Gyakorló feladatok a zárthelyi dolgozathoz (GTK levelező képzés)

1. Követelmények formalizálása temporális logika alkalmazásával

Egy vasúti kereszteződést biztosító fénysorompó viselkedését az állapotaihoz rendelt következő atomi kijelentésekkel jellemezzük: *{kikapcsolt, fehér, piros}*

A kereszteződéshez érkező autós viselkedését az állapotaihoz rendelt következő atomi kijelentésekkel jellemezzük: *{érkezik, körülnéz, megáll, áthalad}*

Formalizálja LTL kifejezések segítségével az alábbi követelményeket, amelyek az autós viselkedésére minden esetben vonatkoznak:

- Kikapcsolt állapotú fénysorompó esetén az autós körülnéz és a következő időpillanatban vagy áthalad, vagy megáll.

  $G(kikapcsolt \Rightarrow (körülnéz \wedge X\,(áthalad \vee megáll)))$

- Az autós előbb-utóbb át fog haladni a vasúti kereszteződésen.

  $GF(áthalad)$

- Ha egy autós érkezésekor piros a lámpa, akkor az autós addig nem halad át, amíg fehérre nem vált a fénysorompó.

  $G((piros \wedge érkezik) \Rightarrow ((\neg áthalad)\; U\; fehér))$

2. Követelmények formalizálása temporális logika alkalmazásával

Egy bonyolult szimulációt futtató szerver állapotait a következő atomi kijelentésekkel jellemezzük: *{kikapcsolt, várakozó, bemelegítés, szimuláció}*

A szerverszoba hűtőberendezésének működését az állapotaihoz rendelt következő atomi kijelentésekkel jellemezzük: *{készenlét, normál, maximális}*

Formalizálja LTL kifejezések segítségével az alábbi követelményeket, amelyek a rendszer működésére minden esetben vonatkoznak:

- Ha egy adott pillanatban a szimuláció a hűtőberendezés készenléti állapota mellett zajlik, akkor a következő pillanatban a szerver várakozó állapotra kapcsol.

  $G((szimuláció \wedge készenlét) \Rightarrow X\, várakozó)$

- Előbb-utóbb elkezdhető a szimuláció.

  $GF(szimuláció)$

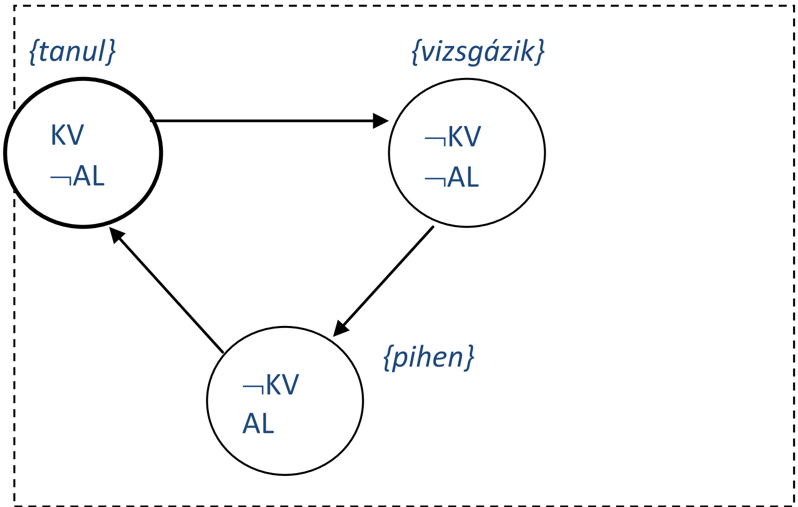
- Csak úgy hajtható végre szimuláció, ha volt bemelegítés a hűtőberendezés normál működése mellett.

  $G((X\, szimuláció) \Rightarrow (bemelegítés \wedge normál))$

3. Modellezés és modellellenőrzés

Egy informatikus hallgató „állapotait” aszerint különböztetjük meg, hogy éppen kávézik vagy nem, valamint éppen alszik vagy nem. A hallgató három tevékenységét különböztetjük meg: tanulás közben kávézik és nem alszik; ezután vizsgázik, ahol nem kávézik és nem is alszik; a vizsgázás után pihen, ekkor alszik és nem kávézik. A hallgató alapállapota a tanulás, amit a vizsgázásig nem is hagy abba. Tanulás nélkül a hallgató nem vizsgázik; a vizsgázás után közvetlenül pedig nem tanul (csak pihenés után).

