

## 2. gyakorlat – Állapot alapú modellezés – Megoldások

**Figyelem:** Jelen anyag belső használatra készült megoldási útmutató, melyet a ZH felkészülés segítése érdekében publikáltunk. A feladatok részletesebb megoldása magyarázattal gyakorlaton hangzott el.

### 1. feladat

Közlekedési lámpát vezérlő elektronikát tervezünk.

- Készítsd el egy egyszerű háromfényű piros-sárga-zöld közlekedési lámpa olyan állapotterét, amely kellően finom ahhoz, hogy a lámpák vezérlését ez alapján lehessen végezni! Győződj meg arról, hogy az állapottér kizárólagos és teljes!
- 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?
- Mik az érvényes állapotátmeneti szabályok? Készítsd el az állapotgráfot!
- 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!
- 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!

### Megoldás

- $\{ \text{piros, sárga, zöld, piros-sárga, villogó sárga, kikapcsolt} \}$ , röviden:  $S = \{p, s, z, ps, \bar{s}, \circ\}$ . Ez kizárólagos és teljes. A kezdeti állapot:  $p$ .
- $S_p = \{p, \circ\}$ ,  $S_s = \{s, \bar{s}, \circ\}$ ,  $S_z = \{z, \circ\}$ .

“Milyen absztrakciós viszony áll fent a lámpa és az egyes égők állapottere közt?” Az absztrakciós viszony a *vetítés*. Egy komponens állapottere mindig állapottere a teljes rendszer állapottérének (hiszen a többi komponens állapota “elhagyható”, azaz összevonható egy állapotba).

Direkt szorzat:  $S_p \times S_s \times S_z$ ,  $2 \times 3 \times 2 = 12$  elemű lesz. A direkt szorzat elemei háromtagú vektorok ( $n$ -esek, tuple), pl.:  $\langle p, s, \circ \rangle$ , amit most  $ps$  formában fogunk rövidíteni.

Ebből természetesen nem minden állapot fordul elő:  $S \subset S_p \times S_s \times S_z$

A  $S_p \times S_s \times S_z$  Descartes-szorzat táblázatban:

		$S_s$		
$S_p$	$S_z$	$s$	$\bar{s}$	$\circ$
$p$	$z$	$psz$	$p\bar{s}z$	$pz$
	$\circ$	$ps^*$	$p\bar{s}^*$	$p^*$
$\circ$	$z$	$sz$	$\bar{s}z$	$z^*$
	$\circ$	$s^*$	$\bar{s}^*$	$\circ^*$

Az  $*$ -gal állapotok az eredeti állapottérben is megjelennek.

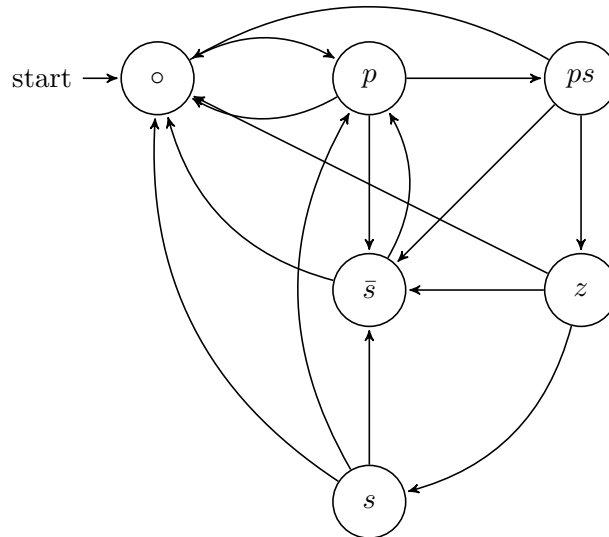
- A cél az érvényes állapotátmeneti szabályok megállapítása a hat állapot között. Sok tervezői döntéssel szembesülünk, ezekre egy-egy választ adtunk zárójelben:
  - Kötelezően pirosnak kell lennie bekapcsolás után? (igen)
  - Csak a szabályos működést modellezzük? (igen)
  - Ha elmegy az áram, az hibás működés? (igen, vegyük úgy, hogy csak tervezett áramszünetek vannak)

- Pirosból pirosba mehet? (nem rajzoljuk fel, mert semmi megfigyelhető nem történik)
- Piros-sárgából mehet rögtön pirosba, pl. baleset esetén? (nem)

A KRESZ könyvben található állapotgép-jellegű ábra, de csak a piros, piros-sárga, zöld és sárga állapotokat tartalmazza.

