineáris regressziót alkalmazva szeretnénk a következő értékeket megjósolni. Mi a lineáris regresszió vezérlő metrikája? Avagy: milyen célfüggvény alapján határozzuk meg a lineáris regresszió paramétereit?
Megoldás: Vezérlő metrika: négyzetes hibaösszeg minimalizálása

c. Az alábbiak közül melyik görbe lehet esetünkben a kihasználtság százalékértékének (0-100) jó lineáris regressziós becslése a t időparaméter alapján?
A. 2t^2+55 (nem lehet)
B. (t/4)^2/87 + (1 - t/4)/55 (nem lehet)
C. 8t+51 (lehet)
D. 8t+45 (lehet)
E. 55 + 2*t + 8*t^2/2+18*t^3/3+32*t^4/4 (nem lehet)

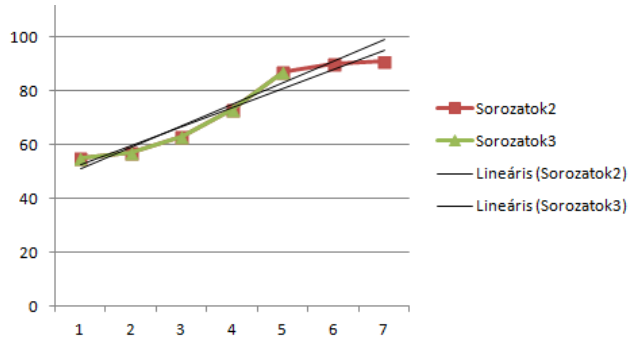
Megj: A “nem lehet” sorok, azért kizártak mert biztosan nem egy lineáris egyenest adnak vissza.

t	**x**	**C (8t+51)**	**Ce (C-X)^2**	**D (8t+45)**	**De (D-X)^2**
0	55	51	16	45	100
1	57	59	4	53	16
2	63	67	16	61	4
3	73	75	4	69	16
4	87	83	16	77	100
			$\sum = 56$		$\sum = 236$

d. A t={5, 6} időpontokban a következő kihasználtságot mérjük: 90%, 91%. Milyen rendszerszintű jelenség okozhatja ezt nyílt, illetve zárt rendszerek esetében?
Nyílt rendszer: telítődik a rendszer, túl sok kérés érkezik be
Zárt rendszer: kérések száma korlátozva van, így nem nő tovább a kihasználtság

Elméleti háttér:
Zárt rendszer modell: A modellbe beleveszem a külvilágot (a felhasználót) is. Megkötéseket teszek a viselkedésére. Ilyenkor a külvilág viselkedését megszorítom. Az ésszerű magatartásra biztos jól fog reagálni.
Nyílt rendszer modell: Nem teszek megkötéseket a külvilágra. Ilyenkor a funkcionalitást és a biztonságot is biztosítani kell. Így sokkal egyszerűbb felkészülni. Ha „tisztességesen” járunk el azt is meg kell nézni, hogy mi van ha nem jó a bemenet.

e. Változna-e (és hogyan) a becslőfüggvény, ha t={5;6} időpontokat is figyelembe vennénk? A lineáris regresszión alapuló becslés (és általában a regressziós módszerek) milyen veszélyeire mutat ez rá?
Változna: laposabb lenne
Veszély: kevés adat alapján, illetve nem lineárisan változó adatokat rosszul becsli.
(Érzékeny:
nagyon kilógó adatokra
multikollinearitásra (két független változó erősen korrelál, közel állnak a linearitáshoz)
rosszul kondicionáltá válik, érzékeny lesz a mérési hibákra.)



4. Kísérlettervezés

Egy hardvereszköz egy paraméterének meghatározására 900 mérést végeztünk el, melyek során mérési hibákból adódóan 1.5%-os szórást tapasztaltunk. A kísérlet eredménye szerint a vizsgált érték 1000 egység; milyen konfidenciaszint mellett tudunk legfeljebb ±0.1% tévedést garantálni?

d=1000 * 0,015 = 15

t=900

konfidencia=d/sqrt(t)=0,5 ← szerintem ez NEM konfidencia, hanem szigma!!!

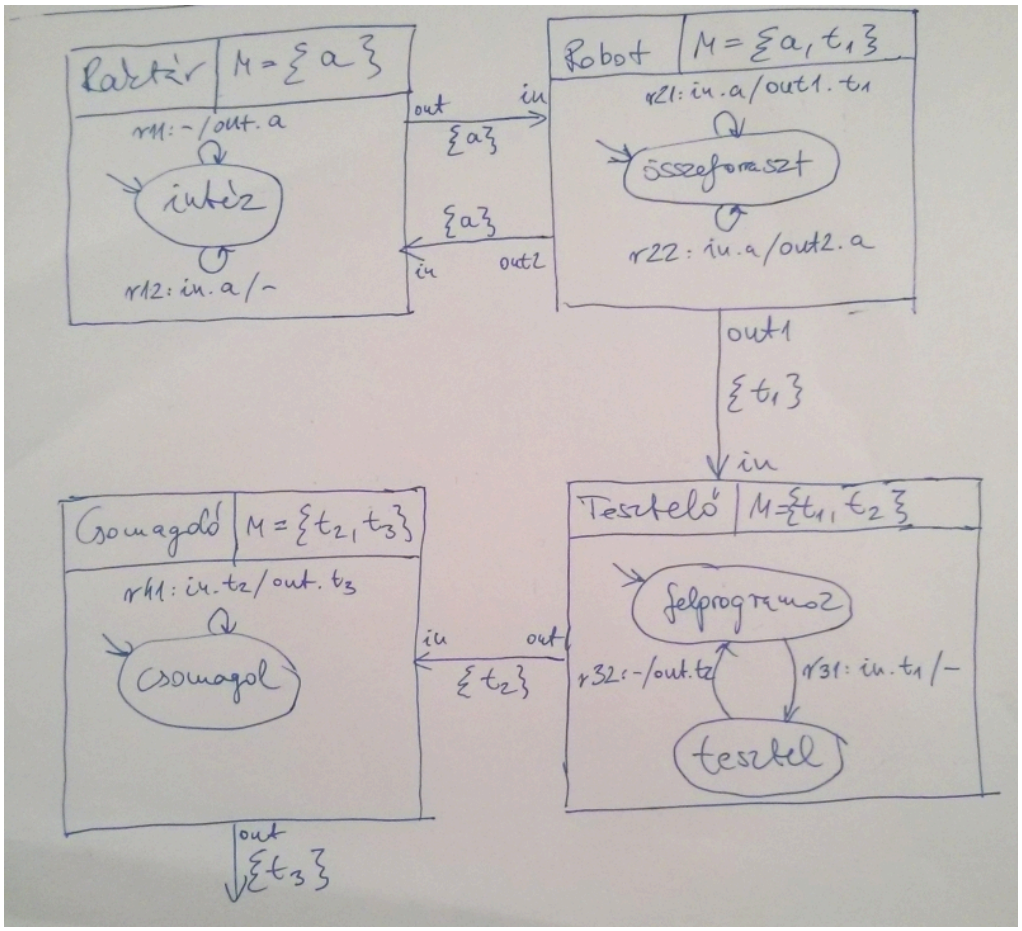
1000 * 0,001 = ±1 tévedés

95% mellett ±2 szórásnyi intervallum, ami pont ±1.

5. Viselkedésmodellezés állapotmodellel, adatfolyamhálóval és munkafolyamattal

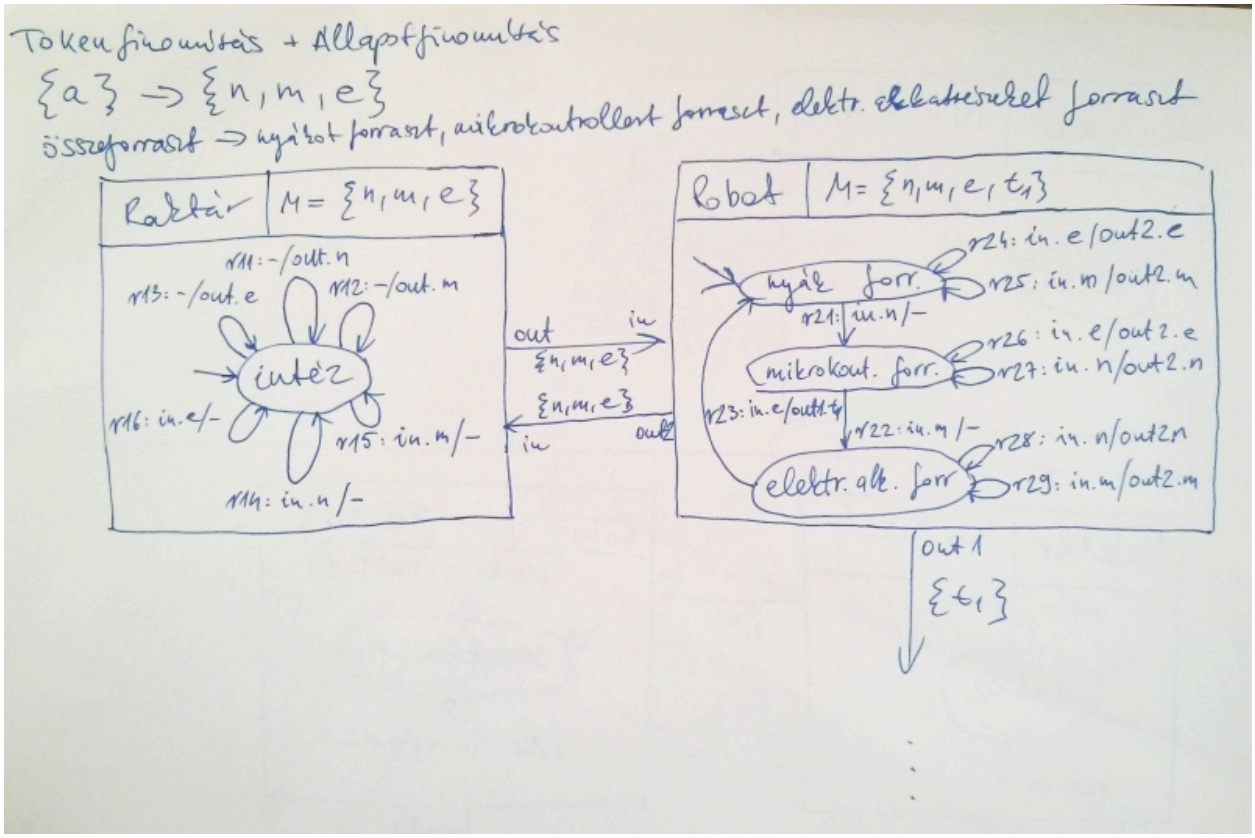
Az Arduino cég mikrovezérlő-panelek gyártásával foglalkozik. Az eszközhöz több alkatrész szükséges, ezek egyenként a raktárból az összeállító szalagra kerülnek. Onnan egy robot veszi fel őket összeforrasztás céljára; időnként visszaküld a raktárba egy-egy alkatrészt, de időről időre egy kész terméket juttat el a tesztelő részlegre. Itt egy tesztelő felprogramozza és teszteli a kész eszközöket, végül átadja azokat a csomagoló részlegnek, ahol becsomagolják a mikrovezérlőket.