- Rajzolja fel a hallgató itt leírt viselkedését modellező Kripke-struktúrát a hallgató kávézását és alvását figyelembe véve! Az egyes állapotokat jellemezze a következő atomi kijelentésekkel: *{pihen, tanul, vizsgázik}*



(Az állapotokba csak a jobb érthetőség kedvéért kerültek a kávézásra és az alvásra utaló megjegyzések, ezek nem részei a Kripke-struktúrának.)

- Ellenőrizze a modellen, hogy a hallgató alapállapotából (ami a tanulás) kiindulva teljesül-e a következő CTL kifejezés: *E(¬vizsgázik U pihen)*! Válaszát indokolja meg!

A *tanul* címkéjű állapotból létezik-e olyan útvonal, amelyen nem szerepel a *vizsgázik* címke, amíg el nem érjük a *pihen* címkéjű állapotot?

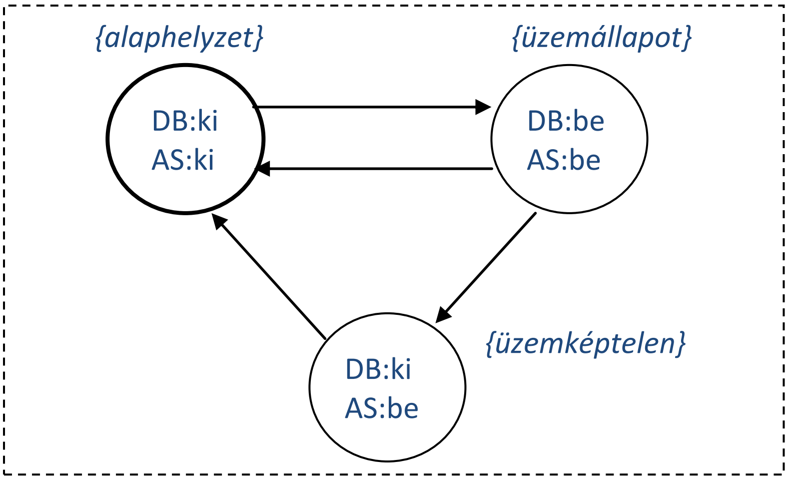
Ilyen útvonal nincs, mert a *tanul* címkéjű állapoton ugyan nem szerepel a *vizsgázik* címke, viszont még a *pihen* címke sem, így tovább kell lépni; a rákövetkező állapot csak *vizsgázik* címkéjű állapot lehet, amin nincs *pihen* címke.

A bizonyítás elvégezhető az iteratív állapotcímkézési eljárással is.

4. Modellezés és modellellenőrzés

Egy informatikai rendszer két szerverből áll, ezek egy adatbázisszerver és egy alkalmazásszerver, amelyek kikapcsolt vagy bekapcsolt állapotban lehetnek. A szervereket hibamentes esetben egyszerre kapcsolják ki és be. Alaphelyzetben mindkét szerver ki van kapcsolva. A megfelelő üzemállapot az, amikor mindkét szerver be van kapcsolva. Ha az üzemállapotban az adatbázisszervert hiba következtében kikapcsolják, az rendszer szinten üzemképtelen állapotnak tekinthető. Ezután az alkalmazásszervert is kikapcsolják, majd mindkét szerver bekapcsolásával indítják újra a rendszert.

- Rajzolja fel a rendszer itt leírt működését modellező Kripke-struktúrát az egyes szerverek bekapcsolását és kikapcsolását figyelembe véve! Az egyes állapotokat jellemezze a következő atomi kijelentésekkel: *{alaphelyzet, üzemállapot, üzemképtelen}*



(Az állapotokba csak a jobb érthetőség kedvéért kerültek a szerverek állapotára utaló megjegyzések, ezek nem részei a Kripke-struktúrának.)

- Ellenőrizze a modellen, hogy az üzemállapotot kezdőállapotnak tekintve teljesül-e a következő CTL kifejezés: *E(¬üzemképtelen U alaphelyzet)*! Válaszát indokolja meg!

Az *üzemállapot* címkéjű állapotból létezik-e olyan útvonal, amelyen nem szerepel az *üzemképtelen* címke, amíg el nem érjük az *alaphelyzet* címkéjű állapotot?

Ilyen az az útvonal, amikor az *üzemállapot* címkéjű állapotból (amin ugyan nem szerepel az *üzemképtelen* címke, de az *alaphelyzet* címke sem, ezért tovább kell lépni) azonnal az *alaphelyzet* címkéjű állapotba lépünk.

A bizonyítás elvégezhető az iteratív állapotcímkézési eljárással is.

