Markov-láncok állapotterének csökkentése állapotösszevonással

GOLDSCHMIDT LÁSZLÓ
Távközlési Kutató Intézet

ÖSSZEFOGLALÁS

A cikk a Markov-láncok elméleti alapjainak összefoglalása után a nagymegbízhatóságú berendezések megbízhatósági analízisére kifejlesztett számítógépes programrendszerek hatékonyságát növelő programcsomaggal foglalkozik. Ismerteti az ún. állapotösszevonást és az ezt alkalmazó „állapotösszevonó" programcsomagot, melynek segítségével a korábbi vizsgálatokhoz képest a felhasznált számítógép gépidő, valamint a szükséges tárkapacitás jelentősen csökken. Az „állapotösszevonó" programrendszer az összes Markov-modellel leírható rendszer analitikus ós szimulációs vizsgálatánál felhasználható.

1. Bevezetés

A megbízhatóság a bonyolult nagyértékű híradástechnikai berendezések egyik legfontosabb jellemzője. A megbízhatóság nem megfelelő értéke meghibásodásokhoz, balesetekhez, nagy anyagi veszteségekhez vezethet. A beruházóknak az előzetes gazdasági, megtérülési és elévülési idő számításokhoz szükséges ismerniük a meghibásodás gyakoriságát, az átlagos javítási időt, valamint az egyes alkatrészek megbízhatósági paramétereit a tartalékalkatrész-beszerzés, valamint a karbantartás megszervezéséhez.

A megbízhatóság számítása azonban nagyon bonyolult. A jellemző paraméterek ugyanis nemcsak a felhasznált alkatrészektől, hanem a készülék struktúrájától is függenek. Az egyre összetettebb berendezéseknél emiatt a klasszikus megbízhatóság-elméleti módszerek nem, vagy csak nehezen felhasználhatók [1], [3], [4]. A megoldást többek között a markovi modellt felhasználó eljárások jelentik. Ez azonban szükségessé tette számítógép felhasználását, melynek során lehetővé vált a rendszerek analízisén kívül az iteratív szintézis is.

A BME-HEI-ben kidolgozásra került a markovi modellt felhasználó PL/1 programnyelven írt számítógépes programrendszer, mely lehetővé teszi mind az analitikus, mind a szimulációs vizsgálatokat, batch és interaktív változatban egyaránt [6], [7], [8].

Az egyre részletesebb vizsgálatoknál, az egyre alacsonyabb strukturális szinteket leíró markovi modellnél újabb problémaként merült fel a rendszer lehetséges állapotai számának nagymértékű megnövekedése. Ezen cikkben olyan programcsomagot ismertetünk, mely az állapotteret redukálja, a markovitást nem befolyásoló, lehetséges állapotösszevonások felismerésével és elvégzésével [9].

Beérkezett: 1986. VII. 29. (□)

GOLDSCHMIDT LÁSZLÓ
Egyetemi diplomáját a Budapesti Műszaki Egyetem Villamosmérnöki Karán, a híradástechnika szak adatátviteli ágazatán szerezte 1982-ben. Azóta a Távközlési Kutató Intézetben dolgozik, ahol kezdetben áramkörfejlesztéssel foglalkozott. Jelenleg szoftver fejlesztési területen dolgozik. 1986-ban a Rádióhírközlő Szakmérnöki szakon kitüntetéses oklevelet szerzett.

A megbízhatóság-elméleti háttér áttekintése után az állapottér redukciójának szükségességét és módjait elemezzük, példával illusztrálva. Ezt az elkészült programrendszer sajátosságai, felépítése és működése követi. Végül a programrendszer alkalmazásához mintapéldát és értékelést láthatunk.

2. A megbízhatósági paraméterek és azok meghatározása

2.1. *A megbízhatósági követelmények osztályozása*

A megbízhatóság megkövetelt mértéke magától a berendezéstől és annak rendeltetésétől függ. A megbízhatósági paramétereket különböző szempontok alapján rangsorolhatjuk. Ezen szempontok alapján egy berendezés lehet:

a) — állandó kapacitású
— funkció-csökkenést megengedő
— hibás kimenetet megakadályozó
— adatbázist védő

b) — nagy megbízhatóságú
— rendelkezésre állás orientált

c) — javítható
— nem javítható (zárt)

d) — hibát nem tűrő (technológiai)
— hibatűrő (reduldancia)

2.2. *A megbízhatósági paraméterek nyerésének módszerei*

a) numerikus: — Markov-modell
— hálózat vagy megbízhatósági blokkdiagram-modell

b) kísérleti

A Markov-modellen alapuló módszerek lehetnek analitikusak vagy szimulációs rendszerűek.

