Házi feladat (Modellellenőrzés)

A feladat címe: **Dijkstra algoritmusa**

Konzulense: **Tóth Tamás**

A probléma leírása

Dijkstra algoritmusa [1] egy megosztott memória segítségével kölcsönös kizárást megvalósító protokoll, mely lehetővé teszi *N* folyamat számára egy globális erőforráshoz való konfliktusmentes hozzáférést.

Globális változók:

```
// b and c are arrays of Booleans
// each element is initialized to false
bool b[1..N] := { false };
bool c[1..N] := { false };

// k is an integer between 1 and N
[1..N] k := 0;
```

Az *i*-edik folyamat pszeudokódja:

```
// Local variables

// j is an integer between 1 and N + 1
[1..N + 1] j := 0;

// Entry protocol
b[i] := true;
L:
if (k ≠ i) {
    c[i] := false;
    if (¬b[k]) {
        k := i;
    }
    goto L;
} else {
    c[i] := true;
    for j in [1..N] {
        if (j ≠ i ∧ c[j]) {
            goto L;
        }
    }
}

// Critical section
```

```
// Exit protocol
c[i] := false;
b[i] := false;
```

Készítse el a protokoll modelljét az alábbi elvárások figyelembe vételével!

- A modellezésnél egy értékadás (pl. `b[i] := true`) végrehajtása, illetve egy atomi formula (pl. $k \neq i$) kiértékelése tekinthető atomi műveletnek. Az ennél összetettebb műveletek végrehajtása (pl. $j \neq i \wedge c[j]$ kiértékelése) nem atomi.
- A folyamatok ütemezéséről egy ütemező gondoskodik, amelyet szintén modellezni kell. Az ütemező minden időegységben léptet egy folyamatot. Az ütemező működésében minden *M*-edik időegységben teljesülnie kell, hogy az utolsó M időegységben minden folyamat legalább *K*-szor, de legfeljebb *L*-szer lépett (tehát $L \cdot N \geq M$ és $M \geq K \cdot N$). Ezt a kényszert leszámítva az ütemező a léptetendő folyamatot nemdeterminisztikusan választja.
- A modell legyen *K*-ban, *L*-ben, *M*-ben és *N*-ben paraméterezhető.

Az ellenőrzendő követelmények

Temporális logikai kifejezések és modellellenőrzés segítségével igazolja az alábbi követelmények teljesülését egy *legalább három folyamatot* tartalmazó rendszeren! A követelmények nem teljesülése esetén ellenpélda segítségével magyarázza meg, miért nem teljesül az adott követelmény.

1. A rendszer holtpontmentes.
2. A kölcsönös kizárás megvalósul, azaz egyszerre legfeljebb egy folyamat tartózkodhat a kritikus szakaszban.
3. A kritikus szakasz elérhető valamelyik folyamat számára.
4. A rendszer éhezésmentes, azaz minden folyamat előbb-utóbb eljut a kritikus szakaszba.

Tippek

Az ütemező modellezéséhez gondolja meg a *select* konstrukció használatát!

Az ütemező nélküli modell is beadható a hozzá kapcsolódó ellenőrzésekkel, de az nem tekinthető teljes megoldásnak.

Hivatkozások

[1] E. W. Dijkstra: Solution of a problem in concurrent programming control.
Communications of the ACM, Vol 8 Issue 9, pp. 569, 1965.