a. Modellezze adatfolyamhálóval az előállítás menetét a szerelőszalagtól a csomagolt termék elkészültéig!

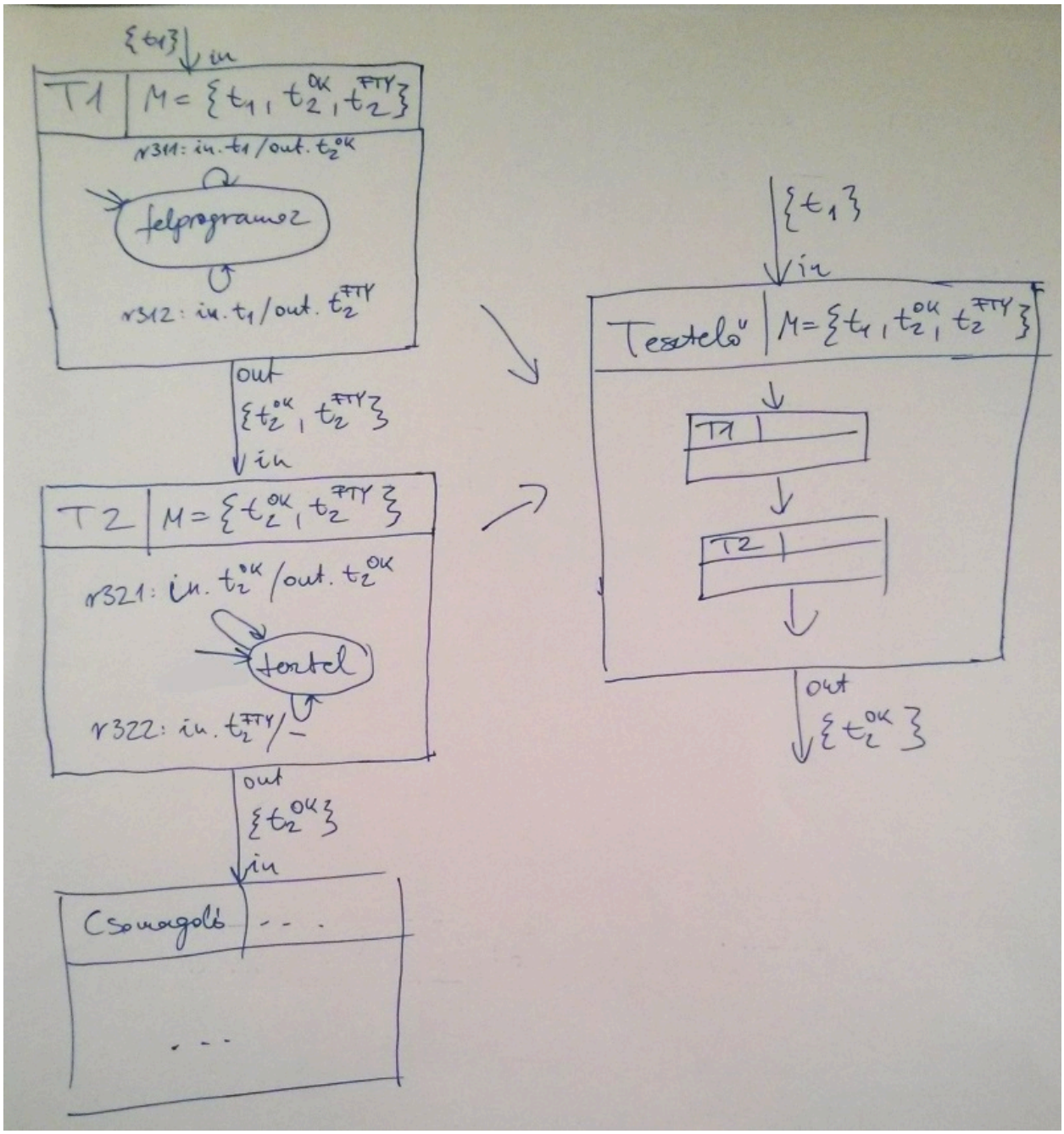


ez nem a hivatalos megoldás!

b. Finomítsa a modellt a következőképpen: a kész panel előállításához a szerelőrobotnak a következő alkatrészekre van szüksége: egy nyákra, egy AVR mikrokontrollerre, valamint egy készlet ellenállásra és kondenzátorra (az egyszerűség kedvéért az ellenállásokat és kondenzátorokat egy közös csomag tartalmazza). Az alkatrészeket egyenként, ebben a sorrendben forrasztja be a nyákba. Ha nem a megfelelő alkatrész kerül a „kezébe", akkor azt visszaküldi a raktárba.



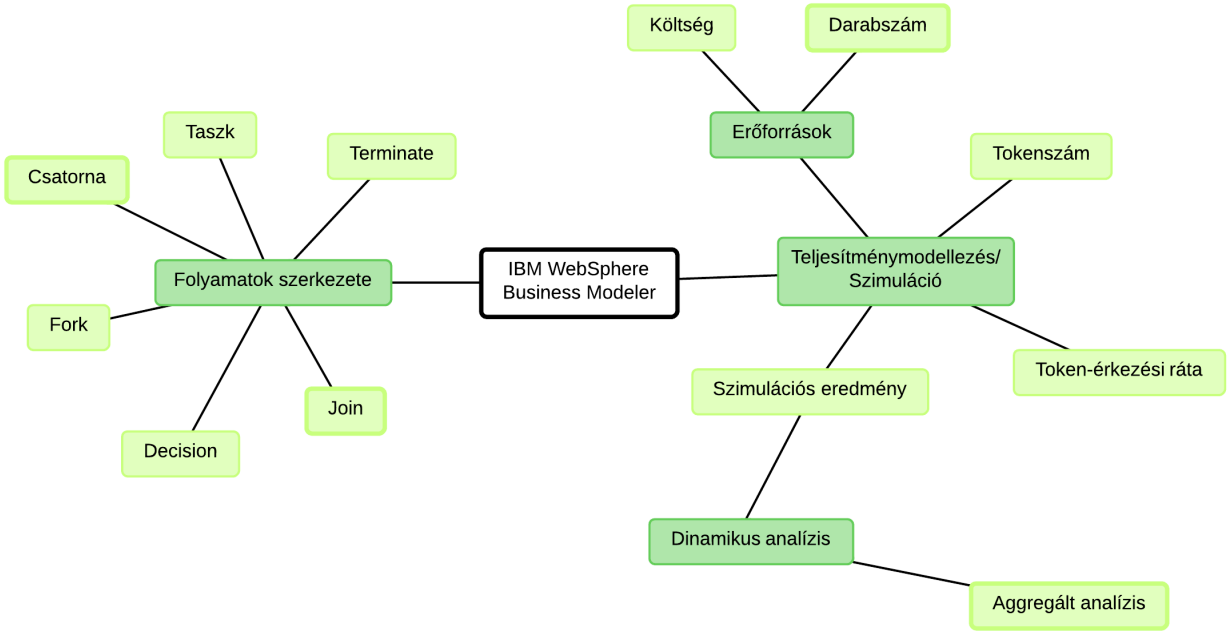
c. Finomítsa a modellt a következőképpen: a tesztelés során a tesztelő a kész panelt felprogramozza és teszteli. A teszt eredményétől függően a hibátlan terméket továbbküldi a csomagoló részlegre, a hibásat viszont eldobja.



d. Ábrázolja a háromféle alkatrész raktárból kiemelését, az összeépítés lépéseit, a tesztelést és csomagolást BPMN munkafolyamat diagramon! Ebben a változatban a mikrokontroller és az elektronikai alkatrészcsomag tetszőleges sorrendű (akár egyszerre történő) beszerelése legyen megengedett!

6. Modellezés ontológiával

Készítsen ontológiát, mely alkalmas a házi feladatban használt modellező program (IBM WebSphere Business Modeler) lényegesebb modellezési elemeinek leírására. Elvárás, hogy a modell elemeinek segítségével le lehessen írni a folyamatok szerkezetét és a teljesítménymodellezéshez/szimulációhoz szükséges beállításokat. Nem szükséges a hierarchikus modellezés (alfolyamatok) támogatása. Adjon példát olyan kényszerek leírására, melyek i) a folyamat felépítésének helyességét, ii) a szimulációs beállítások teljességét vizsgálják.