2.3. *A megbízhatósági paraméterek és fogalmak értelmezése* [1], [3]

Azt, hogy egy berendezés adott időpillanatban működik (jó), úgy értelmezzük, hogy ebben a pillanatban adott specifikációt teljesít.

a) Rendszer-indítás:
Kezdő pillanat, a rendszer minden eleme jól működik.

b) Rendszer-meghibásodás:
A működő rendszer azon pillanata amikor meghibásodik.

c) Rendszer-javítás:
A működésképtelen rendszer azon pillanata, amikor újra működni kezd.

d) Működési idő: *(Up time)*
A rendszer javításától a következő meghibásodásig eltelt idő. (Várható értéke MUT)

e) Kiesési idő: *(Down time)*
Az utolsó meghibásodástól a rendszer javításáig eltelt idő.
(Várható értéke: MDT)

f) Ciklusidő *(Cycle time)*
$MCT = MUT + MDT$
(Várható értéke: MCT)

g) Első meghibásodás bekövetkezésének várható értéke: *(MTFF)*
Az első meghibásodásig eltelt idő, a rendszer indításától számítva.

h) A következő meghibásodás bekövetkezésének ideje és ennek várható értéke: *(MTTF)*
A rendszer egy tetszés szerinti működő időpontjától a következő meghibásodásig eltelt idő.

i) Pont használhatóság: $[P_W(t)]$
Annak a valószínűsége, hogy egy tetszés szerinti időpillanatban a rendszer működik.

j) Intervallum használhatóság: $[I(t)]$
A működési időarány várható értéke a $[0,\ t]$ időintervallumban.

$$I(t) = \frac{1}{t} \int_0^t P_w(u)\, du$$

ha ergodikus a markov folyamat (nem nyelő).

Aszimptotikusan:

$$A = \lim_{t \to \infty} \{P_w(t)\} = \lim_{t \to \infty} \{I(t)\}$$

Ezt az értéket nevezzük készenléti tényezőnek.

Az eddig említett paraméterek közötti összefüggésként említhető, hogy ha egy τ időintervallumban N meghibásodás következik be, akkor

$$MCT = \lim_{\tau \to \infty} \{\tau/N\}$$

$$A = \frac{MUT}{MCT}$$

összefüggések állnak fenn.

3. A markovi modell

A Markov-modell előtérbe kerülésének oka a klasszikus soros-párhuzamos átalakításokat tartalmazó módszerekkel szemben, többek között a klasszikus modell korlátaiban rejlik. A klasszikus modell csak javításmentes, kétállapotú rendszereket képes leírni, mely kétállapotú elemeket tartalmaz. A csökkentett terhelésű tartalék nem fér bele a modellbe. Ezzel szemben a Markov-modell leírja a javítható, többállapotú, csökkentett terhelésű tartalékokkal rendelkező rendszereket is.

3.1. A Markov-lánc definíciója és szemléltetése

Legyen egy $X = X_0 \cup X_1 \cup X_0 \cup \ldots \cup X_n$ véges diszkrét állapottér. Az $X_0 \ldots X_n$ állapotokat feleltessük meg egy rendszer lehetséges állapotainak. X-en értelmezzünk egy sztochasztikus folyamatot úgy, hogy az a lehetséges állapotváltozásokat írja le. Ezen állapotváltozások a vizsgált berendezés viselkedését adják meg, maguk az állapotok pedig a rendszer jellemző paraméterei által meghatározottak.

Az állapotváltozások a $t_0, t_1, \ldots t_k$ időpillanatokban következnek be. Ezt a folyamatot egy gráfstruktúrán szemléltethetjük. A modellező gráf csúcsai a rendszer lehetséges állapotait jelentik, az élek pedig, melyek irányítottak, az állapotok között lehetséges átmeneteket és azok irányát reprezentálják.

Szükséges néhány új fogalom értelmezése: Ha pozitív annak a valószínűsége, hogy egy P állapotból egy Q állapot véges t idő alatt elérhető, akkor a Q-t P-ből elérhetőnek nevezzük. Nyelőnek nevezzük azt az állapotot, amelybe kerülve a rendszer 1 valószínűséggel megmarad. Ezen két fogalom a szemléletes gráfmodellben azt jelenti, hogy ha egy Q állapot egy P-ből elérhető, akkor a P csúcsból a gráf élein, az élek irányításának megfelelően haladva, véges számú lépésben eljuthatunk a Q csúcsba. A nyelő állapot pedig egy olyan gráfcsúccsal jellemezhető, amelybe csak érkeznek élek, de onnan ki nem indulnak.

Tegyük fel, hogy a rendszer a t_k időpillanatban X_j állapotban van. Ha az, hogy melyik állapotba kerül a következő t_{k+1} időpillanatban, csak a pillanatnyi állapotától függ, attól nem hogyan jutott ebbe az állapotba, akkor a folyamat markovi.

$$P\{\xi t_{k+1} \in X_P \mid \xi_{t_k} \in X_j \ldots \mid \xi_{t_0} \in X_k\} =$$
$$= P\{\xi_{t_{k+1}} \in X_P \mid \xi_{t_k} \in X_j\}$$

3.2. Átmenet valószínűségek

Jelöljük a rendszerállapotok közötti átlépések valószínűségét

$$A_{ij}(t_k, t_{k+1}) = P\{\xi_{t_{k+1}} \in X_j \mid \xi_{t_k} \in X_i\}, \text{ ahol}$$

$$t_i \in [0, \infty]$$

A Markov folyamat folytonos vagy diszkrét idejű, attól függően, hogy az átlépések bármikor vagy csak diszkrét időpontokban következhetnek be:

$$A_{ij}(K) = P\{\xi_{t_{k+1}} \in X_j \mid \xi_{t_k} \in X_i\}$$

diszkrét idejű esetben.

