Adatfolyam hálók: széthajtogatás

Az alábbi táblázatban egy adatfolyam háló modell látható. A hálónak két csomópontja van:

- a master: **mst = ({wd}, {out}, {sm_0, sm_1}, sm_0, {r1, r2, r3, r4}, {ok, err, b0, b1})**,
- és a checker: **chk = ({out}, {wd}, {sc_0, sc_1}, sc_0, {r5, r6, r7}, {b0, b1, ok, err})**.

A modellben két darab *egy kapacitású* csatornát definiáltunk: az out jelű a {∅, b0, b1} állapotokban lehet (ahol ∅ az üres halmazt, tehát az üres csatornát jelöli), míg a wd jelű a {∅, ok, err} állapotokat veheti fel. A tüzelési szabályokat az alábbi táblázatban adtuk meg:

mst csomópont, M_{mst}={ok, err, b0, b1}	**chk csomópont, M_{chk}={b0, b1, ok, err}**
r1 = ⟨sm_0; wd=ok; sm_1; out=b0⟩	r5 = ⟨sc_0; out=b0; sc_1; wd=ok⟩
r2 = ⟨sm_1; wd=ok; sm_0; out=b1⟩	r6 = ⟨sm_1; out=∅; sm_0; wd=∅⟩
r3 = ⟨sm_1; wd=∅; sm_0; out=∅⟩	r7 = ⟨sm_1; out=∅; sm_0; wd=∅⟩
r4 = ⟨sm_1; wd=err; sm_0; out=b1⟩	

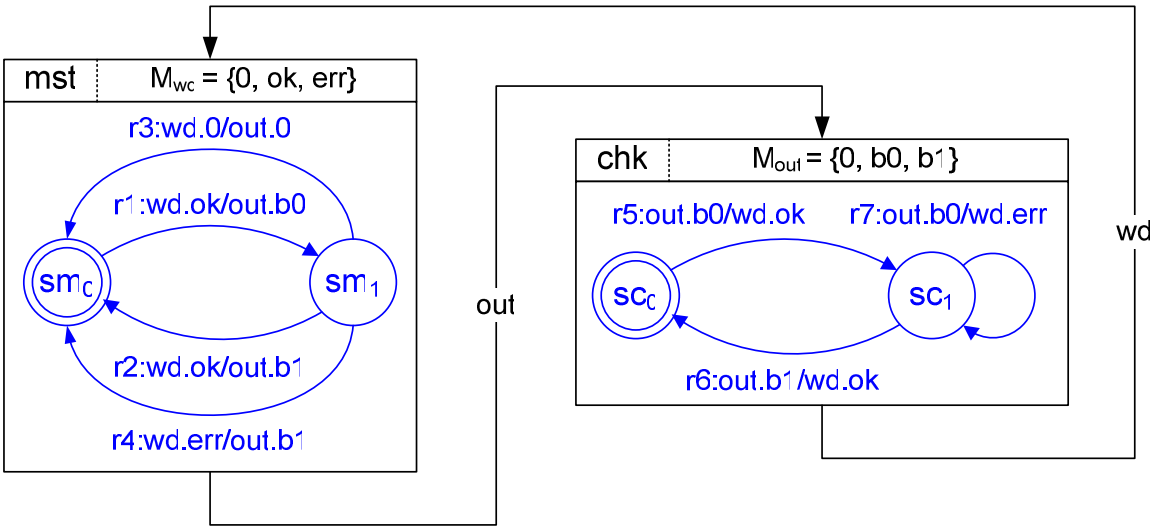
Például az r3 szabály jelentése: az mst csomópont sm_1 állapotból sm_0 állapotba kerül, eközben a wd bemeneti csatornáról nem vesz el, és az out kimeneti csatornára nem tesz ki semmit.

A modell szerint a master csomópont váltakozva ’b0’, ’b1’ tokeneket generál, de néha hibázik. Ezt ellenőrzi a checker csomópont, aki az általa kiadott ’ok’ és ’err’ tokenekkel jelzi az eredményt. A kezdőállapotban az out csatorna üres, a wd csatornában egy ’ok’ token van.