7. Teljesítménymodellezés

Csúcsidőben mérve azt tapasztaltuk, hogy 1000 másodperc alatt 60 000 lekérdezést szolgált ki az infrastruktúránk, további 5000 beérkező kérést sajnos túlterhelés miatt visszautasított. Egy kérés kiszolgáláshoz két lényeges erőforrásfajta szükséges; a 3 db. adatbázisszerver végig telítésben volt, míg az 5 db. webszerveren egyenként 660 másodperc foglaltsági időt regisztráltunk.
Feltételezhetjük, hogy az intelligens terheléselosztó révén az adott típusú szerverek terhelése egyenletes, a visszautasított kérések a vizsgált erőforrásokat már nem terhelték, és a skálázódás lineáris.

a. Mekkora a kérések érkezési rátája, a sikeresen kiszolgált kérések átbocsátási rátája, és a rendszer átbocsátóképessége? Mekkora egy kérés szolgáltatásigénye a webszerveren és az adatbázisszerveren? (2p)
érkezési ráta (λ_i): 65000[kérés]/1000[sec]=65[kérés]/[sec]
átbocsátási ráta (X): 60000[kérés]/1000[sec]=60[kérés]/[sec]
átbocsátóképesség (X_{max}): U = X/X_{max}, ahol U=100% (mivel telítésben van a rendszer), így: X = X_{max}, azaz az átbocsátóképesség is 60[kérés]/[sec].

Kérés szolgáltatásigénye a webszerveren és DB szerveren:
$D_i=n_i$ x U_i / X_0 képlet alapján számolhatunk.
Webszerver: D_{WS} = 5 x 0,66 / 60
Adatbázisszerver: D_{DB} = 3 x 1 / 60

b. Mekkora a kétféle erőforrás kihasználtsága?
Udbi=100% -> 60/3 => átbocsátás=20[kérés]/[sec] egyenként
Uwsi=66% -> 60/5 => átbocsátás=12[kérés]/[sec] egyenként => Uwsmax=18,18[kérés]/[sec] egyenként

c. Ha további két tartalék adatbázisszervert üzembe helyezünk, kiszolgálható-e már az összes kérés, hogyan változnak a kihasználtságok, melyik erőforrás lesz a szűk keresztmetszet, és mennyi lesz az átbocsátóképesség?
Plusz két adatbázis-szerver: == 5 darab, egyenként 20[kérés]/[sec] => 100[kérés]/[sec] => 100000[kérés]/1000[sec]

kihasználtság:
65[kérés]/[sec] => 65/5 => 13[kérés]/[sec] egy adatbázis-szerverre
kihasználtság: 13/20=65% egyenként

65[kérés]/[sec] => 65/5 => 13[kérés]/[sec] egy webszerverre
kihasználtság: 13/18,18=71,5% egyenként

szűk keresztmetszet: webszerver
átbocsátás: 65[kérés]/[sec]
áteresztő képesség (maximum): 18,18*5=90,9 [kérés]/[sec]

2. gyak állapot alapú modellezés

1. Háromrétegű kiszolgáló infrastruktúránk viselkedésének modellezésére megfelelő

állapotpartíciók-e a következők:

a. {Webszerver, Alkalmazásszerver, Adatbázisszerver}
b. {Webszerver dolgozik, Alkalmazásszerver dolgozik, Adatbázisszerver dolgozik}
c. {Leállítva, Tétlen üzemel, Aktívan dolgozik}
d. ℵ(mint a pillanatnyilag feldolgozás alatt álló kérések száma)
e. {A kérés feldolgozása még nem kezdődött el, a szerverek épp dolgoznak a kéréssel, a kérés kiszolgálása befejeződött}
f. {igaz}

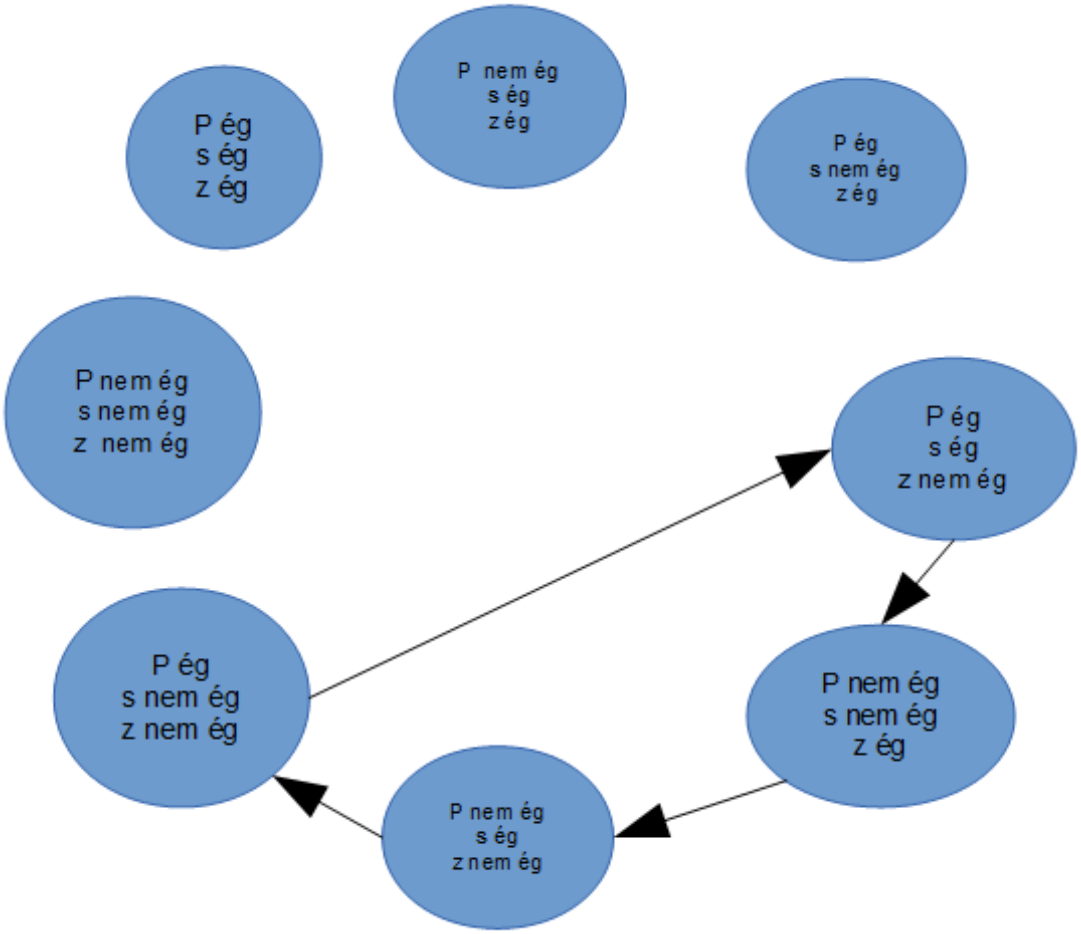
Megoldás:

Akkor megfelelő, ha teljes és nincs átfedés.
(Az állapottér a rendszer valamennyi lehetséges állapotát tartalmazó halmaz. Egyszerre csak pontosan egy állapota lehet és kell, hogy legyen a rendszernek.)

a. megfelelő-e: Nem. ok: ezek erőforrások, nem állapotok.
b. megfelelő-e: Nem. ok: Lehetséges, hogy pl. mindhárom egyszerre, vagy egyik sem dolgozik
c. megfelelő-e: Igen. ok: Teljes, nincs átfedés (persze, kivéve ha azt mondjuk, hogy van pl. hibernált állapot is, mert akkor nem lesz megfelelő). Tehát megfelelőségre jó megoldás lehet egyes feladatoknál “igen” és “nem” is, lényeg hogy megindokoljuk.
d. megfelelő-e: Igen. ok: Teljes, nincs átfedés
e. i.) megfelelő-e: Igen. ok: Ha egy kérés ciklusát nézzük, megfelelő
ii.) megfelelő-e: Nem. ok: Ha több kérését nézzük, nem megfelelő
Mindkét megoldás jó, a legjobb ha mindkettőt le is írjuk
f. megfelelő-e: Igen. ok: Teljes állapotpartíció, átfedés nyilvánvalóan nincsen.

2. Közlekedési lámpát vezérlő elektronikát tervezünk.

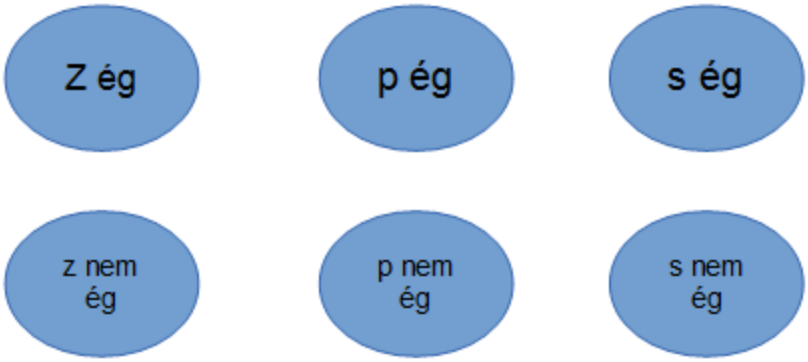
a. Készítsd el egy egyszerű háromfényű piros-sárga-zöld közlekedési lámpa olyan állapotpartícióját, amely kellően finom ahhoz, hogy a lámpák vezérlését ez alapján lehessen végezni! Győződj meg arról, hogy az állapotpartíció kizárólagos és teljes!