Homogén Markov folyamatról akkor beszélünk, ha

$$A_{ij}(t, t^*) = A_{ij}(t^* - t)$$

tehát csak az időkülönbségtől függ az átlépési valószínűség.

3.3. Folytonos idejű Markov láncok

A folytonos idejű Markov folyamatra a következő megállapításokat tehetjük:

— diszkrét állapotterű
— véges sok állapotot tartalmaz
— az állapotok teljes eseményteret alkotnak
— az átlépési valószínűségek nemnegatívak.

Származtassuk az

$$a_{ij} = \lim_{\Delta t \to 0} \frac{A_{ij}(\Delta t)}{\Delta t}$$

kifejezéssel a rendszerállapotok közötti átlépések valószínűségeiből az átlépési gyakoriságot. Fizikai jelentése: Ha 10^5 óra alatt következik be egy átlépés, akkor az átlépési gyakoriság értéke 10^{-5}/óra. Ez az átlépés kezdő és végcsúcs jellegétől függően a meghibásodási vagy javítási tényező. A matematikai kezelhetőséghez szükséges még egy N dimenziós sorvektor bevezetése, N a rendszer állapotainak száma:

$\mathbf{P(t)} = [P_1(t), P_2(t) \ldots P_N(t)]$ ahol a k. elem a k. állapot valószínűségét reprezentálja. Az A_{ij} átlépési valószínűségek $N \times N$-es mátrix alakba rendezhetők, ahol az i-edik sor j-edik eleme egy feltételes valószínűség, amely azt adja meg, hogy milyen valószínűséggel kerül a rendszer a j-edik állapotba, ha előzőleg az i-edik állapotban volt. Az átlépési gyakoriságokat mátrix alakba rendezve

$d/dt\, \mathbf{P(t)} = \mathbf{P(t)} \cdot \mathbf{A}$

állapotegyenlethez juthatunk, ahol A az átlépési gyakoriságok mátrixa, [1], [4]. Ennek általános megoldása:

$\mathbf{P}(t) = \mathbf{P}(0) \cdot \exp(\mathbf{A} \cdot t)$, ahol $\mathbf{P}(0)$ a kezdeti időpillanatban az állapotok eloszlását adja.

Mivel $\sum_{i=1}^{N} P_i = 1$, valamint stacionárius esetben

$\mathbf{P} \cdot \mathbf{A} = 0$,

ezekből a megbízhatósági paraméterekre vonatkozó formulák a következő módon nyerhetők. Számozzuk a rendszer állapotait úgy, hogy 1-től k-ig a jó, $k+1$-től N-ig a rossz állapotokat jelöljük. Így a kezdeti időpillanatban a kezdeti eloszlásvektor

$P_W(t) = [P_1(t) \ldots P_k(t)]$
$P_F(t) = [P_{k+1}(t) \ldots P_N(t)]$

vektorokra bontható, az $\mathbf{A}$ mátrix pedig

$\mathbf{E}\ K \times K$-s $\quad \mathbf{F}\ K \times (N-K)$-s
$\mathbf{G}\ (N-K) \times K$-s $\quad \mathbf{M}\ (N-\mathbf{K}) \times (N-K)$-s

mátrixokra particionálható. Ezen mátrixok, valamint P_W és P_F vektorok segítségével felírhatók az állapotegyenletek. Ha Bartlett megközelítéssel élünk [2], akkor attól függően, hogy a rossz, vagy a jó állapotokat tekintjük nyelőnek,

$\mathbf{G} = \mathbf{M} = 0$ vagy $\mathbf{G} = \mathbf{E} = 0$.

Mindkét feltétel mellett felírva az állapotegyenleteket, valamint azokat Laplace-transzformálva kapjuk, hogy a meghibásodások közötti közepes idő:

$$\tau_1 = P_W(0) \cdot (-\mathbf{E})^{-1} g_k$$

a hibás állapotban tartózkodás átlagos ideje:

$$\tau_2 = P_F(0) \cdot (-\mathbf{M})^{-1} g_{N-k}$$

ahol g_x egy x hosszúságú oszlopvektor, melynek minden eleme 1.

3.4. A Markov-modell alapján nyert összefüggések

A megbízhatósági paraméterek számítására korábban elkészült programrendszer, a 3.3. fejezet egyenletei alapján kapható, következő összefüggéseket használja fel:

a) Várható működési idő (MUT):

$$MUT = \frac{\mathbf{P}_W \cdot g_K}{\mathbf{P}_W \cdot \mathbf{F} \cdot g_{N-K}}$$

b) Várható kiesési idő (MDT):

$$MDT = \frac{\mathbf{P}_F \cdot g_{N-K}}{\mathbf{P}_W \cdot E \cdot g_{N-K}}$$

c) Ciklusidő (MCT):

$$MCT = MUT + MDT$$

d) Első meghibásodás bekövetkezésének várható ideje ($MTFF$):

$$MTFF = \mathbf{F}_W(0) \cdot (-\mathbf{E})^{-1} \cdot g_K = \sum_{j=1}^{K} -e_{1j}^{-1}$$

Ha a rendszert az 1-es sorszámú állapottól indítjuk, tehát

$$\mathbf{P}_W(0) = (1, 0, 0, 0, \ldots 0)$$

e) A következő meghibásodás bekövetkezésének várható ideje ($MTTF$):

$$MTTF = \frac{\mathbf{P}_W \cdot (\mathbf{E})^{-1} \cdot g_K}{\mathbf{P}_W \cdot g_K}$$

Ezzel tehát a Markov-modell vázlatos ismertetésén túl, felsorolásra kerültek azok az összefüggések, melyeket az eddig elkészült programok alkalmaztak a megbízhatósági paraméterek analitikus előállításakor.