5. Temporális logikai követelmények értelmezése

Indokolja meg, hogy következő ekvivalencia helyes-e: $F\ Stop \vee F\ Start \equiv F\ (Stop \vee Start)$, ahol $\vee$ a logikai VAGY operátort jelöli, a *Stop* és *Start* pedig atomi kijelentések.

Az ekvivalencia helyes. Ha a jövőben van olyan állapot, ahol a Stop atomi kijelentés előfordul, vagy van olyan állapot, ahol a Start előfordul (ld. az ekvivalencia bal oldala), akkor ez egyben azt is jelenti, hogy van olyan állapot, ahol a Start vagy Stop előfordul (akár önmagában, akár a másikkal együtt). Hasonlóan a fordított irány is ellenőrizhető (az ekvivalencia jobb oldalából következik a bal oldala).

6. Temporális logikai követelmények értelmezése

Indokolja meg, hogy következő ekvivalencia helyes-e: *AG Stop* ≡ *not EF (not Stop)*, ahol *not* a logikai negálás operátort jelöli, a *Stop* pedig egy atomi kijelentés.

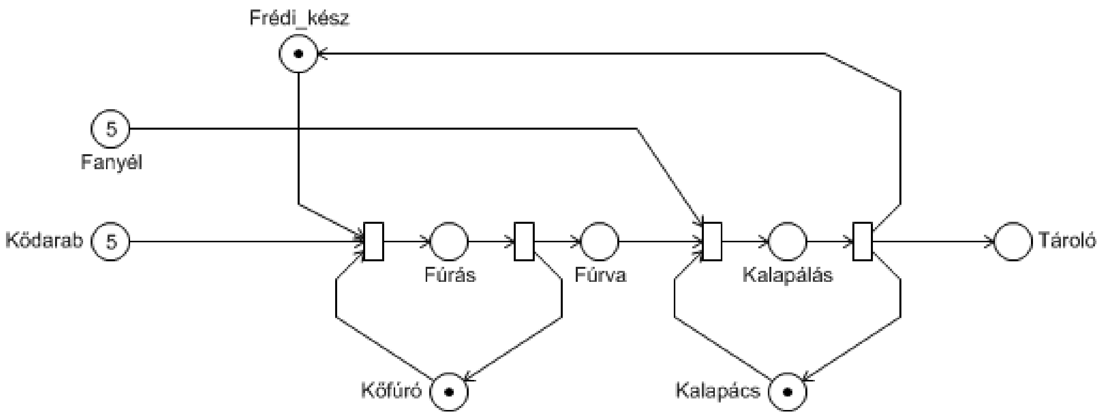
Az ekvivalencia helyes. Ha minden útvonalon, azok minden állapotán szerepel a Stop atomi kijelentés az állapot címkéi között (ld. az ekvivalencia bal oldala), akkor nem lehetséges, hogy létezik valamely útvonalon olyan állapot, ahol nem szerepel a Stop atomi kijelentés az állapot címkéi között. Hasonlóan a fordított irány is ellenőrizhető (az ekvivalencia jobb oldalából következik a bal oldala).

7. Modellezés Petri hálóval

Egy kőbalta készítéséhez egy ködarabra, egy fanyélre, egy kőfúróra és egy kalapácsra van szükség. Kezdetben egy-egy eszköz (kőfúró és kalapács) valamint öt ködarab és öt fanyél áll rendelkezésre. Először a ködarabba kell lyukat fúrni a kőfúró eszközzel, majd a kalapáccsal bele kell verni a fanyelet a lyukba. A kész kőbaltákat egy tárolóba kell tenni. Az eszközöket használat után vissza kell tenni a helyükre, hogy a következő kőbalta készítéséhez használhatók legyenek.