Az állapotpartíció (=állapottér) akkor teljes, ha a rendszer valamennyi lehetséges állapotát tartalmazza és igaz rá, hogy minden időben pontosan egy állapot igaz a rendszerre.
A megoldás teljesíti ezt a követelményt.
A megoldás elég részletes ahhoz, hogy a lámpát ez alapján lehessen vezérelni.
A nyilazás a megoldásnak nem része, hisz a feladat csak az állapottér állapotait kérte.

b. A három égőnek külön-külön mi az állapottere? Milyen absztrakciós viszony áll fent a lámpa és az egyes égők állapottere közt? Hogy viszonyul a lámpa állapottere a három állapotváltozó direkt szorzatához?

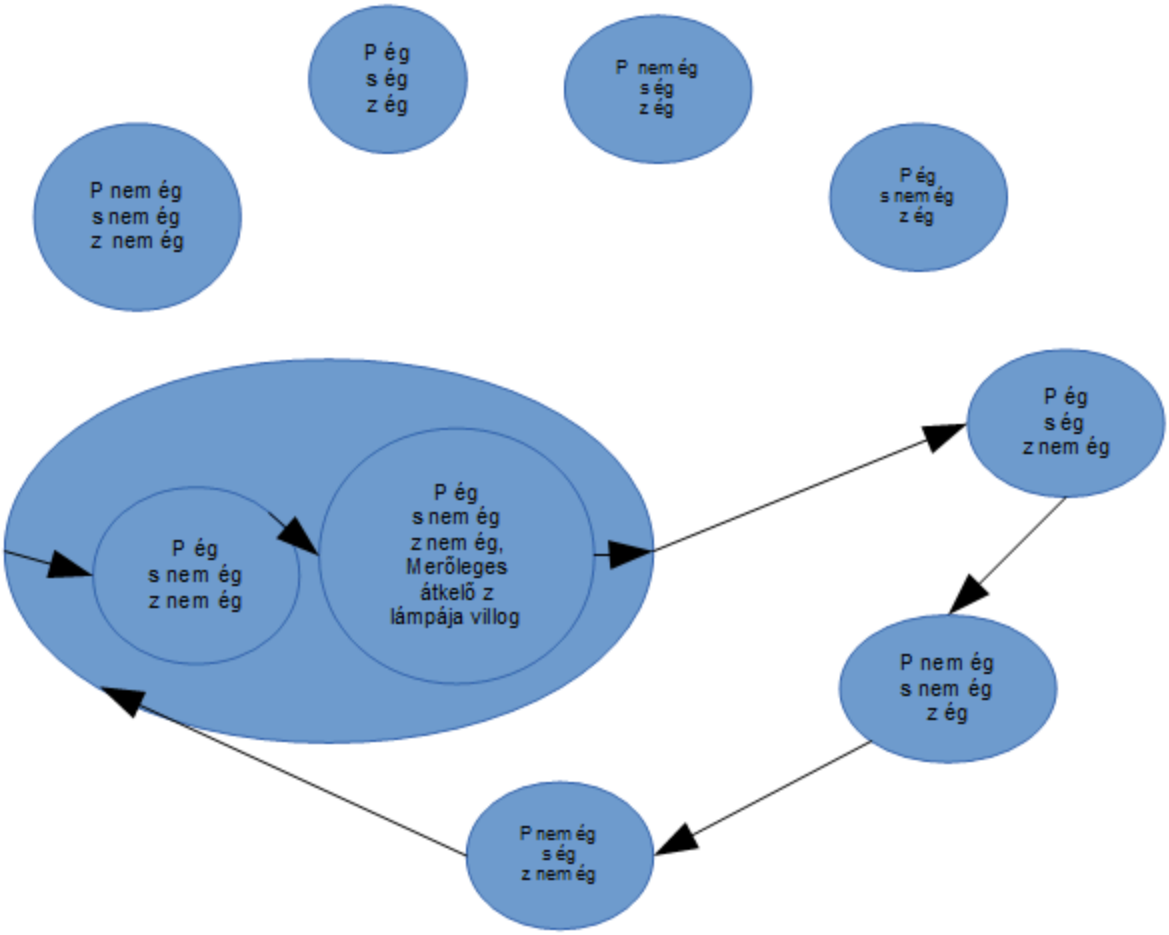
A három égőnek külön-külön mi az állapottere?



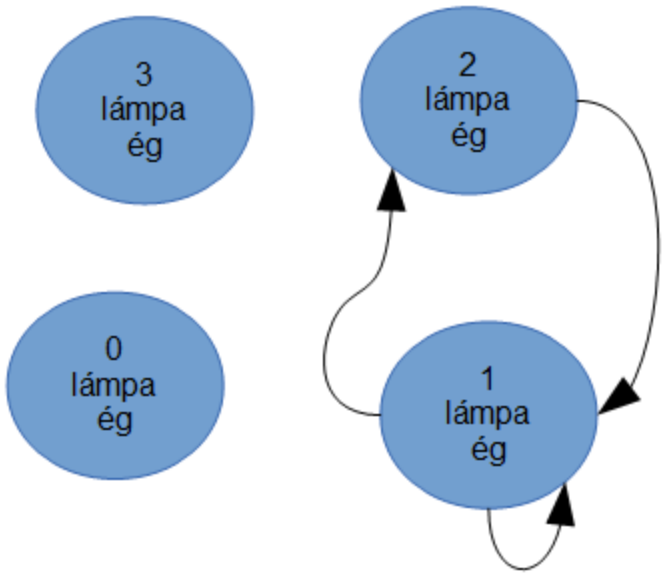
Milyen absztrakciós viszony áll fent a lámpa és az egyes égők állapottere közt?
A lámpa állapottere az egyes égők állapotterének szinkron szorzata.
Azért szinkron szorzat, mert az állapotváltozók egyszerre változhatnak.
Ha egyszerre csak egy állapotváltozó változhatna, akkor aszinkron szorzat volna.
Állapot alapú modellezés - előadás / 47-es dia
Hogy viszonyul a lámpa állapottere a három állapotváltozó direkt szorzatához?
A három állapotváltozó direkt szorzata, azaz minden kombinációban képzett hármasai kiadják a lámpa állapotterét.

c. Mik az érvényes állapotátmeneti szabályok? Készítsd el az állapotgráfot!
a) feladat megoldásában

d. A piros jelzés végén van egy olyan időszak, amikor a merőleges gyalogosátkelő zöld lámpája már villog. Finomítsuk úgy az állapotgráfot, hogy ez az állapot elkülöníthető legyen!



e. Amikor a lámpa elektromos fogyasztását vizsgáljuk, csak az érdekel, hogy a három égőből hány ég egyszerre. Absztraháld az állapotgépet úgy, hogy az állapotokat csak a fogyasztásuk különböztesse meg!



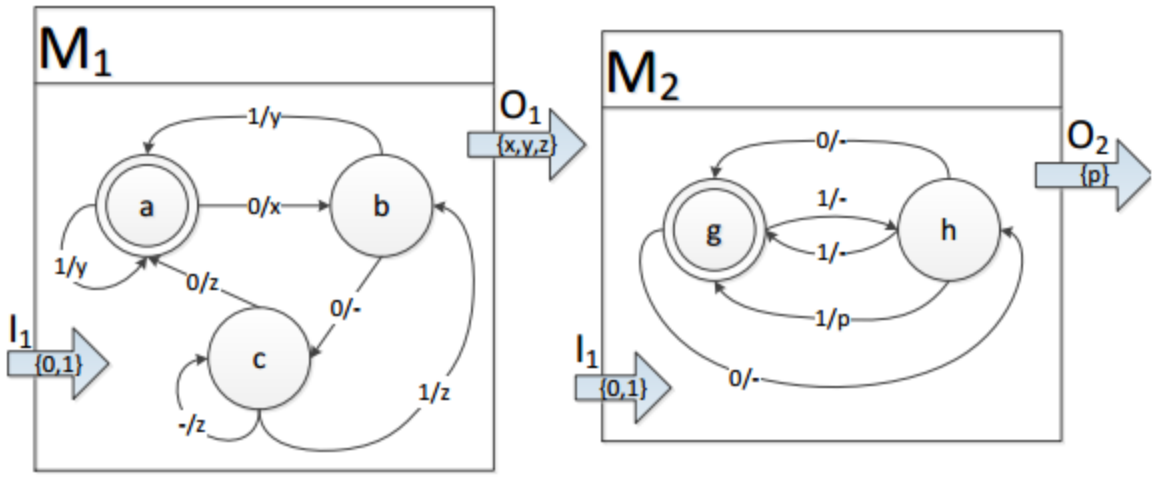
A párhuzamos átkelő lámpáját nem vettem figyelembe.

3. Modellezzük állapotgéppel egy mobiltelefon érintőképernyőjére tervezett virtuális

billentyűzetet! A billentyűzeten egyszerre vagy a kisbetűk, vagy a nagybetűk, vagy a számok és fontosabb szimbólumok, vagy ritkább szimbólumok láthatóak. Az elsődleges üzemmódváltó gomb a betűk és a számok/szimbólumok beírása között vált, a másodlagos üzemmódváltó pedig ezen kategóriákon belül. Létezik továbbá egy olyan nagybetűs állapot is, amely egy betű leütése után automatikusan kisbetűsre vált. Vegyük figyelembe a bal felső gombot (q/Q/1/=) ill. a két üzemmódváltó gombot mint inputot, és a szövegmezőbe begépelt karaktereket mint outputot!