4. Az állapottér redukálásának okai és módjai

4.1. Gyakorlati tapasztalatok a megbízhatóságelméleti programcsomag alkalmazásával kapcsolatban

A megbízhatóság-elméleti programcsomag gyakorlati alkalmazása néhány új problémát vetett fel. A rendszer figyelembe vett elemei számának növelésével a lehetséges állapotok száma rohamosan nő. Így már 1024 állapot lehetséges 10 elem esetén is. Ez a felhasználói oldalról azt jelenti, hogy a gyakorlatban szükséges méretű rendszerek esetén a hatalmas mennyiségű adatot nem elég csak számítógépbe vinni, hanem az elvégzett számítások után a kapott információkat át kell látni és értékelni. Például az állapotvalószínűségek kiszámításakor a nagyszámú állapothoz rendelt számérték áttekinthetetlenné teszi az egyébként hasznos adatokat. A másik probléma a megbízhatósági paraméterek kiszámításakor ütközik ki. $MTFF$ és $MTTF$ számításakor egy-egy $K \times K$-s mátrixot kell invertálni. Ez nagy K értékek mellett tetemes futási időt eredményez a számítógépen. Az alkalmazott számítógép IBM 370/115, melynek működési sebessége messze elmarad a napjainkban lehetséges számítógép működési sebességektől. Új feladatot jelent tehát az állapotok csökkentése úgy, hogy közben hasznos információ ne vesszen el.

4.2. *Az állapottér redukálásának lehetséges módjai*

a) a meghibásodási időintervallumok közelítő ekvivalenciája:

MTFF és *MCT* közelítőleg egyenlő nagy megbízhatóságú rendszerek esetén. Így *MTFF* könnyen megkapható kevés állapotnál. Azt azonban nem tudjuk eldönteni, hogy az egyenlőség teljesítéséhez milyen megbízhatóság kell.

b) Rendszer-dekompozíció:

Tételezzük fel vizsgált rendszerünkről, hogy az, egymástól független alrendszerekből áll. Ha meghatározzuk a Markov aparátussal külön az alrendszerek lehetséges állapotait, akkor a teljes rendszer állapotait az alrendszerek állapotainak összes lehetséges kombinációja szolgáltatja. Tegyük fel, hogy a k. alrendszer n_k állapotot tartalmaz. A rendszer állapotainak száma $N=\prod_{k=1}^{r} n_k$, ahol r az alrendszerek száma. Mivel függetlenek az alrendszerek, így az alrendszer állapotok valószínűségeinek szorzata adja a rendszerállapotok valószínűségét. Az eddigiek alapján a rendszert akkor tekintjük jónak, ha az egyes alrendszerek jók. Ezen módszer előnyös volta azzal magyarázható, hogy r darab $n_1, n_2, \ldots n_r$ ismeretlenes egyenletrendszert kell megoldani. Végeredményben tehát 2^r állapotot vizsgálunk N állapot helyett.

c) Állapotösszevonás:

Egy állapotcsoport összevonható, ha az átmeneti gyakoriságok akármelyik más állapothoz, vagy összevont állapotok csoportjához ugyanaz minden állapot számára a csoportban, valamint az összevonandó állapotok a felhasználó, az eredményeket kiértékelő számára egyenértékűek. Például, hogy a rendszerről ne veszítsünk információt, alapkövetelmény, az összevont állapotok mindegyikének működőképes, vagy mindegyikének hibás állapotnak kell lennie. Az összevonást főleg akkor érdemes alkalmazni, ha a rendszer számos alrendszert tartalmaz.

4.3. *Mintapélda az állapotösszevonásra*

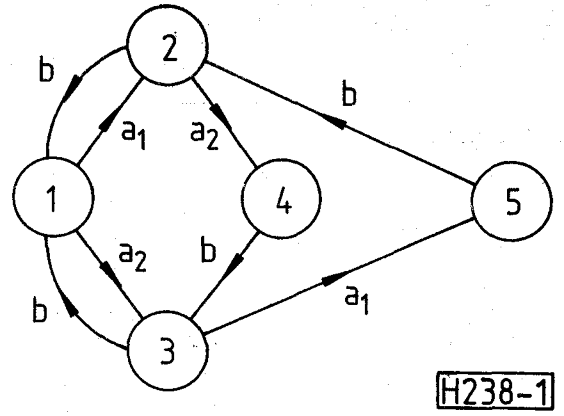
Vizsgáljunk meg egy kétegységes párhuzamos rendszert egy javító személlyel, aki a hibás egységeket exponenciális eloszlással, b paraméter szerint javítja (b átlépési gyakoriság), mely b megegyezik mindkét egységre. Az egységek meghibásodási tényezői a_1 és a_2. A szemléletes gráfstruktúrát az 1. ábrán láthatjuk. A gráf csúcsait körökkel jelöljük, a beleírt szám az állapot sorszámát adja. Az átmeneti gyakoriság mátrixa:

$$A=\begin{bmatrix} -a_1-a_2 & a_1 & a_2 & 0 & 0 \\ b & -a_2-b & 0 & a_2 & 0 \\ b & 0 & -a_1-b & 0 & a_1 \\ 0 & 0 & b & -b & 0 \\ 0 & b & 0 & 0 & -b \end{bmatrix}$$