Megoldás

Széthajtogatás 1. lépése: a grafikus adatháló modell elkészítése

A formális leírás (a struktúrát a csomópontok leírása adja meg!) tüzelési szabályok alapján kialakítjuk a csomópontok működését leíró állapotgép modellt.

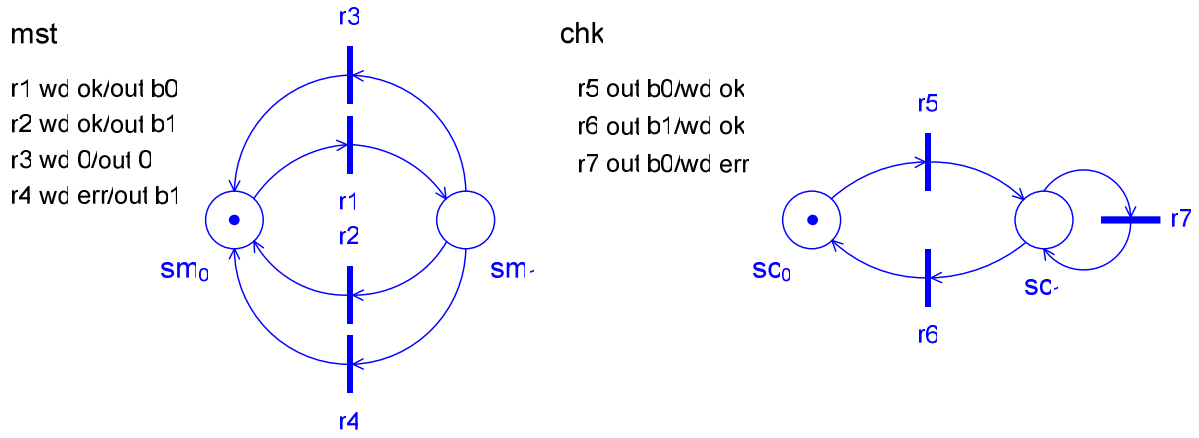


1. ábra: Adatfolyam háló grafikus modellje

Széthajtogatás 2. lépése: a csomópontok Petri háló modelljének kialakítása

A csomópontok Petri háló modelljének kialakítása triviális feladat, hiszen csak egy állapotgép modell Petri hálós megfelelőjét kell kialakítani. Nyilván ez egy állapotgép (SM) alosztályba tartozó hálót eredményez, amelyben az állapot-csomópontoknak a Petri háló helyei, a tüzelési szabályoknak a megfelelően irányított élekkel kiegészített tranzíciók, a