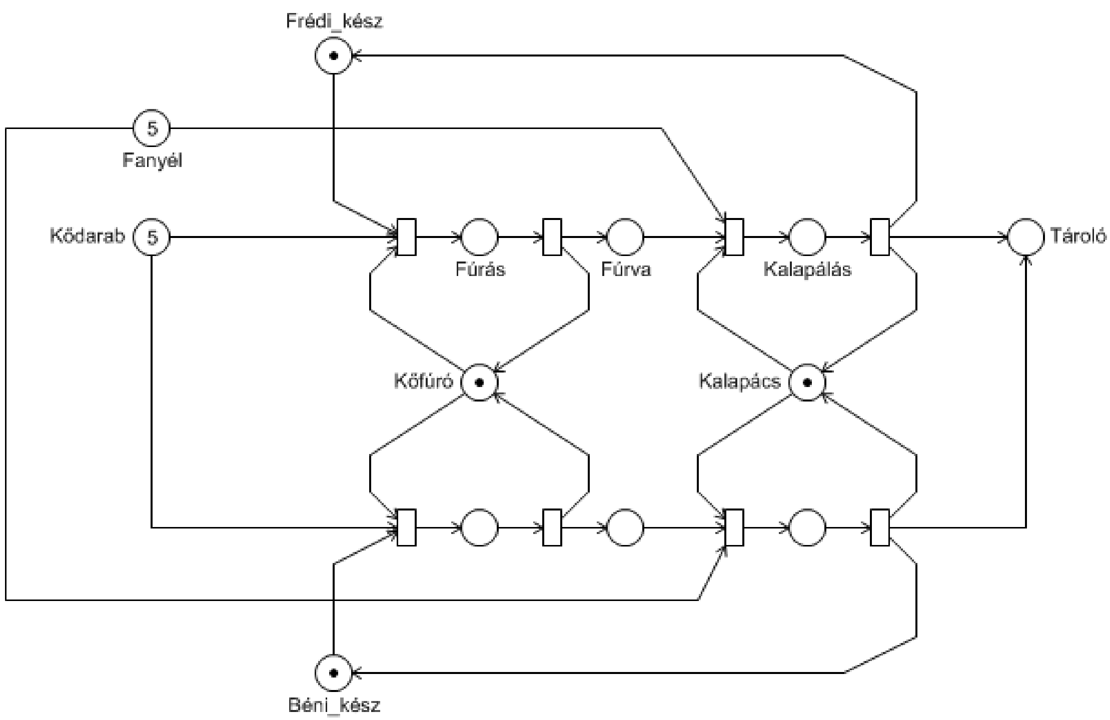
- Készítsen egy Petri hálót, ami modellezi azt a munkafolyamatot, amelynek során Frédi kőbaltákat készít! Tételezzük fel, hogy Frédi csak akkor fog a következő kőbalta készítésébe, ha az előző teljesen készen van.

  Megoldás: Egy részletes modell a munkafolyamatról:



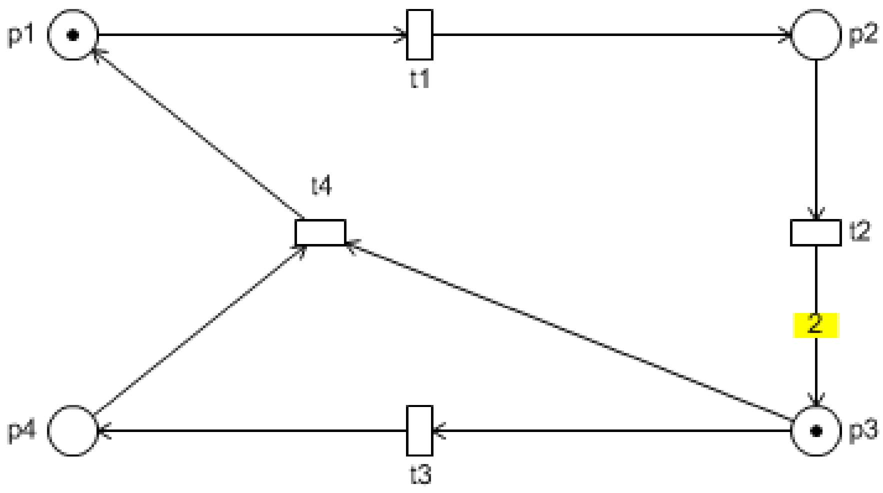
- Egészítse ki a modellt a következők szerint: Béni is dolgozik, azaz a szabad eszközöket használva a hozzávalókból ő is kőbaltákat készít. Béni is csak akkor fog a következő kőbalta készítésébe, ha az előző teljesen készen van.

  Megoldás: Egy részletes modell a munkafolyamatról:



8. Petri hálók elérhetőségi gráfja

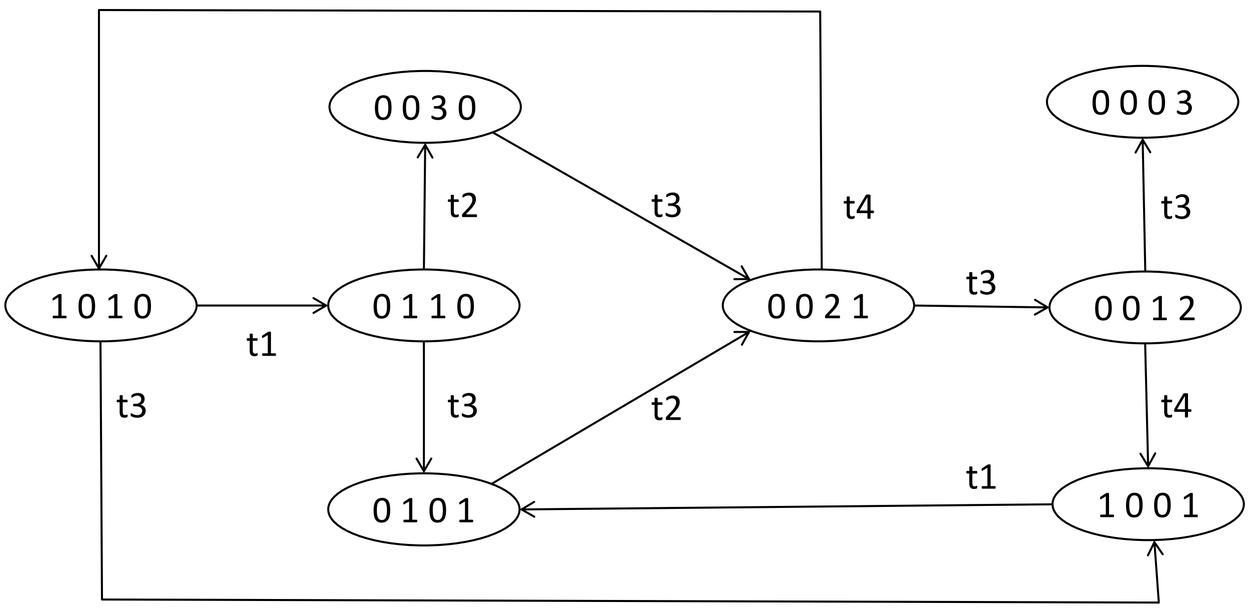
Rajzolja fel az alábbi Petri háló elérhetőségi gráfját!



Megoldás: Az elérhetőségi gráf a 9. feladat (lentebb) esetén felrajzolt elérhetőségi gráf.

9. Petri hálók dinamikus tulajdonságai

Alább látható egy Petri háló elérhetőségi gráfja. A kezdőállapot az (1 0 1 0) állapot.



Döntse el az alábbi állításokról, hogy igazak vagy hamisak, és indokolja meg döntését!

1. A Petri háló holtpontmentes.
    - Nem igaz, mert a (0 0 0 3) állapot holtpont (nincs kivezető tranzíció).
2. A Petri háló korlátos.
    - Igaz, mert minden elérhető állapotban, minden helyen legfeljebb 3 token található.

3. A Petri háló élő.
    - Nem igaz, mert például a (0 0 0 3) állapotból nem tüzelhet egyik tranzíció sem.
4. A Petri háló megfordítható.
    - Nem igaz, mert a kezdőállapot nem érhető el a (0 0 0 3) állapotból.
5. A (0 0 1 2) állapot visszatérő állapot.
    - Nem igaz, mert a (0 0 0 3) állapot felé lépve nem lehet ezt újra elérni.
6. A Petri hálónak van visszatérő állapota.
    - Nem igaz, mert bármelyik állapotból el lehet jutni a (0 0 0 3) állapotba, és onnan nem lehet az adott állapotba visszatérni.
7. A t3 tranzíció L1-élő.
    - Igaz, mert legalább egyszer tüzelhet (rögtön a kezdőállapotból is).
8. A t3 tranzíció L3-élő.
    - Igaz, mert van olyan tüzelési szekvencia, pl. a t3, t1, t2, t4 tüzelési szekvencia, ami a kezdőállapotból ciklust képez (azaz visszavezet a kezdőállapotba), és amiben t3 előfordul, tehát ebben a tüzelési szekvenciában t3 végtelenül sokszor tüzelhet.
9. A t1, t2, t3, t4 tüzelési szekvencia egy T-invariánst alkot.
    - Igaz, mert a kezdőállapotból ez a szekvencia visszavezet a kezdőállapotba, tehát az állapotot a lefutása nem változtatja meg.
10. Az (1, 1, 1, 1) súlyvektor egy P-invariánst határoz meg.
    - Nem igaz, mert ezekkel a súlyokkal összegezve az egyes állapotokban a helyek jelöléseit, nem lesz állandó az érték. Pl. az (1 0 1 0) állapot esetén 2 a súlyozott tokenösszeg, míg a (0 0 3 0) állapot esetén 3.