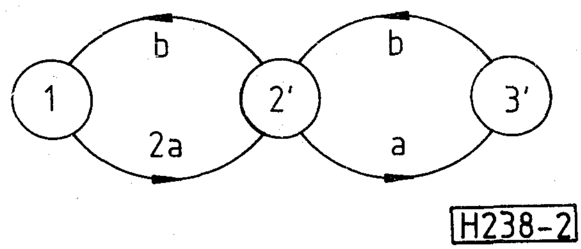
Ha az $a_1=a_2$, vagyis a meghibásodási tényezők megegyeznek mindkét egységre, akkor a 2-es és a 3-as, valamint a 4-es és 5-ös állapotok összevonhatók. Eredményül tehát a 2. ábrán látható gráfstruktúrát kapjuk, ahol a

2'. = 2. és 3.

3'. = 4. és 5. állapotok összevonva.



1. ábra. Kétegységes párhuzamos rendszer megbízhatósági modellje



2. ábra. Kétegységes párhuzamos rendszer, összevonás után

Ezen összevont állapotokat tartalmazó rendszer átlépési gyakoriság mátrixa:

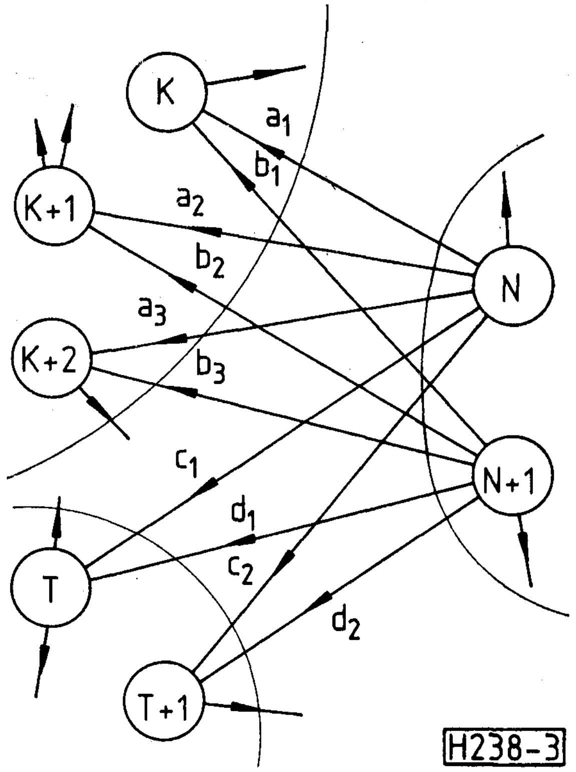
$$A'=\begin{bmatrix} -2a & 2a & 0 \\ b & -a-b & a \\ 0 & b & -b \end{bmatrix}$$

A példánál kimutatható a következő probléma:

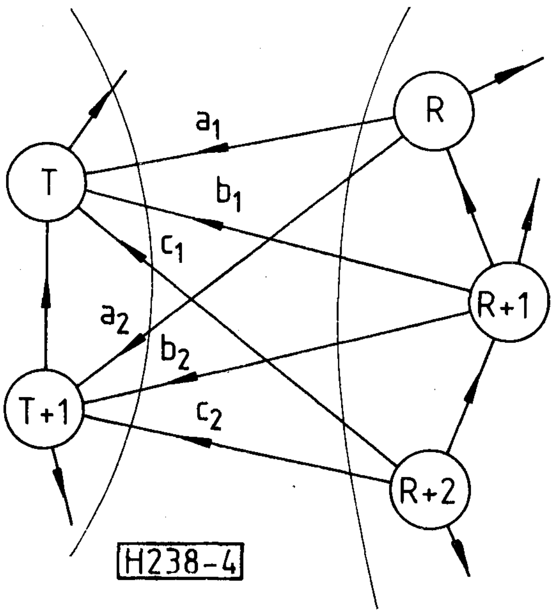
Az állapotösszevonásra adott kritériumot állapotpáronként sorban alkalmazva azt kapjuk, hogy a mintapélda 5 állapota közül egy sem vonható össze. A 2-es és 3-as állapot nem vonható össze mivel a 2-ből a 4-be, a 3-ból az 5-be van olyan átmenet ami nem azonos. A 4-es és 5-ös szintén nem vonható össze mivel különböző csúcsokhoz rendelkeznek átmenettel. A 2. és 3., valamint a 4. és 5. állapot mégis összevonható. A 2. és 3. állapot összevonható, ha a 4. és 5. összevonható, a 4. és 5. összevonható, ha a 2. és 3. összevonható. Mivel ez mindkét irányban igaz, ezért végezhetjük el az összevonást. Az állapotpáronkénti összevonást nevezzük feltétel nélküli, a más csoportok összevonhatóságára tett előzetes feltételezéssel végzett összevonhatósági vizsgálatot pedig feltételes összevonásnak a továbbiakban.