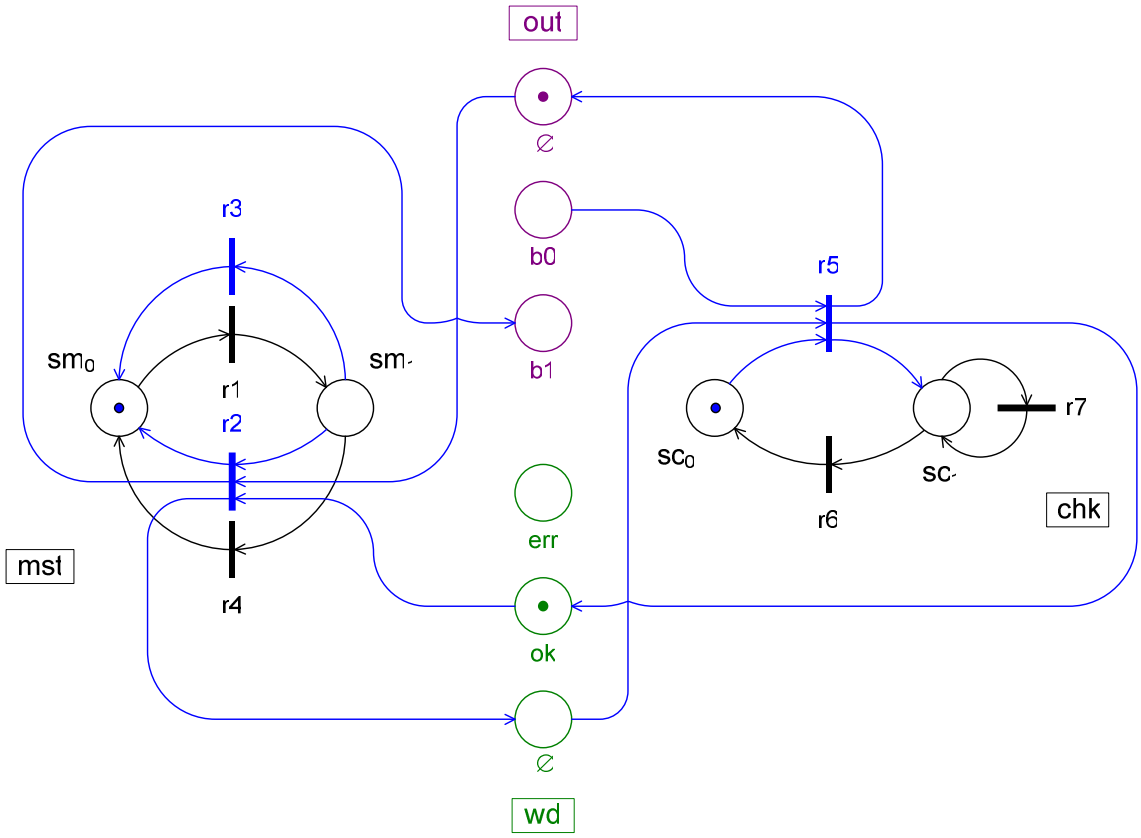
kezdőállapotnak pedig a megfelelő helyekre tett tokenekkel megadott kezdő token eloszlás felel meg.



2. ábra: Adatfolyam csomópontok "széthajtogatása"

Széthajtogatás 3. lépése: a csatornák modellje és kapcsolat a tüzelési szabályokkal

A következő lépésben elkészítjük a csatornák Petri háló modelljét. Ez egy 1 kapacitású csatornánál egyszerűen annyi helyből áll, amekkora a csatorna tokenkészlete (+1 hely az üres állapotnak), hiszen a csatorna ennyiféle állapotban lehet (2 vagy több kapacitású csatornánál a helyzet egészen más! Hiszen ott token n-esekkel kell dolgoznunk, ahol a sorrend is számít!).



3. ábra: A csatornák modellje és a tüzelési szabályok (az ábrán az r2, r3 és r5) kapcsolata a csatornákkal

Az ábrán a lila szín az out csatornát, a zöld a wd csatornát jelöli. Természetesen itt is meg kell adnunk a csatornák kezdőállapotának megfelelő kezdő tokeneloszlást.

Miután megalkottuk a csatornák modelljét, meg kell teremteni a csomópontok és a csatornák kapcsolatát, ami a tüzelési szabályokon keresztül valósul meg. A fenti ábrán az r2,

r3 és r5 szabályokat ábrázoltuk (kék színnel). A többi szabály „széthajtogatása” ezek alapján már önállóan elvégezhető.