4. Egy összetett állapotgépnek 10 állapotváltozója van, egyenként 4 állapottal. Legfeljebb mekkora a teljes állapottér?

Ez egy kombinatorika feladat: 10 helyiértékre kerülhet 4 féle számjegy. Hány különböző számot tudunk generálni így:
4^10



5. Tekintsük az M1 szintetikus állapotgépet!

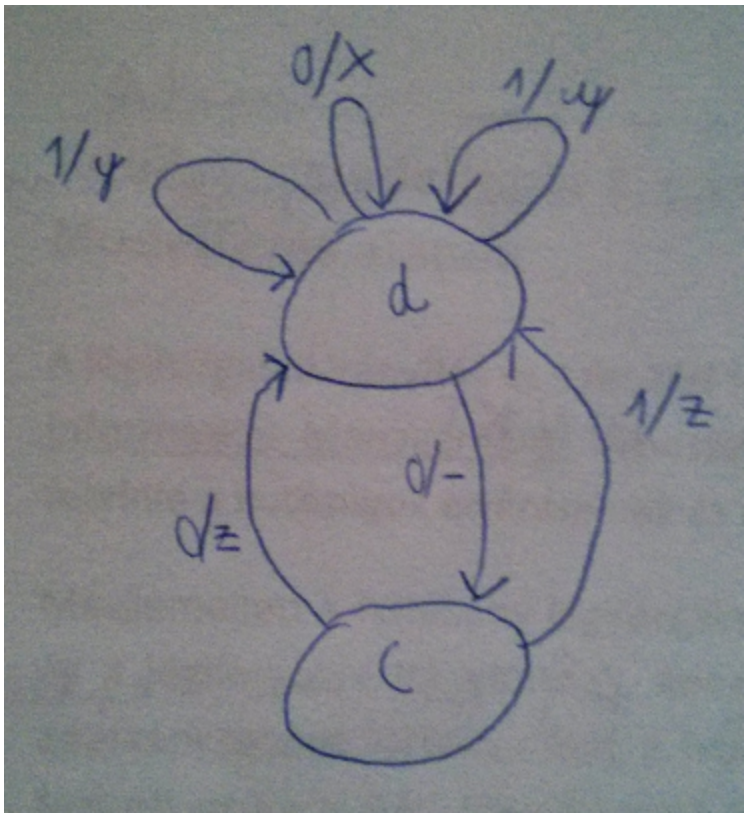
a. Determinisztikus-e a viselkedésmodell? Hozzávehető-e, ill. elhagyható-e egyetlen állapotátmeneti szabály, hogy ez megváltozzon?

Nem determinisztikus, mert nem tudjuk megmondani, hogy C állapotból mikor ad ki z-t. Ha ezt az élt elhagyjuk determinisztikus lesz.

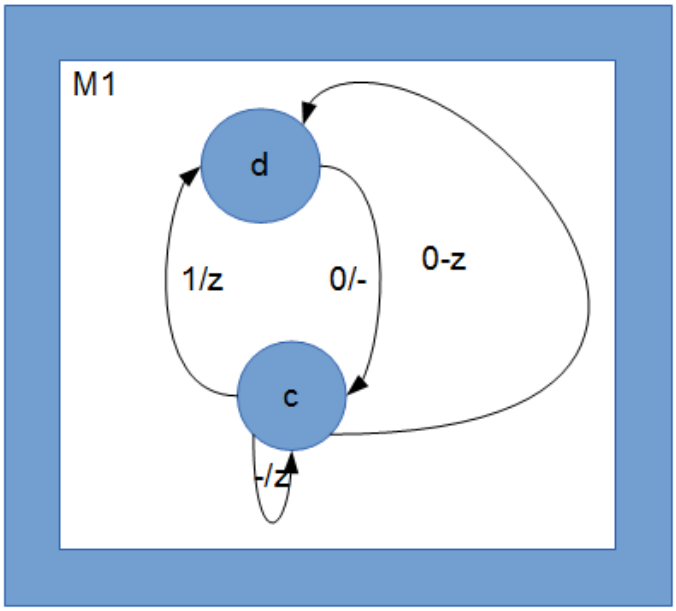
Akkor determinisztikus, ha minden állapotból egyértelmű, hogy adott bemenet hatására melyik állapotba megy tovább, azaz az állapotokból egy bemenethez csak egy nyíl van.
Az állapotgép ez alapján determinisztikus.--><*Én szerintem meg nagyon nem determinitsztikus, mivel a “C” állapotban a -/z nyíl hatására önmagába megy, és még van 2 másik nyíl is “A” és “B” állapotba, így nem tudjuk egyértelműen eldönteni onnan merre menjünk…. ez alapján NEM determinisztikus és ha elhagynánk azt a nyilat akkor az lenne (FIX ME)*
Ha jól értem, állapot átemeneti szabály alatt a nyilakat érti. Ha nyilat elhagyunk, bármelyiket, a determinisztikusság nem sérül. Ha hozzáveszünk új nyílat az állpotgéphez, akkor már sérülhet, hisz fel tudunk venni úgy nyilat, hogy egy állapotból egy bemenet hatására több új állapotba is eljuthassunk.

b. Absztraháld az állapotgépet {a,b} d állapotabsztrakcióval! Hány lehetőség van?

Absztrakció növelésén, azaz absztraháláson azt értjük, hogy az adatrejtés mértékét növeljük. Itt: az ‘a’ és ‘b’ állapotokat egy új állapottá ‘d’-vé vonjuk össze.
Erre csak egy lehetőségünk van, mert az összevont állapotba ugyanazon élek mehetnek be, mint a-ba és b-be kívülről és ugyanazok jöhetnek ki.



Belső éleket is meg kell tartani! (1/y-t nem kötelező 2x odaírni)



Mellesleg ez sem determinisztikus.

c. Finomítsd az állapotgépet a -> {e,f} állapotfinomítással! Hány lehetőség van?
A feladat: valamelyik állapotot helyettesítsük két újjal. Erre sok lehetőségünk van, hiszen bármely állapotot felbonthatjuk. Kívülről ugyanazok a nyilak jönnek bele az új állapotba és ugyanazok mennek ki belőle, mint az eredeti állapotnál, de a két új állapot közti nyilazást is meg kell csinálni. Erre is sok lehetőség van.
Itt sem szabad megfeledkezni az 'a' állapoton belüli nyílról. A felbontásban annak meg kell jelennie.

d. Absztraháld az állapotgépet {0,1} -> w tokenabsztrakcióval! Hány lehetőség van?
Mit kér a feladat? Az adatfolyam hálók azt modellezik, hogy az adatvezérelt rendszerünkben az adatokkal a függvények mit csinálnak: milyen bemeneti adatból milyen kimeneti adatot csinálnak. A tokenek reprezentálják az adatot. Azaz két fajta adatot (nulla és egy) vonjunk össze eggyé. Pl erre: a pincér a függvény (adatfolyamháló egy csomópontja), és a rendelésünk a token. Idáig kérhettünk tőle sajtburgert és csibeburgert is, most már csak hamburgert, mert a kétféle burgert összevontuk.
Az új DFN-en tehát az idáigi két féle token helyett csak egy lesz, azaz vagy jön adat, vagy nem.
Ezt egy féle képp csinálhatjuk meg.
Az állapotok közti nyilazást (tüzelési szabályokat) updatelni kell: az új bejövő token hatására történhet csak valami, ha két állapot közt egyforma nyilak mennek az új diagrammon, csak egyet tartunk meg.

e. Finomítsd az állapotgépet z -> {z1,z2} tokenfinomítással! Hány lehetőség van?
3 helyen ment eddig z, ezeket kell z1-re vagy z2-re lecselérlni. Egy élen két féle értékre tudjuk lecserélni, 3 élünk van, így a megoldás 2*2*2=2^3

Egy jelenlegi token helyett két újat veszünk fel. Abból, hogy melyik token helyett vesszük fel a két újat nincs olyan sok lehetőség, de a tűzelési szabályoknál ez sok féle képp jelenthet meg.
A két új token, annak aminek a helyére felvettük őket, a specializált változata. Lásd korábbi megoldásban: hamburger helyett sajt és csibeburgeres példa.
Amelyik csomópontok közt idáig hamburger ment, ott most el kell dönteni, hogy sajt vagy csibe burger mehet-e.
Valamelyik biztosan.

6. Készítsd el az M1 és M2 állapotgépek…
a. … szinkron szorzatát!
b. … aszinkron szorzatát!

Állapotgépek, pontosabban állapotgépek által reprezentált állapotterek szorzatára vagyunk kíváncsiak. A DFN reprezentálja a rendszer pillanatnyi állapotát, pl egyidejűleg lehet a és g állapotban is, de a-ban és b-ben nem, hisz a csomópontjaink egy-egy állapotteret reprezentálnak.
Mi a rendszer összes lehetséges állapotára vagyunk kíváncsiak:ag, ah, bg, bh, cg, ch
Tehát ez az állapotgépek szorzata.
…
Felírod a közös állapotokat és elkezded ezen állapotok mentén felírni az átmeneteket, ag-hez mind az a-t mind a g-t érintő átmenetek tartoznak.
Szinkron és asszinkron között az a különbség, hogy szinkronnál egy bemenetre mind a két állapotod változhat (tehát pl mehetsz ag-ből bh-ba), asszinkronnál csak az egyik változhat (ag-ből csak ah-ba vagy bg-be mehetsz)

3. gyakorlat - Teljesítménymodellezés

3. gyak Rendszermodellezés - Teljesítményelemzés gyakorlat 2013.10.07.

1. Little törvény

Csúcsidőben átlagosan 20 kliens érkezik egy fodrász szalonba óránként. Átlagosan egy kliens 15 percet tölt el a szalonban. Átlagosan hány kliens tartózkodik egyszerre a szalonban?

N: rendszerben lévő kérések átlagos száma
R: rendszerben töltött idő átlagosan
X: átbocsátási ráta

X=20 kliens/óra R= 15 perc = ¼ óra
Little törvény: N= X*R = 20*¼ = 5 kliens

2. Kihasználtság számítás

Egy diszk 50 kérést szolgál ki másodpercenként. Minden kérés kiszolgálása 0.005 másodpercet vesz igénybe. A rendszerben nincs átlapolódás.

a. Mekkora a kihasználtság?
X=50 kérés/sec
R=0.005 sec
N= X*R = 0.25 kérés
Mivel nincs átlapolódás -> C=1
U=N/C =25%

Vagy: U = S * X = 0,005 * 50 = 0,25

b. Mekkora a maximális kiszolgálási intenzitás?