Mindkét esetet vizsgáljuk meg általánosan!

Feltétel nélküli esetben vizsgáljuk a 3. ábrán szemléltetett részrendszert: Keretezzük be a felhasználó számára egyenértékű állapotokat. Az összevonhatóság nem kizárt, ha



3. ábra. Részrendszer, a feltétel nélküli összevonás vizsgálatához



4. ábra. Részrendszer, a feltételes összevonás vizsgálatához

$$a_1 = b_1$$
$$a_2 = b_2$$
$$a_3 = b_3$$
$$c_1 = d_1$$
$$c_2 = d_2$$

A feltételeknek azonban a többi állapot felé is teljesülniük kell.

Feltételes esetben a 4. ábrán szemléltetett részrendszert vizsgáljuk. Feltételezzük, hogy T és $T+1$ összevonható, ekkor R, $R+1$, $R+2$ összevonhatóságának szükséges feltétele:

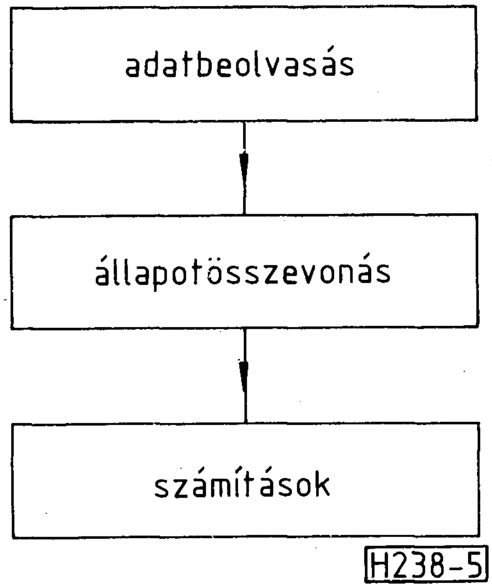
$$a_1 + a_2 = b_1 + b_2 = c_1 + c_2.$$

Ha azonban a későbbiekben kiderül, hogy T és $T+1$ nem összevonható, akkor R, $R+1$, $R+2$ összevonhatóságának szükséges feltétele:

$$a_1 = b_1 = c_1$$
$$a_2 = b_2 = c_2.$$

5. Számítógép-program az állapotösszevonások elvégzésére

Az elkészített programcsomag feladata az, hogy az állapottér szűkítését az állapotösszevonás útján végezze el. Felismerje az egymással összevonható állapotokat és egy olyan szűkebb állapotteret hozzon létre, melyben már a megváltozott, összevonásnak megfelelő átlépési gyakoriságok szerepelnek. Szükséges még az is, hogy a program információt szolgáltasson arról, a régi struktúra, mely állapotai lettek összevonva az új állapottér állapotaiban. Az összevonás feltételeinek matematikai megfogalmazása [2]-ben, valamint általánosan a 4.3. pontban történt meg. A követelmények tehát matematikailag viszonylag egyszerűen megadhatók. Komoly nehézséget jelent viszont az adott tulajdonsággal rendelkező állapotkombinációk megkeresése az állapottérben, különösen feltételes esetben. Ekkor ugyanis az állapottér N számú állapotából ki kell választani K számút és a maradék $N-K$-ból Q-t. Feltételezve a K állapot összevonhatóságát meg kell vizsgálni a kiválasztott Q összevonhatóságát és viszont. Amennyiben az összes lehetséges kombináció-kombináció párt végigvizsgáljuk, úgy a számítási idő összemérhető lesz a 4.1.-ben említett mátrix-invertálás számítási idejével. Az általam készített algoritmus megtalálja a feltéte-



5. ábra. Az „állapotösszevonás" helye a programban lesen összevonható állapotokat az összes lehetséges kombináció-kombináció pár vizsgálata nélkül, így gyorsan eredményre vezet.

5.1. *Az összevonást végző program helye a korábban készült programrendszerben*

Az állapotösszevonás módszerének választását az indokolta, hogy korábban más módon már készült program az állapotszám csökkentésére. Az állapotösszevonást végző programblokk elhelyezkedése a korábban készült, megbízhatósági paramétereket számító programrendszerben az 5. ábrán látható. A felhasználó szabadon választhat, igényli-e az állapotösszevonást vagy sem. A soros rendszerből következik, az állapotösszevonást végző résznek olyannak kell lennie, hogy a ,,számítások" nevű blokk felé ugyanúgy szolgáltasson információkat, mint az ,,adatbeolvasás" nevű blokk. Ez azonos adatstruktúrát és azonos kimenő állapotneveket követel meg az ,,adatbeolvasás" és az ,,állapotösszevonás" programblokkoktól. Mivel a ,,számítások" rész működését nem befolyásolja az összevonás elvégzése vagy el nem végzése, ezért elég, ha az ,,állapotösszevonás" közli a felhasználóval az összevonás megtörténtét vagy meg nem történtét, valamint a régi és az új állapotsorszámok egymáshoz rendelését.

