

Az UPPAAL egyes modellezési lehetőségeinek összefoglalása

Majzik István

BME Méréstechnika és Információs Rendszerek Tanszék

Résztevők együttműködése (1)

- Automaták interakciói üzenetküldéssel
 - Szinkron üzenetküldés
 - Alapértelmezett szemantika a csatorna (**chan**) esetén
 - Randevú (kölcsonös várakozás) közvetlenül támogatott
 - Aszinkron üzenetküldés
 - Csatorna **explicit modellezése** szükséges (átvesz és továbbít)
 - Konkurens üzenettovábbítás korlátozott (ha kézbesítésre vár, nem tud újat fogadni)
 - Broadcast üzenetküldés
 - Broadcast csatorna használata (**broadcast chan**)
 - Küldő feltétel nélkül továbbléphet (akkor is, ha nincs fogadó)
 - Az éppen várakozó fogadók szinkronizálódnak
- Paraméterezhető számú résztvevő
 - Paraméterezett méretű csatorna- és adatvektorok használata
 - Processz paraméterek, automatikus példányosítással is
 - **Követelményekben:** paraméterekre **forall** és **exists** kvantorok

Résztevők együttműködése (2)

- Automaták interakciói **megosztott változókkal**
 - **Globális változó deklarációja és használata**
 - Megosztott beállítás és lekérdezés
 - Lazán csatolt elosztott rendszerekben az implementáció kérdéses
 - Konkurencia problémákat vethet fel (atomi hozzáférés)
- Üzenetküldés **adattartalommal**
 - **Szinkron üzenetküldés és megosztott változó használata az adattartalomra**
 - A megosztott változó beállítása majd a szinkronizáció kezdeményezése
(külön átmeneteken, mert az Update a Sync után következik)
 - Ha több résztvevő írhatja a megosztott változót:
Atomi végrehajtásra figyelni kell (beállítás és szinkronizáció egy atomi művelet, köztük committed állapot)
 - **Modellezési technika** lehet (nem implementációs mintaként!)

Konkurens viselkedés

- Atomi műveletek
 - **Committed** állapot: Bejövő állapotátmenet után csak committed állapotból kimenő állapotátmenet következhet
 - Könnyű modellezés, de használatát különösen meg kell gondolni (a modellezett valós világban valóban megoldható-e?)
 - Implementációs nehézségek lazán csatolt elosztott rendszerekben
 - Hasonló hatású: Feltételvizsgálat és értékadás egy élen (egy állapotátmenet esetén)
 - Pl. test-and-set, nem ugyanaz, mintha külön átmeneten lenne
 - Függvények belseje nincs lépésenként kezelve
- Prioritás aktivitások között
 - Prioritás processzek között
 - Prioritás csatornák (azaz szinkronizációs lehetőségek) között
 - Használhatók, de a modellellenőrzést korlátozzák:
A<>, **E[]**, **-->**, **deadlock** nem ellenőrizhető

Időbeliség

- Várakozás korlátozása
 - Vezérlési hely invariáns (legfeljebb meddig)
 - Időhöz kötött feltétel a kilépő átmeneten (legalább meddig)
- Várakozás tiltása (de nem atomi művelet)
 - Sürgős (**urgent**) állapot: Nem várakozhat az adott állapotban, ha kiléphet
 - Sürgős (**urgent**) csatorna: Ha lehetséges a szinkronizáció, akkor nem várakozhat

Adattípusok

- Adattípusok finomítása
 - Tömbök, rekordok létrehozhatók
 - Paraméterezhető méretű vektorok hasznosak
 - Értékek tartománya megadható
 - Célszerű a legkisebb tartományt megadni (tárigény csökkenthető modellellenőrzéskor)
 - Meta változók
 - Állapotvektorba bekerül, de nem számít különbségnek az állapotban, ha eltérő az értéke
- Processzek és függvények paraméterei
 - Érték és referencia szerinti paraméterek lehetnek

Adatmanipuláció

- Összetett feltételvizsgálat
 - Boole **függvény** használata őrfeltételben (áttekinthetőbb)
 - Konkurens viselkedést korlátozza (függvények belseje nincs lépésenként kezelve)
- Összetett adatmanipuláció
 - **Függvény** használata akcióban
 - Feltételes végrehajtás (**if ... else...**), iteráció (**for, while, do**) és **return**
 - Példák: Üzenetsor kezelés, adatvektorok inicializálása, ...
 - Konkurens viselkedést korlátozza (függvények belseje nincs lépésenként kezelve)

Véletlen választás és kiértékelés

- Véletlen választás modellezése

- Választás (**select**) konstrukció egy átmenethez:

Változót véletlenszerűen adatértékhez köt egy adott tartományból

- Az átmeneten szinkronizációban, őrfeltételben, akcióban használható a kötött változó
- Minden lehetséges választást bejár modellellenőrzéskor

- A szekciók kiértékelése

- A szinkronizációhoz az őrfeltételek teljesülése szükséges
- Szinkronizáló élek esetén a küldő Update-ja a fogadóé előtt fut le, de a fogadó Guard után
- Nem számíthatunk a (fogadó) őrfeltételében arra, hogy a (küldő) szinkronizáló él akciója már beállított egy globális változót