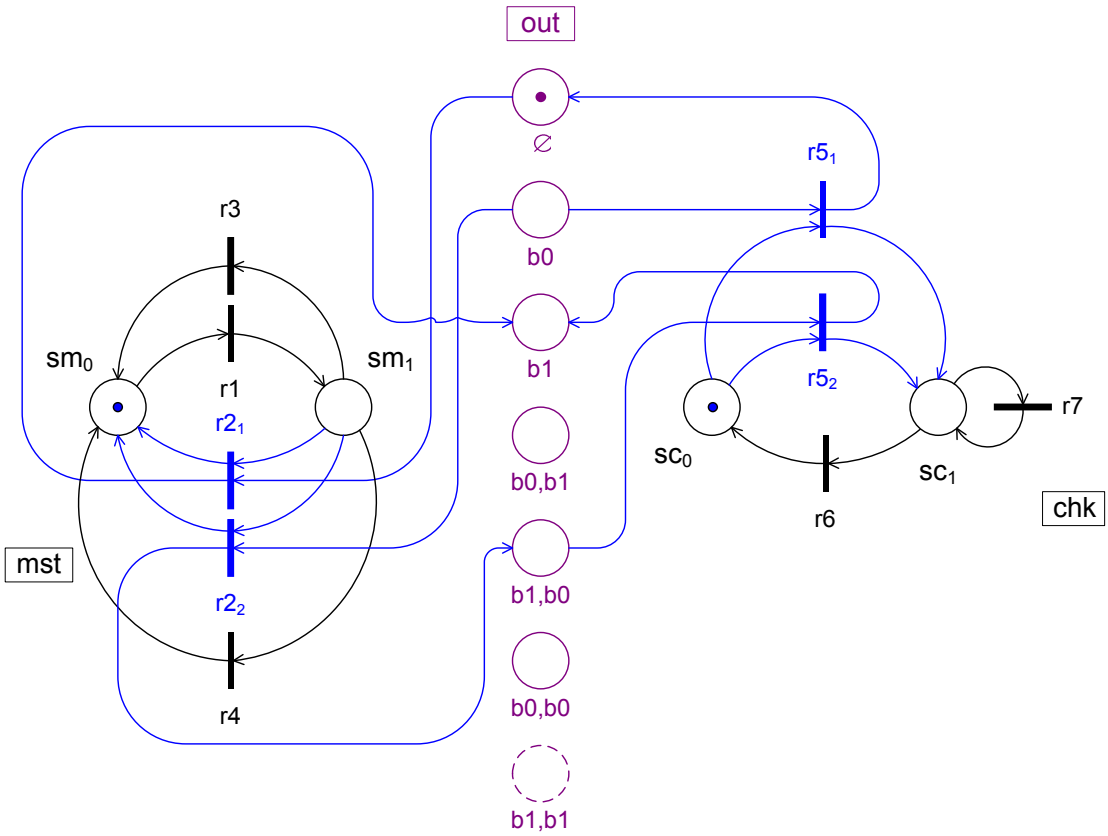
Az r2-es szabály magyarázata: ez a szabály – miközben a csomópont állapotát sm_0-ból sm_1-be viszi át – a wd csatornáról elvesz egy ’ok’ tokent (és ezzel kiüríti a wd csatornát, hiszen az 1 kapacitású), és az out csatornára kitesz egy ’b1’ tokent. Ez utóbbit (szintén az 1 kapacitású kimeneti csatorna miatt) csak akkor teheti meg, amikor a csatorna üres.

Az r5-ös szabály modellje az r2-eshez hasonlóan adódik. Az r3-as szabály érdekes: mivel ez a szabály nem igényel tokent a bemenő csatornáról és nem tesz ki tokent a kimenő csatornára, így ezekkel nincs is kapcsolata, csak az csomópont állapotát változtatja meg a csatornákét nem.

Széthajtogatás 2 kapacitású csatorna esetén

Ha a példában pl. az out csatorna 2 kapacitású lenne, hogyan kellene a széthajtogatást elvégezni? Ez esetben egyrészt a csatorna modellje összetettebb, hiszen nem csak egyedi tokenek, hanem 2 hosszúságú token szekvenciák lehetnek a csatornában.

A tüzelési szabályok végrehajtása is bonyolódik. Egyetlen tranzíció már nem tudja leírni a szabály működését, hiszen nem csak egy üres csatornába lehet egy tokent betenni, hanem egy 1 tokent tartalmazó csatornába is egy újabbat (illetve kivenni is lehet egy 2 tokent tartalmazó csatornából egyet). A csatorna állapotának megváltoztatásakor pedig figyelembe kell venni a FIFO működést. Az alábbi ábra erre mutat példát (csak az alapgondolatot vázolva fel: a wd csatornát az egyszerűség kedvéért elhagytuk, és az r2-es és r5-ös szabályok széthajtogatásának sem az összes lehetőségét rajzoltuk fel).



4. ábra: Széthajtogatás 2 kapacitású csatorna esetén