5.2. *A ,,szint" fogalma és az ,,átlépés" struktúra*

Az 5.1. fejezetben ismertetett okok miatt az állapotösszevonást végző program átveszi az ,,adatbeolvasás" és ,,számítások" blokkok adatábrázolási struktúráját, ezért szükséges az ,,állapotösszevonás" megismeréséhez a címben említett két fogalom ismertetése.

Szint:

Ha egy berendezést megbízhatósági szempontból leíró állapottérben egyes állapotok bizonyos szempontból egyenértékűek, akkor úgy mondjuk, hogy egy szintre kerülnek. Ezt az egyenértékűséget a felhasználó dönti el, ezért a felhasználónak a programcsomag használata során meg kell adnia, hogy mely állapotokat kezeljen a program egy szinten levőknek. Például egy távközlő berendezés 30 csatornája közül 6 db rossz. Az egyébként egyforma csatornák mellett (azonos megbízhatósági paraméterű) teljesen mindegy, hogy melyik 6 rossz. Tehát a pontosan 6 hibás egységet reprezentáló állapotok egy szintre kerülnek. A szintekhez egy *SZINT* vektort rendelhetünk, ezen vektor X. eleme azt adja meg, hogy a X. állapot hányas sorszámú szinten található.

Átlépés-struktúra:

A számítások elvégzéséhez szükséges megadni a programnak az átlépési gyakoriság **A**-mátrixának az A_{ij} elemeit. Ez a mátrix azonban sok értéktelen nulla elemet tartalmaz, ezért ezeket felesleges lenne bemenő adat formájában közölni. Emiatt a bemenő adatokkal egy tömör úgynevezett ÁTLÉPÉS struktúrát töltünk fel, melybe csak a nullától különböző átlépési gyakoriságú tényleges átlépések kezdő és végállapota, valamint paraméterei kerülnek.

5.3. *A program általános leírása*

A program a folytonos idejű Markov-láncokkal leírható rendszerek megbízhatósági analízisére készült programcsomaghoz illeszkedik, ezért annak programnyelvén, PL/1 programnyelven íródott. Feladata, hogy egy állapottérben felismerje a markovitást nem befolyásoló lehetséges állapotösszevonásokat és elvégezze azokat. Mivel a feladat külön önálló részfeladatokra bontható, ennek megfelelően a program szubrutinokra tagolódik. A program a homogén, folytonos idejű, diszkrét állapotterű, ergodikus Markov-lánccal leírható rendszerek esetén alkalmazható. Futásához az ,,adatbeolvasás" adatai szükségesek. Az eredményeket továbbadja a ,,számítások" szubrutinnak, a régi és új állapotok egymáshoz rendelését pedig táblázatos formában kiíratja. Szintenként vizsgálja meg az összevonhatóságot, mivel összevonást csak itt engedhetünk meg. A vizsgálatokat, a 4.3. pontban tárgyalt példánál látott probléma miatt két úton végzi el:

— feltétel nélküli vizsgálattal, vagyis állapotpáronként sorban haladva,

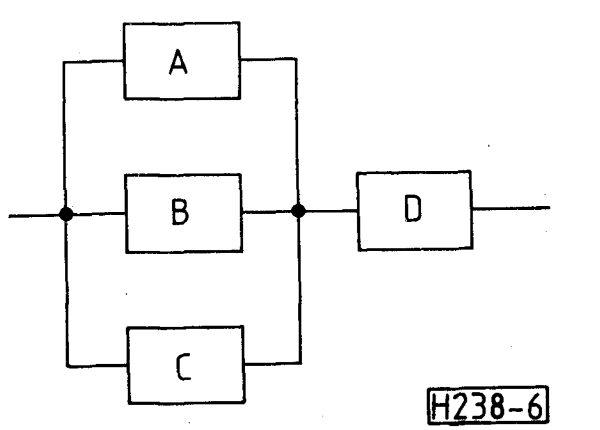
— előzetesen bizonyos csoportok összevonhatóságát feltételezve, vagyis feltételes összevonhatósági vizsgálattal.

6. A programrendszer alkalmazása

A továbbiakban a programcsomag felhasználói leírását mellőzve megismerünk egy mintapéldát. A felhasználói leírást [9], az egyéb szükséges ismereteket [5] tartalmazza. A futtatási eredmények és táblázatok [9] mellékletében részletesen tanulmányozhatók.

6.1. *Mintapélda*

Vizsgáljunk egy háromcsatornás rendszert, melyhez 1 db közös egység csatlakozik, ld. 6. ábra. A csatornák A, B, C, a közös egység jele legyen D. Tekintsük a rendszer teljesen hibátlan állapotának az 1-es állapotot. Ekkor A, B, C és D jó. A rendszer működésképtelen ha A és B és C vagy D rossz. A rendszer állapotainak száma $2^4=16$.