Az állapotgép bárhonnan kerülhet kikapcsolt ( $\circ$ ), ill. villogó ( $\bar{s}$ ) állapotba (utóbbi esetén kivéve kikapcsolt állapotból).

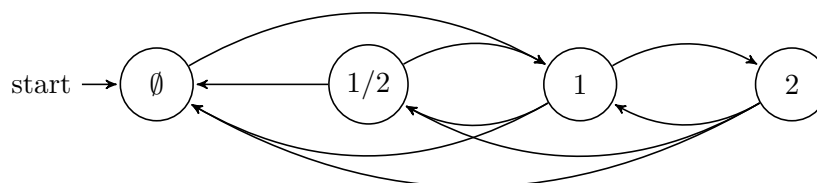
Egy lehetséges megoldás:



- d) Bontsuk fel a piros állapotot két állapotra: piros és a gyalogosoknak folyamatos zöld ( $p_{fz}$ ), piros és a gyalogosoknak villogó zöld ( $p_{vz}$ ). Az állapotgráfra is vigyük át a változtatást: a két állapot között  $p_{fz} \rightarrow p_{vz}$  átmenet van, a többi értelemszerűen berajzolandó.

Itt tovább lehet gondolni azt, hogy szükség van-e olyan állapotra, amikor az autósoknak és a gyalogosoknak is piros a lámpa. Ha az előző részben úgy döntöttünk, hogy a lámpának kötelezően pirosnak kell lennie bekapcsolás után, akkor itt is érdemes ezt megvalósítani: ebben az esetben a piros állapotot három állapotra kell felbontani:  $p_p, p_{fz}, p_{vz}$ .

- e) 4-állapotú állapottér, fogyasztások: 0, 1/2, 1, 2 (egyszerre mindhárom nem világíthat). Az 1-es állapotra rajzolhatnánk hurokélet (pl. zöld  $\rightarrow$  sárga  $\rightarrow$  piros esetén mindig csak egy lámpa világíthat), de ezt nem tesszük meg, mivel nem figyelhető meg kívülről. Többszörös éleket (ha nincs rajtuk különböző bemenet/kimenet) nem rajzolunk be.



## 2. feladat

A háromrétegű architektúra felépítése:

réteg	komponens
megjelenítési réteg	webszerver
üzleti logikai réteg	alkalmazáserver
adatelérési réteg	adatbázisszerver

Háromrétegű kiszolgáló infrastruktúránk viselkedésének modellezésére megfelelő állapotterek-e a következők:

- $\{ \textit{Webszerver}, \textit{Alkalmazáserver}, \textit{Adatbázisszerver} \}$
- $\{ \textit{Webszerver dolgozik}, \textit{Alkalmazáserver dolgozik}, \textit{Adatbázisszerver dolgozik} \}$
- $\{ \textit{Leállítva}, \textit{Tétlen üzemel}, \textit{Aktívan dolgozik} \}$
- $\mathbb{N}$  (mint a pillanatnyilag feldolgozás alatt álló kérések száma)
- $\{ \textit{A kérés feldolgozása még nem kezdődött el}, \textit{A szerverek épp dolgoznak a kéréssel}, \textit{A kérés kiszolgálása befejeződött} \}$

### Megoldás

Fogalmak:

- **Adatbázisszerver:** hosszútávon tároljuk az adatokat
- **Alkalmazáserver:** az üzleti logikáért felelős alkalmazást futtatja
- **Webszerver:** megjelenítést felelős, generálja a HTML oldalakat

Válaszok:

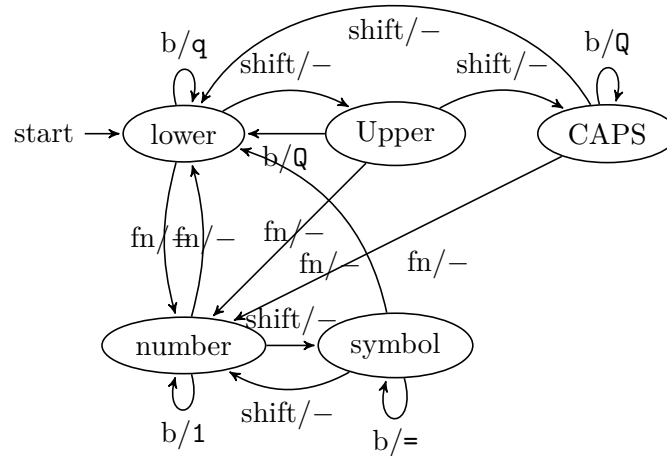
- Nem. A három közül *pontosan egynek* kell igaznak lennie minden időpillanatban.
- Nem. Egyszerre több gép is dolgozhat.
- Igen. Természetesen elképzelhetők további állapotok is, pl. *hibernált*.
- Igen, ez egy végtelen állapotteret eredményez. A kizárólagosság mellett fontos vizsgálni, hogy teljes-e. Pl. 3,5 vagy  $-9$  kérés nem lehet a rendszerben, ezért teljes. Ezzel persze a kérések eloszlását nem tudjuk megnézni.
- Igen, ha egyszerre egy kérés lehet a rendszerben. Ezzel azonban a modellünk nem lesz alkalmas terhelések vizsgálatára.