U=X/Xmax -> Xmax = X/U =50/0.25 =200

3. Ciklust tartalmazó folyamat

Vállalatunk nyilvános szakmai tudástára egymásra is hivatkozó szócikkeket kínál a cég termékeit világszerte használó ügyfeleknek. Egy szócikk lekérés kiszolgálásának szerveroldali igénybevételének ideje normális eloszlással modellezhető, 60ms várható értékkel és 45ms szórással. A szócikk megtekintése után az olvasó csak 30% valószínűséggel hagyja el az oldalt, az esetek többségében ugyanis egy újabb szócikkre mutató hivatkozásra kattint.
a. Egy olvasó tudásszomjának kielégítéséhez összesen átlagosan mekkora szerveridő szükséges?

60ms/szócikk
szórás=+-45ms/szócikk (nem foglalkozunk vele)

(Egy felhasználó hány cikket olvas meghatározása)
a=1+ 30%*0 +70%*a (1 szócikket biztos elolvas a felhasználó, ezután 30% eséllyel egyet sem, 70% eséllyel pedig egy újabbat, rekurzió)
a=10/3 szócikk/user
Bővebben: Mivel nincs lehetőséged rekurziót implementálni ZHn, zárt alakban írod fel. Végiggondolva:
1 + 0.7 * (1 + 0.7 * (…
ami => SUM i=0 to inf (0.7^i)

Mértani sor összegképletével: 1 / (1-0.7) = 1 / 0.3 = 3.33, ami fentebb is kijött.

Ezek után Dsz = Vsz x Ssz = 10/3 szócikk/user * 60 ms/szócikk =200 ms/user idő szükséges

b. Tekintsük úgy, hogy az egyes kérések a szerveren nem párhuzamosíthatóak. Óránként hány egyedi látogatót képes kiszolgálni a szerver?

Umax = 1 (100%) = Xmax * S = Xmax * 200 ms/ user -> Xmax= 5 user/sec
átváltva órába:
5 user/sec * 3600 sec/óra = 18000 user/óra

4. Terheléselosztás

Adott egy webszerver (WS) és két fürtözött adatbázisszerver (DB1, DB2). A két adatbázis szerver közt súlyozott RR terheléselosztás alapján választunk, 1:2 arányban. Minden felhasználói kérés kiszolgálása során mindkét fajta erőforrást használjuk. A csúcsidőszakban 30 percig monitorozzuk a rendszert, ezalatt 9000 kérést szolgál ki. A szerveken mért foglaltsági idők: WS - 1350s CPU idő; DB1 - 810s, DB2 - 1320s diszk IO idő.
a. Mekkora az egyes szerverek jelenlegi átbocsátása?
S_{ws}=1350/9000=0,15sec/kérés
U_{ws}=1350/1800=75%
X_{ws}= U/S
X_{ws}=5kérés/sec

S_{db1}=810/3000=0,27sec/kérés
U_{db1}=810/1800=45%
X_{db1}=1,67kérés/sec

S_{db2}=1320/6000=0,22sec/kérés
U_{db2}=1320/1800=73%
X_{db2}=3,3kérés/sec

b. Mekkora a rendszer maximális áteresztőképessége?
X_{wsmax}=5/0,75=6,6kérés/sec

X_{db1max}=1,67/0,45=3,7kérés/sec
X_{db2max}=3,3/0,73=4,56kérés/sec

X_{db}=3,7+4,56=8,26kérés/sec

max áteresztőképesség: 6,6kérés/sec

c. Mennyi időt töltenek egy-egy kérés kiszolgálásával a szerverek?
S_{ws}=0,15sec/kérés
S_{db1}=0,27sec/kérés
S_{db2}=0,22sec/kérés

d. Miért nem egyféle foglaltsági időt vettünk figyelembe a két erőforrástípusnál?
mert a rendszer részeit külön akartuk vizsgálni

e. Hol csal még így is a modell?
átviteli idő, lineáris skálázódás

5. Forced Flow törvény

Egy adatbázis szervert 15 percig monitorozunk. Ez alatt az idő alatt a szerver processzora 12 percig volt foglalt. Azt figyeltük meg, hogy minden tranzakció általában 2-szer használta a processzort, és átlagosan 1 ms ideig használatonként (és ezalatt teljesen lefoglalja a CPU-t, nincs párhuzamosítás).
Mekkora a rendszer átbocsátása és áteresztő képessége?
$T_{össz}$=900sec
B_{cpu}=720sec
U_{cpu}=80%

V=2db/tranz
B_{tranz}=1ms
$S=V*B_{tranz}$=2ms/kérés
U=X*S
$1=X_{max}*0,002$
X_{max}=500kérés/sec => 100% (áteresztő)
80% => 400kérés/sec (átbocsátó)

6. Szolgáltatás igény törvénye (Service Demand)

Egy web szervert monitorozunk 10 percig, amely időtartam alatt a processzor 90%-ban volt foglalt. A web szerver logját megvizsgálva azt állapítottuk meg, hogy ezalatt az időtartam alatt 30 000 kérést szolgált ki. Mekkora a web szerver processzorra vonatkozó szolgáltatási igénye?
Di=Bi/C0 => Bi=foglaltsági ideje a monitorozás alatt C0=tranzakciók száma
$T_{össz}$=600sec
B_{cpu}=540sec

Di=540sec/30000kérés=0,018sec/kérés

7. Összetett teljesítménymodell

Egy sziget lakói minden reggel munkába menet átkeltnek a szigetet ölelő tavon. Észak felé híd vezet, dél felé autóskomp. Az irányonként egysávos híd 200m hosszú, és 60 km/h sebességgel szabad rajta haladni, a követési távolság (hátsó lámpától hátsó lámpáig 30m) betartása mellett. A négy komphajó egyenként 15 percenként teszi meg a sziget-szárazföld-sziget kört, és így óránként négyen együtt legfeljebb 800 autót tudnak átvinni a szárazföldre.
a. Mekkora a híd átbocsátóképessége (észak felé)? (1p)
A hídon egyszerre 6 autó lehet
Átérni a hídon 12 sec
X=N/S=6[autó]/12[sec] = 0,5 [autó]/[sec]

b. Hány autó fér el egy kompban? (1p)
1óra 800autó 4komp
1óra 200autó 1komp
15perc 50autó 1komp

Tehát 50 autó van egy kompban.

c. A reggeli csúcsforgalomban mekkora a szigetet elhagyó két útvonal együttes átbocsátóképessége? (1p)
Híd: 1800 [autó]/[óra] <- ez rossz 2000 a híd (2000, ha 6,666 autóval számolunk a hídon egyszerre, 1800, ha 6 autóval)
Komp: 800 [autó]/[óra]

Össz: 2**8**00 [autó]/[óra]

d. Ha délben a szárazföldi főutat baleset miatt lezárták, és a szigeten keresztül (a hídon, majd a kompon átkelve) terelik a forgalmat, mekkora a terelőútvonal átbocsátóképessége? (1p)
800[autó]/[óra]

e. Valamelyik reggel 7:00 és 8:30 között 900 autó hagyta el a szigetet komppal. Mennyi volt ebben az időszakban a kompok átbocsátása és kihasználtsága? (1p)
1,5 óra 900 autó 4 komp
1 óra 600 autó 4 komp

900autó/90min= Xautó = 10 autó/perc
Xkomp=600 [autó]/[óra]
Ukomp=600/800=75%

f. A fenti mérésben átlagosan hány autó állt sorba egyszerre a parton, ha az autók jól időzítve, átlagosan fél perccel a beszállásuk előtt érkeztek kompkikötőhöz? (2p)
R=0.5 perc
N=?
N=R*Xautó = 0.5[perc]*10[autó]/[perc] = 5 autó

8. Szálkészlet

Legalább hány aktív szálat kell engedélyeznünk egy webszerveren alkalmazásunknak, ha az egyenletes terhelés melletti teljesítményét nem szeretnénk visszafogni? Szálkorlát nélküli mérésekkel megállapítottuk, hogy egy kérés átlagosan 120 ezredmásodpercet tölt a rendszerben, és a szerver másodpercenként 50 felhasználót szolgál ki.
R=0,12[sec]
X=50[user]/[sec]

N=50[user]/[sec] * 0,12[sec] = 6[user]

6 szálat kell indítani.

9. Infrastruktúra méretezése

Internetes közösségi oldalt működtetünk. Az utóbbi időben számottevően megnőtt a népszerűsége, de ezáltal a válaszidő is kellemetlenül megnyúlt. Az üzleti cél, hogy csúcsidőszakban egyszerre 1500 felhasználót átlagosan négy másodperces válaszidővel szolgáljon ki a honlap.
a) Minimálisan mekkorára kell tervezni a kiszolgáló infrastruktúra átbocsátóképességét, ha az azon kívüli késleltetés (hálózati forgalom, HTML megjelenítés a kliensoldalon) egy másodpercnek becsülhető?
1500 user
válaszidő < 4 sec => késleltetés miatt: válaszidő < 3 sec
átbocsátó képesség: X=N/R=1500[user]/3[sec]=500[user]/[sec]

b) Az újratervezett weboldalon a mérések szerint egyetlen kérés kiszolgálása átlagosan 20ms CPU-időt igényel a webszerveren, és 12.5ms erejéig foglal le egy adatbázisszervert. Jelenleg 15 webszerver fogadja a kéréseket és az adatbázis 5 kiszolgálóra van replikálva. Lineáris skálázhatóságot feltételezve, milyen számítógépből és mennyit kell még legalább venni, hogy a fenti cél teljesülhessen?
Sws=20[ms]/[user] => 0,02[sec]/[user] 15darab

Sdb=12,5[ms]/[user] => 0,0125[sec]/[user] 5darab

Dws=500[user]/[sec] * 0,02[sec]/[user] = 10 darab => 10darab webszerver dolgozik adott terhelés alatt
Ddb=500[user]/[sec] * 0,0125[sec]/[user] = 6,25 => 7 darab adatbázisszerver dolgozik adott terhelés alatt, tehát még kettőt venni kell.

c) A kibővített rendszerben mekkora lesz az egyes szervertípusok kihasználtsági aránya? Ha az a cél, hogy még a csúcsidőszakban is legfeljebb 50%-os legyen a kihasználtság, meddig kellene még bővíteni a rendszert?
Uws=10/15=66%
Udb=6,25/7=89%

U'ws=10/20=50% => Webszerverből 20 darab
U'db=6,25/13=48% => Adatbázisszerverből 13 darab kell.

