6. gyakorlat: Megbízhatósági modellezés és analízis

A gyakorlaton a megbízhatósági modellezés módszereivel fogunk foglalkozni.

1. Egy email szolgáltatás megbízhatósági modellje (megbízhatósági blokkdiagram)

Egy vállalat levelező (email) szolgáltatásának működéséről a következőket tudjuk:

- A levél küldése a munkatárs *munkaállomásáról* történik és azt a vállalati *mail szervere* továbbítja a címzettnek.
- A vállalat munkaállomásai és szerverei egy *belső hálózaton* keresztül vannak összekötve.
- A munkaállomásról a vállalati mail szerver eléréséhez, valamint a kimenő levelek továbbküldéséhez ismerni kell a célszerverek IP címeit. A vállalaton belül melegtartalékoltan *2 DNS szerver* van (elsődleges és másodlagos).
- A belső hálózatról az internet elérése egy *útválasztón* (router) keresztül történik. A vállalatnak 2 internet szolgáltatóval van kapcsolata, ezek mint *elsődleges* illetve *másodlagos internet kapcsolat* használhatók. Az összeköttetések megbízhatósági adatai ismertek.

Mivel a kimenő levelek további sorsáról nincs ismeretünk, a megbízhatósági analízist eddig a lépésig végezzük el.

Az egyes komponensek megbízhatósági adatai a következők (órában mérve):

	Munka-állomás	Belső hálózat	Mail server	DNS szerverek	Útválasztó (router)	Elsődleges internet kapcsolat	Másodlagos internet kapcsolat
MTTF	4500	45000	4000	4500	9000	13500	5400
MTTR	2	3	2	2	1	4	6

Feladat:

1. Rajzoljuk meg a rendszer megbízhatósági blokkdiagramját!
2. Számítsuk ki a levelezés szolgáltatás rendelkezésre állását (egy munkatárs szemszögéből) a fenti adatokat felhasználva!
3. Adjuk meg, átlagosan hány óra kiesésre kell számítanunk egy évben!

2. Egy TMR rendszer megbízhatósági modellje (Markov lánc)

Adott egy 3 azonos komponensből álló, többségi szavazást megvalósító redundáns rendszer, nem ideális szavazóval. A komponensek meghibásodásai egymástól függetlenek, meghibásodási tényezőjük λ, javítási tényezőjük μ. Egyszerre csak egy komponens javítható. A szavazó meghibásodási tényezője λ_{sz}. A szavazó hibáját önellenőrzés észleli, ez esetben a rendszer leáll, és teljes javítása következik μ_r javítási tényezővel.

Feladat:

- Rajzoljuk fel a rendszer megbízhatósági modelljét *Markov lánc* felhasználásával!
- Adjuk meg, hogyan számolható ki a rendszerszintű *rendelkezésre állás* illetve a rendszerszintű *megbízhatóság*, ha a Markov lánc megoldásaként megkaphatók az egyes állapotvalószínűségek időfüggvényei!

3. Egy szerver fürt megbízhatósági modellje (sztochasztikus Petri háló)

Egy szerver fürt *k+h* szerverből áll. A kezdeti konfigurációban *k* számú aktív szerver van, és *h* számú tartalék, amelyek nincsenek bekapcsolva. A működésről a következőket tudjuk:

- Az aktív szerverek λ meghibásodási tényezővel hibásodnak meg.
- A hibás szervert azonnal elkezdik javítani.
- Egy hibás szerver javítási ideje μ paraméterű exponenciális eloszlással jellemezhető.
- Egyszerre több szerver is javítható (több szerelő is tud dolgozni).
- Ha egy aktív szerver meghibásodik, akkor megpróbálnak helyére egy tartalékot bekapcsolni (ha van még tartalék).
- A tartalék szerverről bekapcsoláskor *c* valószínűséggel kiderül, hogy nem képes elindulni, azaz hibásnak bizonyul (és ennek megfelelően *1-c* valószínűséggel képes elindulni, azaz hibamentesnek bizonyul). A hibásnak bizonyult tartalék szervert is javítani kell.
- Egy megjavított szervert csak akkor kapcsolnak ki (és lesz belőle így tartalék), ha éppen nincs meghibásodott szerver, aminek a helyére lenne illeszthető.

Rendszerszintű hiba akkor következik be, ha egy aktív szerver sincs a rendszerben.

Feladatok:

1. Készítsük el a rendszer megbízhatósági modelljét a *GSPN formalizmust használva*, megengedve az élsúlyok és a jelöléstől függő paraméterek használatát is!
2. Adjuk meg, hogyan számítanánk ki a rendszer *rendelkezésre állását*! A rendszer akkor működőképes, ha legalább egy hibamentes aktív szerver van.
3. Adjuk meg, hogyan tudnánk a legegyszerűbben kiszámítani a rendszer *megbízhatóságát*!
4. Adjuk meg, hogyan kellene módosítani a modellt akkor, ha *egyszerre csak egy* szerver lenne javítható!