## 3. feladat

Modellezzük állapotgéppel egy mobiltelefon érintőképernyőjére tervezett virtuális billentyűzetet! A billentyűzetben 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 begépett karaktereket mint outputot!

### Megoldás

Fontos, hogy csak egy billentyűt kell feltüntetnünk az állapotgépen, ez segíti az áttekinthetőséget is. Egy lehetséges megoldás az alábbi:



A jelenlegi megoldásnál a *number* és a *symbol* állapotokból az *fn* megnyomására mindig a *lower* állapotba kerül az automata. A valóságban a fejlettebb implementációk képesek megjegyezni azt, hogy a *lower*, *Upper* és *CAPS* állapotok közül melyikben voltak és abba lépnek vissza.

Ez az állapotgépben úgy valósítható meg, hogy 1) finomítjuk a *number* és a *symbol* állapotokat (*numbers with lower*, *numbers with Upper*, *numbers with CAPS*) helyett 2) history state-et vezetünk be (erről a *A programozás alapjai 3* tárgyban lesz részletesen szó).

## 4. feladat

Egy út mentén 10 jelzőlámpa található, egyenként 4 állapottal. Legfeljebb hány állapota lehet a teljes rendszernek?

### Megoldás

$$4^{10} = 2^{20} = (2^{10})^2 \approx 10^6.$$

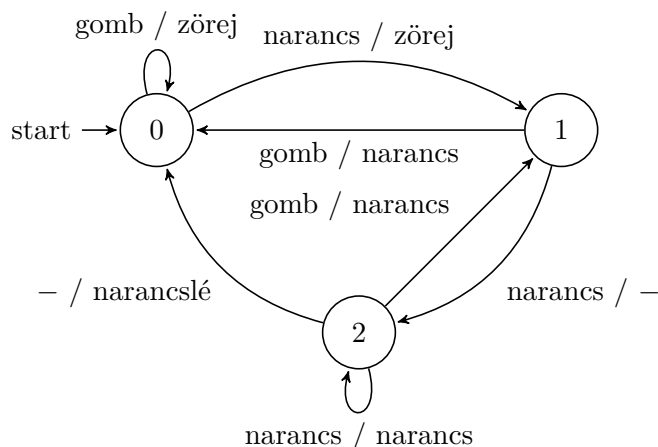
Természetesen nem biztos, hogy minden állapot elérhető is lesz, hiszen a jelzőlámpák állapotátmenetei nem biztos, hogy függetlenek egymástól.

## 5. feladat

Figyelem: a  $-$  jel a bemeneti jel pozíciójában spontán állapotátmenetet jelöl, a kimenet helyén pedig kimenet hiányát, egyik esetben sem *don't care* szimbólum.

- Tekintsük az  $M$  állapotgépet! Determinisztikus-e a viselkedésmodell? Hozzávehető-e, ill. elhagyható-e egyetlen állapotátmeneti szabály, hogy ez megváltozzon?
- Absztraháljuk az  $M$  állapotgráfot a  $\{ \{0\}, \{1, 2\} \}$  állapotpartíció szerint! A nemdeterminizmus milyen formái figyelhetők meg az eredményként kapott állapotgráfon?