4. Rendszermodellezés gyakorlat, 2013.10.15.

Adatfolyamháló

Elakadásjelző háromszögeket előállító gyárunkat adatfolyamhálóval modellezzük. A háló kezdetben két csomópontot tartalmaz: az első csomópont egy gép, amely fényvisszaverő oldallapokat állít elő, és a futószalagra helyezi őket; a második csomópont az összeszerelő gép, amely a futószalagról felveszi a lapokat, és időnként egy összeszerelt háromszöget bocsájt ki az egész háló kimenetén.
a. Készítsük el a feladat adatfolyamháló modelljét.
b. Finomítsuk az új modellt is a következőképp: az első gép időnként deformált oldallapokat gyárt, amelyet a második képtelen feldolgozni.
c. Végül finomítsuk tovább a modellt a következőképp: az összeszerelő gép az eredeti funkcionalitás elé kapcsolva tartalmaz egy bevizsgáló berendezést is, amely képes kidobni a deformált lapokat.
d. Finomítsuk a modellt a következőképp: az összeszerelő gép mindig bevár három fényvisszaverő lapot, és belőlük szerel össze egy háromszöget.

Kísérlettervezés feladatok

1. Kísérlet kiértékelése a

Infrastruktúránk méretezését megnehezíti, hogy egy adott feladttípus végrehajtási ideje a körülmények függvényében ingadozik, például lapozás, memória szemétgyűjtés, memória cache találatok stb. változékonysága folytán. Ezért összeállítottunk egy valós munkaterhelést jól jellemző benchmarkot, és ennek többszöri lefuttatása során a futási időket átlagolva szeretnénk meghatározni a rendszer átlagos teljesítményét.
a. Az első tíz futtatás eredményei: 37s, 34s, 35s, 39s, 57s, 41s, 36s, 35s, 61s, 35s. Mennyi ez alapján a rövid kísérlet alapján a tapasztalati átlag és tapasztalati szórás?
Tap átlag = SUM(xi)/t = 41s
(t a kísérletek száma)
Tap szórás = GYÖK((SUM(Xi-TAP)^2)/(t-1)) = 9,76 (<- korrigált tap. szórás)
http://hu.wikipedia.org/wiki/Tapasztalati_sz%C3%B3r%C3%A1s

b. Nagyobb léptékben futtatva a kísérletet, a benchmark 10 000 futtatása átlagban 44.3 másodpercig tartott, 11.6 másodperc tapasztalati szórással. Mennyire lehetünk biztosak a kapott eredmény pontosságában?
t=10000
/X=44,3s
d=11,6s
d/gyök(t)=0,1s tehát

68%, hogy 44,3 ± 0,1s
95%, hogy 44,3 ± 2*0,1s
99,7%, hogy 44,3 ± 3*0,1s

2. Kísérlettervezés

Egy modellezett folyamat átbocsátóképességére szimuláció alapján szeretnénk egy közelítő értéket és hozzá tartozó konfidencia-intervallumot meghatározni.
a. Hány szimuláció mérési eredményeiből számoljunk átlagot?
legalább 100 (Fixme: a centrális határeloszlás tétele miatt)<- Azért mert van olyan ökölszabály, ha ismretlen a szórás akkor legalább száz, ha ismert akkor legalább 30.
b. Az így elvégzett mérési eredmények tapasztalati közepe 500 kérés / sec; a tapasztalati szórás 10%. Szeretnénk, hogy 95% konfidencia mellett egy legfeljebb 40 kérés / sec széles

intervallumba essen az átbocsátóképesség. Hány mérést végezzünk még?

d=500*0,1=50
95% => 2σ
40 kérés/sec széles => +-2σ
40 = 4σ
σ = 10

σ=d/sqrt(t)
10=50/sqrt(t)
t=25

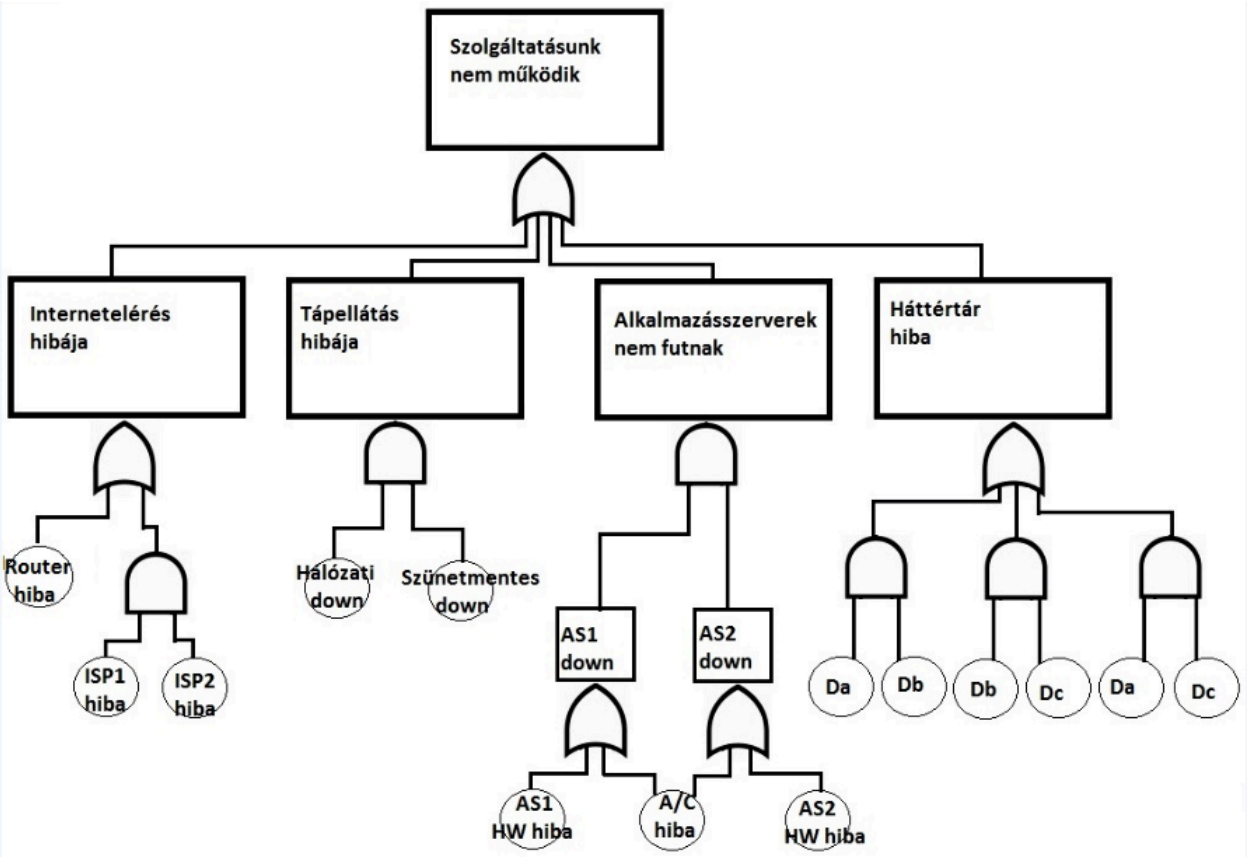
???
t=25 jelentése, hogy legalább 25 mérést kell végeznünk a kívánt konfidencia eléréséhez. Az a, feladatrészben viszont azt mondtuk, hogy legalább 100 mérés kell. Így MAX[25,100]=100<- legalább 25öt kell mérnünk, így 100-at mérni felesleges, azt csak az ökölszabály miatt mondtuk.

Emlékeztető: a σ szórású normális eloszlás 68%-os konfidenciaintervalluma 1σ, a 95%-os 2σ, a 99.7%-os 3σ sugarú

6. gyak Hibafa

Internetes reklámcégünk szolgáltatása ügyfélre szabott hirdetéseket ajánl fel weboldalak részére. Szolgáltatásunk működéséhez szükséges, hogy a szerverteremből elérhető legyen az internet, rendben legyen a tápellátás, a meleg tartalék rendszerben üzemeltetett két alkalmazásszerver közül legalább az egyik fusson, és az általuk közösen használt külső SAN háttértár kifogástalan állapotban legyen. Mindkét alkalmazásszerver kiesését belső hardware meghibásodás vagy a szerverterem hűtésének leállása okozhatja. A netcsatlakozás működése a router üzemképességén múlik, és persze a két ISP legalább egyikének szolgáltatnia kell. A hibatűrő SAN tárolórendszer RAID-5 elven három merevlemezt tartalmaz (Da, Db, Dc), és egyetlen (tetszőleges) lemez kiesését még tolerálja. A szerverterem tápellátása mindaddig biztosított, amíg a hálózati tápellátással és a szünetmentes tápegységgel nincs egyszerre gond.

a. Rajzolja le a „nem érhető el a szolgáltatás” rendszerszintű hibajelenség hibafáját!
Megoldás:



Megjegyzés: a "tovább ki nem bontott elemek" gyémánt (rombusz) alakban jelölendők; kör: alapesemény.

b. A hibafa-redukció módszerrel azonosítsa a rendszer egyszeres hibapontjait és kritikus eseményeit!