6. ábra. Háromcsatornás rendszer modellje

az új állapot		a régi állapotszám
sor-száma	szint-száma	
1	1	1
2	2	2, 3, 4
3	3	5, 6, 7
4	4	8, 9, 10, 11, 12, 13, 14, 15, 16

H238-7

7. *ábra.* Az állapotösszevonás eredménye

MTFF	=	$3{,}3333 \cdot 10^4$
MTTF	=	$3{,}3333 \cdot 10^4$
MCT	=	$3{,}3335 \cdot 10^4$
MUT	=	$3{,}3333 \cdot 10^4$
MDT	=	2,0008

H238-8

8. *ábra.* A számítás eredménye

A szintek száma 4 és ezek a következő állapotokat tartalmazzák:

— A rendszer hibátlan, 1. szint 1. állapot
— A rendszer 1 db hibás csatornát tartalmaz, 2. szint 2., 3., 4. állapot
— A rendszer 2 db hibás csatornát tartalmaz, 3. szint 5., 6., 7. állapot
— A rendszer működésképtelen, 4. szint 8., 9., 10., 11., 12., 13., 14., 15., 16. állapot.

Legyen A, B és C meghibásodási gyakorisága $a=2\cdot 10^{-5}$/h, D meghibásodási gyakorisága $d=3\cdot 10^{-5}$/h.

A javítás történjen:

— a 2. szintről az 1-re $\mu_1=0{,}05$/h-val,
— a 3. szintről a 2-ra $\mu_2=0{,}2$/h-val,
— a 4. szintről az 1-re $\mu_3=0{,}5$/h-val.

A jó állapotok száma 7.
A számítógéppel történt összevonás után a rendszer új állapotainak szinthez, valamint a régi állapotokhoz való hozzárendelése a 7., a számítási eredmények a 8. ábrán láthatók.

6.2. *A futtatások értékelése*

A példák futtatása után az értékelés három szempont szerint történt:

a) A számítógép üzeneteiből megállapítható volt, hogy az egyes szubrutinok és ezeken belül az egyes utasítások hányszor futottak. Kis állapotszám esetén így igazolható volt a szubrutinok helyes működése.
b) Számos feladat megoldása történt meg mind az összevonó szubrutin hívásával, mind anélkül. A kétféle úton eredményül kapott megbízhatósági paraméterek egyezése bizonyította az összevonást végző programrendszer helyes működését.
c) Az egyes példák futtatása során az összevonó szubrutin hívásakor (összevonás elvégzésekor) lényegesen kisebb futási idők adódtak, mint anélkül. Az ismertetett mintapélda esetén összevonással 26,68 sec, anélkül 60,37 sec CPU idő adódott. Nagyobb rendszereknél az összevonásra fordított idő egyre inkább elhanyagolható ahhoz az időhöz képest, amennyit nyerünk azzal, hogy a korábbinál kisebb állapotszámmal kell számolnunk (már a mintapéldában is 7×7-es mátrix helyett csak 3×3-ast kellett invertálni). Így a gyakorlatban vizsgált rendszerek nagy számú állapota esetén még fokozottabb aránybeli eltérések adódnak a kétféle futás időigénye között.

A programrendszer tehát gazdaságosan alkalmazható, mivel lerövidíti a megbízhatósági vizsgálathoz szükséges gépidőt, ezzel lehetővé téve a számítógép más feladatra való felhasználását.

6.3. *Alkalmazási és továbbfejlesztési javaslatok*

Az összevonást végző programcsomag egyenlőre a megbízhatósági paraméterek analitikus előállításánál került felhasználásra, de alkalmas szimulációs eljárással való együttműködésre is. Lehetőség van tehát arra, hogy szimulációs eljárások előtt is hívásra kerüljön az összevonó programrendszer.

Az állapottér redukálására korábban felsorolt módszerek közül a rendszer dekompozíciós eljárás eddig nem került felhasználásra. További feladatot jelenthet tehát ezen eljárás realizálása és a különböző eljárások futási eredményeinek egymással való összehasonlítása.

7. Köszönetnyilvánítás

A szerző köszönetét fejezi ki dr. Simonyi Ernő tudományos főosztályvezetőnek, a műszaki tudományok kandidátusának értékes tanácsaiért és támogatásáért, valamint Ladvánszky János aspiránsnak a kézirat átnézéséért.

IRODALOM

[1] *Dr. Almássy György:* Elektronikus készülékek szerkesztése. Műszaki Könyvkiadó, 1979.
[2] *John A. Buzacott:* Markov Approaeh the Finding

Failure Times of Repairable Systems. IEEE TRANSACTIONS ON RELIABILITI, VOL. R—19, NO. 4. NOVEMBER 1970, pp. 128—134.
[3] *Farkas Gy.:* Elektronikus berendezések konstrukciója I. Egyetemi kézirat, 1979.
[4] Dr. *Izsák M.:* Konstrukciós számítások. Egyetemi kézirat, 1979.
[5] Dokument, a BME/HEI IBM 370/115 számítóközpont kiadványa. 1981.
[6] *Jereb L., Pongor Gy.:* Programrendszer kidolgozása összetett struktúrájú javított rendszerek megbízhatóságának analízisére ós tervezésére batcli ós interaktív üzemmódban. BME/HEI. 1980.
[7] *Szabó I.:* Bonyolult rendszerek megbízhatósága. Diplomaterv, BME/HEI, 1980.
[8] *Felföldi* T.: Tömegkiszolgálási rendszerek vizsgálatára alkalmas programcsomag továbbfejlesztése. Diplomaterv, BME/HEI. 1981.
[9] *Goldschmidt L.:* Markov-láncok állapotterének csökkentése állapotösszevonással. Diplomaterv, BME/HEI. 1982.