4. Egy irányítórendszer veszély analízise (sztochasztikus Petri háló, Stochastic PetriDotNet eszköz)

Egy kritikus ipari folyamat irányítórendszerét úgynevezett *kétcsatornás architektúra* szerint alakították ki. Ezt a következőket jelenti:

- Az ipari folyamatból származó bemeneti adatok feldolgozása *két számítógépen* történik (mindkét számítógép párhuzamosan megkap minden bemenetet).
- A vezérlési kimeneteket mindkét számítógép előállítja, ezek felhasználását (a számítógépek esetén előforduló hibák detektálása és kezelése érdekében) egy külön áramkörrel megvalósított *arbiter logika* határozza meg.
- A számítógépek rendszeresen *öntesztet* hajtanak végre, ennek eredményét kapja meg az arbiter logika.

Az arbiter logika működése a következő:

- A kimenetek felhasználhatók, ha nem jeleznek az öntesztek hibát.
- Ha mindkét számítógép öntesztje hibát jelez, akkor az irányítórendszer *detektált hibával lekapcsol*.
- Ha az egyik számítógép öntesztje hibát jelez, akkor azt az arbiter lekapcsolja, és ezután a másik számítógép működik önállóan (egycsatornás üzemmódban). Az önteszt viszonylag kis hatékonysága miatt az egycsatornás üzemmód működési ideje korlátozva van, ennek letelte után az irányítórendszer *egycsatornásan lekapcsol*.
- Ha egycsatornás üzemmódban az aktív számítógépen az önteszt hibát jelez, akkor az irányítórendszer *detektált hibával lekapcsol*.

Az irányítórendszer lekapcsolása biztonságos leállásnak tekinthető (irányítás nélkül az ipari folyamat is leáll). Ha az irányítórendszer kimenete detektálatlanul hibás, az *veszélyes állapotnak* tekinthető.

A rendszer paraméterei a következők:

- Egy számítógép meghibásodási gyakorisága 5,0E-06 1/óra.
- Az önteszt átlagosan 2 másodpercenként fut le (gyakoriság: 1800 1/óra) és 70% valószínűséggel detektálja a számítógép hibáját. Hibátlan számítógép esetén hibajelzés (azaz téves hibajelzés) nincs.
- Az arbiter működése (a kimenetek frissítése) 0,2 másodpercenként történik (gyakoriság: 18000 1/óra).
- Az egycsatornás üzemmód működési ideje 10 perc (ezt átlagos értéknek tekintjük, gyakoriság: 6 1/óra).

Feladat:

1. Készítsük el a rendszer megbízhatósági modelljét a *PetriDotNet eszközt használva*! Tipp: Külön helyek reprezentálják a következő rendszerállapotokat: „normál működés”, „detektált hibával lekapcsolt irányítórendszer”, „egycsatornásan lekapcsolt irányítórendszer”, „veszélyes állapot” (detektálatlan hibás kimenet).

2. A *Mean Time to First Failure* analízis alkalmazásával számítsuk ki a „veszélyes állapot” bekövetkezésének várható idejét! Tipp: A kiszámolt MTFF értéket osztani kell a megadott *p* számmal a tényleges idő kinyeréséhez.
3. Ugyancsak a *Mean Time to First Failure* analízis alkalmazásával számítsuk ki a „detektált hibával lekapcsolt irányítórendszer” állapot bekövetkezésének várható idejét.
4. (Opcionális feladat) Az arbiterbe egy komparátort is beépítenek, ami összehasonlítja a két számítógép kimeneteit. Ha egyik számítógép öntesztje sem jelez, de a komparátor az egyes csatornák kimenetei között eltérést mutat, akkor az irányítórendszer *detektált hibával lekapcsol*. A komparálás átlagosan 0,2 másodpercenként történik, és 99% valószínűséggel detektálja a kimenetek eltérését. Azonos kimenetek esetén téves hibajelzés nincs. Számítsuk ki ez esetben is a „veszélyes állapot” bekövetkezésének várható idejét!
5. (Opcionális feladat) Az irányítórendszer lekapcsolása után (akár detektált, akár egycsatornás leállásról van szó) a várható javítási idő 2 óra (gyakoriság: 0,5 1/óra). Az irányítórendszer hibás kimenete esetén a probléma detektálása majd az irányítórendszer javítása és az ipari folyamatban okozott károk helyrehozása 50 órát igényel (gyakoriság: 0,02 1/óra). A javítások modellezése után a megfelelő reward hozzárendelésével számítsuk ki az irányítórendszer rendelkezésre állását (a veszélyes állapot nem jelent hibamentes szolgáltatást).