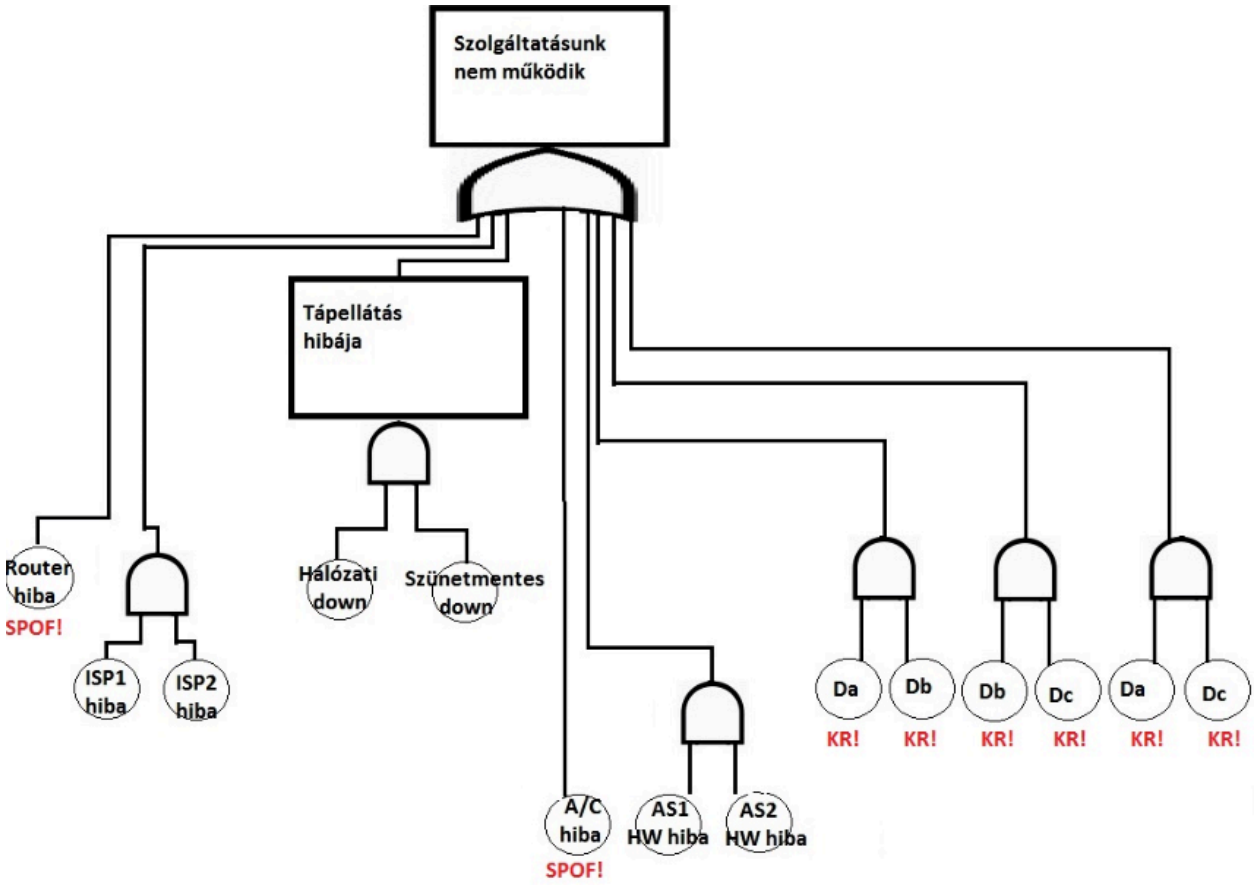
SPOF(Single Point of Failure, avagy egyszeres hibapont): Egyszerűen fogalmazva azok a hibák, melyek ha fellépnek, az egész rendszer leáll. Tehát ha a rendszer egyetlen eleme leáll, és ennek következtében az egész rendszer meghibásodik, akkor az SPOF.

Kritikus esemény: több úton is hibajelenséget okoz

Ezek megtalálására Digit1-ben tanult diszjunktív normálformát kell találni (Sok ÉS összeVAGYolva)

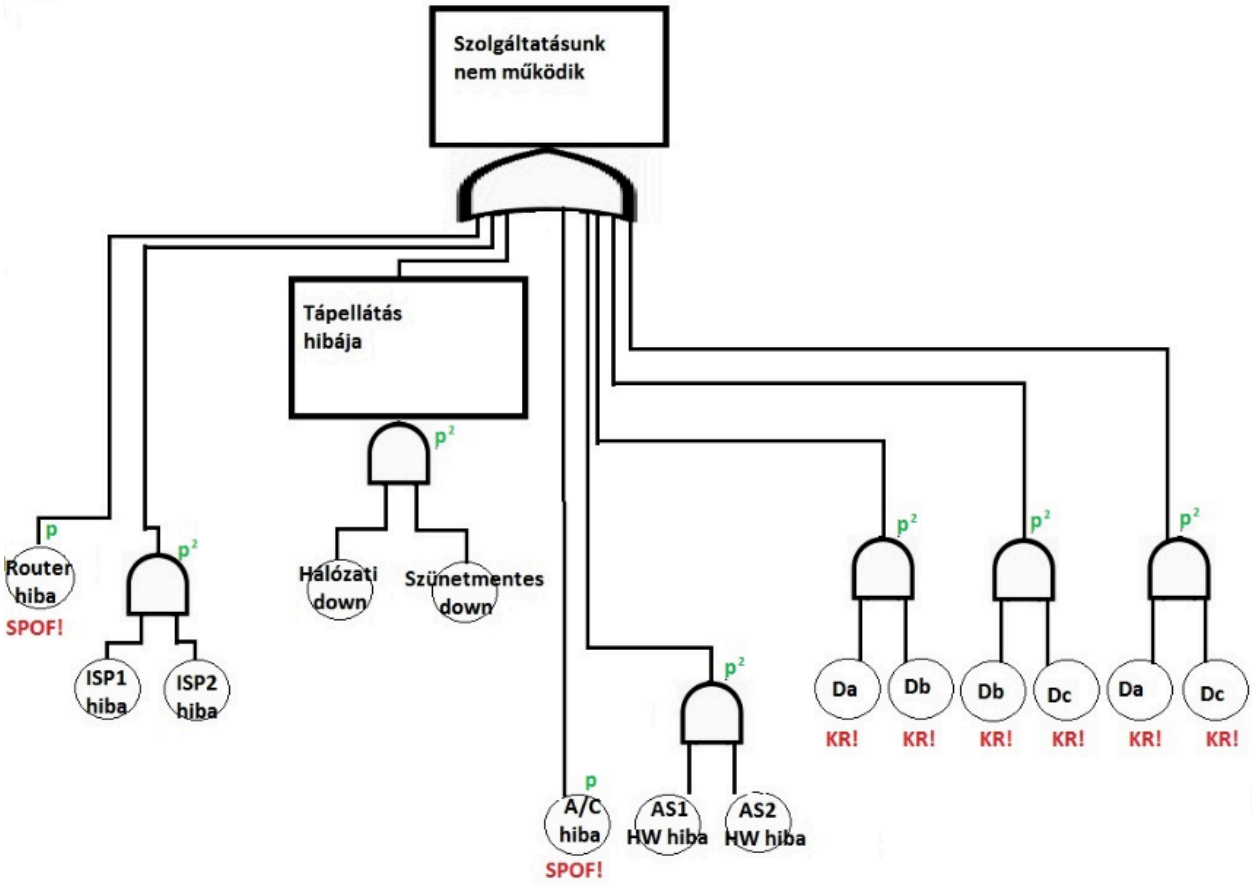
- RouterHiba v (ISP1hiba ^ ISP2hiba)
- HálózatiHiba ^ SzünetmentesDown
- (HWhiba1 ∨ AChiba) ∧ (HWhiba2 ∨ AChiba)
    - A fa legtöbb ága triviális, az “alkalmazászerverek nem futnak” az egyetlen, ami kisebb gondot okoz, erre algebrai alakban felírhatjuk:
        - (HWhiba1 ∨ AChiba) ∧ (HWhiba2 ∨ AChiba), mely diszjunktív normál alakban: (HWhiba1 ∧ HWhiba2) ∨ AChiba
        - (Mivel halmazelmélet → disztributivitás, de Morgan-azonosságok, stb. egyike: (A v C) ^ (B v C) = (A ^ B) v C. BTW hasonlóan: (A v B) ^ C = (A^C) v (B^C). Meg ha esetleg ilyen kellene: NOT(A^B) = NOT(A) v NOT(B); NOT(AvB) = NOT(A) ^ NOT(B))
- (Da ^ Db) v (Db ^ Dc) v (Da ^ Dc)

Tehát ezután így fog kinézni a REDUKÁLT HIBAFA:



c. Az összes elemi hibaállapot tetszőleges időpillanatban $p=10^{-5}$ valószínűséggel áll fent. Mi következik a rendszerszintű hibajelenségre? Milyen feltételezéseket és közelítéseket kellett tenni az elvégzett számításhoz?

Itt a redukált hibafa alapján kell számolnunk, azt feltételezve, hogy a hibák függetlenek egymástól. A hibák valószínűsége a fán zölddel van jelölve:



2 esemény közül minimum egy bekövetkezésének valószínűsége: az események külön-külön bekövetkezésének valószínűsége - az események együttes bekövetkezésének valószínűsége.
Tehát: P(AvB) = P(A) + P(B) - P(A^B) (<= P(A) + P(B) természetesen)

A keresett valószínűség: $p+p^2+p^2+p+p^2+p^2+p^2+p^2 = 2p + 6p^2 = 2*10^{-5}+6*(10^{-5})^2$ = ez kb. $2*10^{-5}$.