$M$  állapotgép:

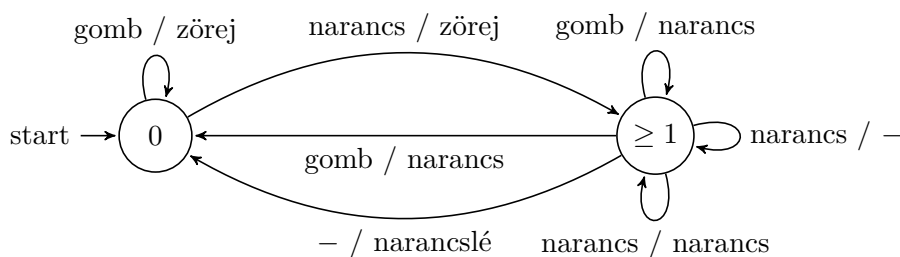


## Megoldás

A állapotgép egy narancsprést modellez. A rendszer állapotai az adott pillanatban a narancsprésben található narancsok darabszáma. A *gomb* egy bedobott narancsot ad vissza, ha a gépben van narancs, egyébként *zörejt* okoz. Az első narancs bedobása szintén *zörejt* vált ki (a többi narancs narancsra esik). A rendszer nem engedi, hogy kettőnél több narancs legyen a gépben. Amennyiben két narancs van a gépben. Egy *spontán állapotátmenet* kimeneteként előáll a narancslé.

A *zörej* kiemeneti jel finomításával (*tokenfinomítás*) meg lehetne különböztetni a *gomb* és a *narancs* által kiváltott *zörejt* (pl *kattogás* és *puffanás*), ezzel tovább konkretizálva a modellt.

- Ez úgy dönthető el, ha végignézzük minden állapotot és a kimenő élekre ellenőrizzük, hogy minden input esetén csak egy él megy-e ki. Ennek *majdnem* megfelel az állapotgépünk, azonban van olyan állapotátmenet, amelynél  $-$  karakter van a bemeneten (tehát bármely input helyett választható ez), ezért az állapotgép *nem determinisztikus*.
- A megoldást az alábbi állapotgráf szemlélteti:



Az állapotgép az 1 és 2 állapotok összevonásával keletkezik.

A kapott állapotgép az eredeti állapotgép egy absztrakciója, mely az eredeti állapotgép végrehajtássorozatain kívül számos más végrehajtássorozatot is megenged. Ezek közül néhány akár valós rendszer viselkedése is lehet: az absztrakt rendszer a közös absztrakciója az  $n$  narancsból ( $n > 1$ ) narancslevet készítő narancspréseknek. Az absztrakt állapotgép ugyanakkor olyan végrehajtássorozatokat is megenged, amelyeket nem egy valós narancsprés viselkedését modellezzik.

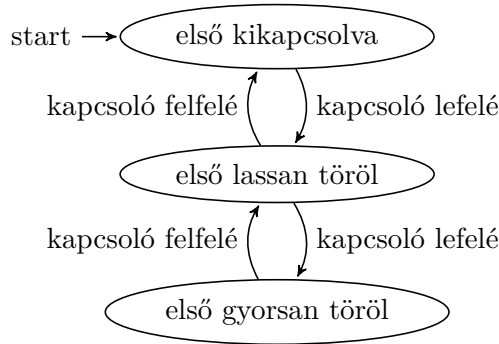
Az absztrakt állapotgépben a spontán állapotátmeneten kívül is jelenik meg nemdeterminizmus:  $\geq 1$  állapotban a *gomb*, illetve *narancs* bemeneti események több tranzíciót is kiválthatnak. Az eredeti nemdeterminizmus abból fakadt, hogy maga a modellezett rendszer volt nemdeterminisztikus működésű. A most újonnan bevezetett nemdeterminizmus forrása az absztrakció: elhanyagoljuk azt a tudást, amely alapján determinisztikusan dönteni lehetne (ez jó is lehet, pl. ha sok rendszerről egyszerre akarunk beszélni).

## 6. feladat

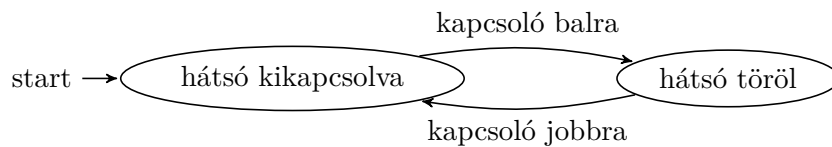
Egy autóban az első ablaktörlőnek három állapota van (*első kikapcsolva*, *első lassan töröl*, *első gyorsan töröl*), a hátsó ablaktörlőnek kettő (*hátsó kikapcsolva*, *hátsó töröl*). Az első ablaktörlő működését az  $M_1$  állapotgép, a hátsóét az  $M_2$  állapotgép modellezi.

Készítsd el az  $M_1$  és  $M_2$  állapotgépek aszinkron szorzatát!

$M_1$  állapotgép:



$M_2$  állapotgép:



## Megoldás

Az aszinkron szorzat:

