Lecţia 4:

Sisteme interactive finite; Limbaje 2-dimensionale

v1.0

Gheorghe Stefanescu — Universitatea Bucureşti

Metode de Dezvoltare Software, Sem.2
Februarie 2007— Iunie 2007

Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- *Generalităţi*
- Sisteme interactive finite
- Sisteme de pavare (tiling systems)
- Logică monadică de ordinul 2
- Concluzii, diverse, etc.

Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- Generalităţi
- *Sisteme interactive finite*
- Sisteme de pavare (tiling systems)
- Logică monadică de ordinul 2
- Concluzii, diverse, etc.

Sisteme interactive finite

Sisteme interactive finite (ori FIS-uri) – punct de plecare:

Sunt un fel de automate 2-dimensionale care mixează
– un automat pentru *fire de execuţie* cu
– un automat pentru *interacţia firelor*

Util, dar înşelător:

- *nu se mixează două automate*, ci *cele două viziuni se combină* spre a da acest concept

  [Spre exemplu, există două FIS*'uri cu aceleaşi automate proiecţie, dar care sunt diferite (recunosc limbaje diferite).]*

- termenul "automat" e cumva greşit: un automat reprezintă o "transformare de stări", deci un "automat 2-dimensional" este un fel de automat peste stări 2-dimensionale

..Sisteme interactive finite

Util, dar... (cont.):

- sistemele interactive finite sunt de departe diferite:
  – o dimensiune (convenţional, cea verticală) este pentru această *transformare de "stare"*, dar
  – cealaltă dimensiune este pentru o *transformare a "claselor de job-uri"* – (*un agent este considerat şi ca o funcţie care transformă job-urile*)

- *sistemele sunt deschise*: se dă
  – o stare iniţială şi un job în locul de intrare
  spre a obţine
  – o stare finală şi un job în locul de ieşire.

..Sisteme interactive finite

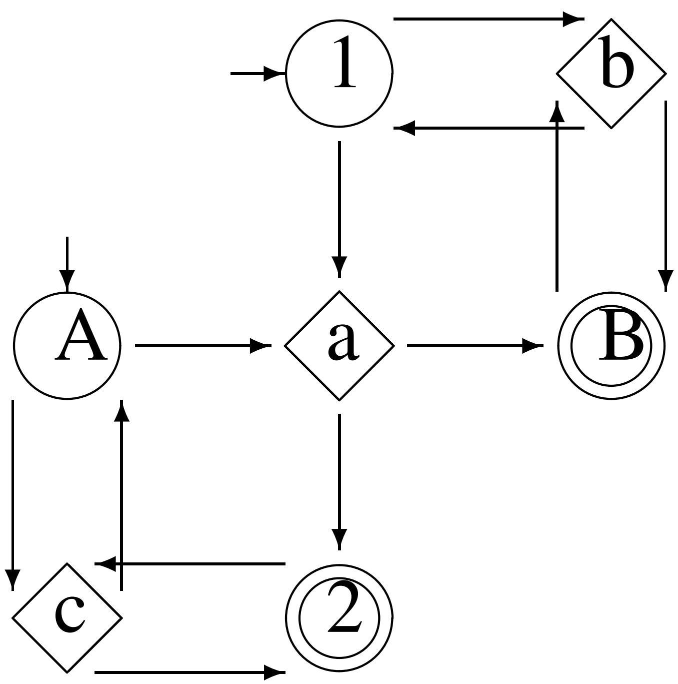
Un *sistem interactiv finit* (ori FIS) este un hiper-graf cu

- două tipuri de noduri
  - unul pentru *stări* (etichetat cu numere ori litere mici)
  - unul pentru *clase* (etichetat cu litere mari)
- un tip de (hiper) arce, numite *tranziţii*
  - tranziţiile sunt etichetate cu litere (pentru atomi 2-dim) şi au *două săgeţi care intră* (una de la un nod-clasă, cealaltă de la un nod-stare) şi *două săgeţi care ies* (una spre un nod-clasă, cealaltă spre un nod-stare)
- unele clase ori stări sunt
  - *iniţiale* (indicate folosind mici săgeţi de intrare) ori
  - *finale* (indicate prin folosirea cercurilor duble)

..Sisteme interactive finite

Exemplu: Un FIS S şi

reprezentarea sa grafică:



reprezentarea sa textuală (ori *"cu cruci"*):

- $\begin{array}{|c|c|c|} \hline & 1 & \\ \hline A & a & B \\ \hline & 2 & \\ \hline \end{array}$, $\begin{array}{|c|c|c|} \hline & 2 & \\ \hline A & c & A \\ \hline & 2 & \\ \hline \end{array}$, şi $\begin{array}{|c|c|c|} \hline & 1 & \\ \hline B & b & B \\ \hline & 1 & \\ \hline \end{array}$
- $A, 1$ sunt iniţiale
- $B, 2$ finale

Scenarii, criterii de recunoaştere

Sistemele interactive finite *recunosc* griduri în felul următor:

(1) Fie date: un FIS S; un grid w (m linii, n coloane); o secvenţă de stări iniţiale $b_n = s_1 \ldots s_n$; şi una de clase iniţiale $b_w = C_1 \ldots C_m$. Atunci:

- Inserăm $s_1, \ldots, s_n$ pe bordura de nord a lui w (de la stânga la dreapta) şi $C_1, \ldots, C_m$ pe bordura de vest a lui w (de sus în jos).
- (1) Parsăm w selectând atomi minimali încă neprocesaţi.
- (2) Pentru fiecare astfel de atom a, dacă s este starea din nord şi C clasa din vest, alegem o tranziţie $\begin{array}{ccc} & \boxed{s} & \\ \boxed{C} & \boxed{a} & \boxed{C'} \\ & \boxed{s'} & \end{array}$ din S (dacă există), inserăm s' pe bordura sa din sud şi C' pe bordura sa din est, şi considerăm că acest atom a fost procesat.
- (3) Repetăm paşii 1-2 cât timp se poate.

..Scenarii, criterii de recunoaştere

In final, dacă

- toţi atomii lui w au fost procesaţi,
- bordura de est b_e conţine doar clase finale, şi
- bordura de sud b_s conţine numai stări finale

atunci avem un *scenariu de succes pentru w relativ la* b_n, b_w, b_e, b_s.

(2) Un grid w este *recunoscut de S relativ la condiţiile de bordură* B_n, B_w, B_e, B_s dacă există un scenariu de succes pentru w cu $b_n \in B_n$, $b_w \in B_w$, $b_e \in B_e$, $b_s \in B_s$; mulţimea lor este $L(S; B_n, B_w, B_e, B_s)$

(3) Un limbaj de griduri L este *recunoscut de S* dacă există nişte limbaje regulate B_n, B_w, B_e, B_s pentru borduri astfel ca $L = L(S; B_n, B_w, B_e, B_s)$.

(4) Scriem simplu $L(S)$ când condiţiile de bordură sunt subînţelese (limbaje complete de tipul Γ^*).

..Scenarii, criterii de recunoaştere

Exemplu (un scenariu de succes):

- Dat sistemul interactiv finit de mai sus (Fila 1.18), limbaje complete pentru borduri, şi gridul $w =$
```
abb
cab
cca
```
, avem:

- $w_0 =$
```
 1 1 1
Aa b b

Ac a b

Ac c a
```
; $w_1 =$
```
 1 1 1
AaBb b
 2
Ac a b

Ac c a
```
; $w_2 =$
```
 1 1 1
AaBbBb
 2 1
Ac a b

Ac c a
```
; $w_3 =$
```
 1 1 1
AaBbBb
 2 1
AcAa b
 2
Ac c a
```
;

  $\ldots$; $w_9 =$
```
 1 1 1
AaBbBbB
 2 1 1
AcAaBbB
 2 2 1
AcAcAaB
 2 2 2
```

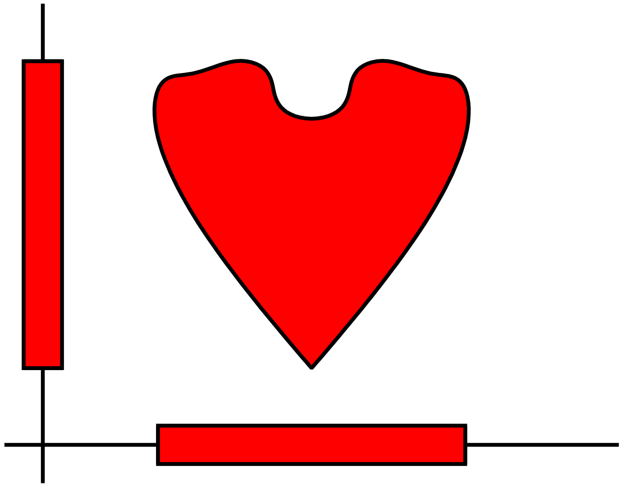
De fapt, limbajul recunoscut $L(S)$ este format din *pătrate cu a pe diagonala principală, cu b în jumătatea din dreapta-sus, şi cu c în jumătatea din stânga-jos*.

Proiecţii pe stări şi pe clase

Automate finite nedeterministe familiare (NFA-uri) se obţin *neglijând o dimensiune*

- *proiecţia pe stări* este NFA-ul state(S)
  – obţinut neglijând clasele
- *proiecţia pe clase* este NFA-ul class(S)
  – obţinut neglijând stările

proiecţiile pot pierde informaţie



Exemple de proiecţii

Pentru FIS-ul introdus anterior:

- Proiecţia pe clase $\mathsf{class}(S)$ este definită de tranziţiile $A \xrightarrow{a} B, \quad A \xrightarrow{c} A, \quad B \xrightarrow{b} B$, cu A iniţială şi B finală.
- Proiecţia pe stări $\mathsf{state}(S)$ este definită de tranziţiile $1 \xrightarrow{a} 2, \quad 2 \xrightarrow{c} 2, \quad 1 \xrightarrow{b} 1$ cu 1 iniţială şi 2 finală.

Acest sistem interactiv finit are o interesantă *proprietate de intersecţie*:

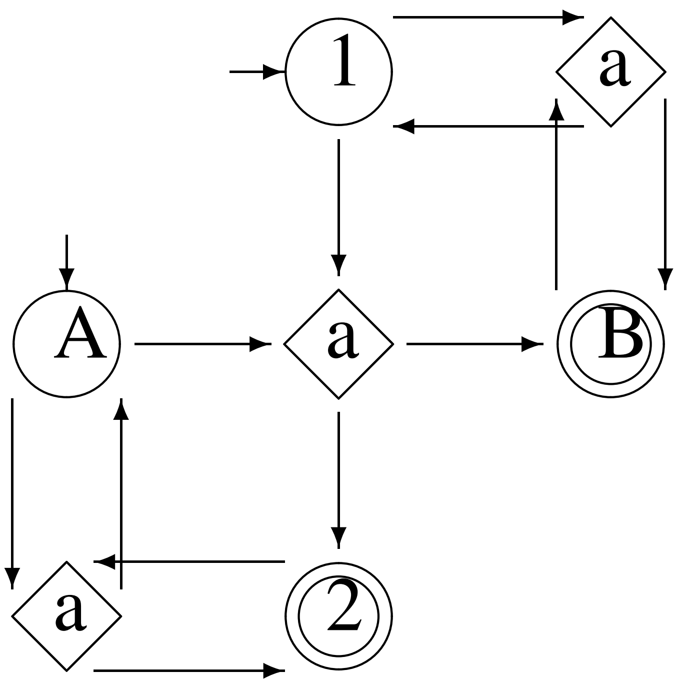
$$L(S) = L(\mathsf{state}(S))^\dagger \cap L(\mathsf{class}(S))^\star$$

evidenţiind o *slabă interacţie* între transformarea pe stări şi cea pe clase. (Un grid w este în $L(\mathsf{state}(S))^\dagger$ dacă fiecare coloană din w este recunoscută de $\mathsf{state}(S)$ în sens uzual. Asemănător pentru $L(\mathsf{class}(S))^\star$, folosind liniile.)

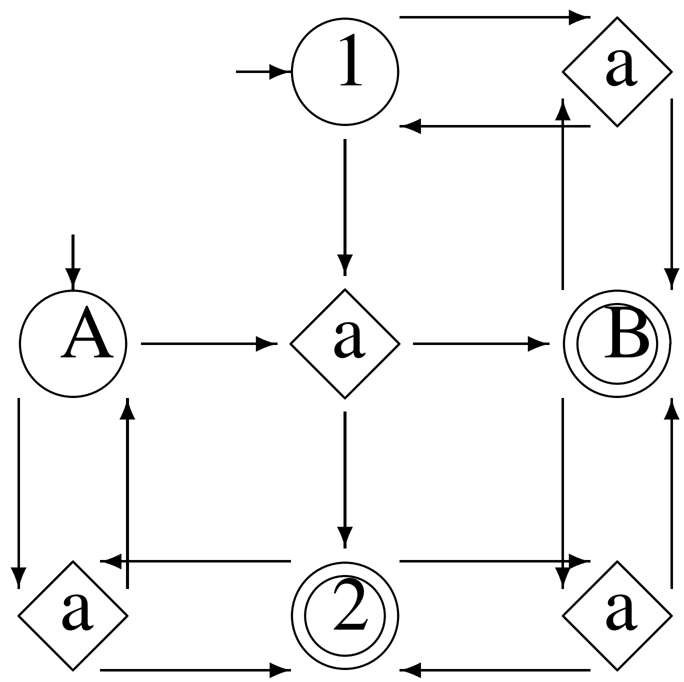
Alte exemple: Griduri pătrate

FIS-uri pentru griduri pătrate de *a*-uri

```
a      aa      aaa      aaaa     ...
       aa      aaa      aaaa
               aaa      aaaa
                        aaaa
```



$Sq1$:

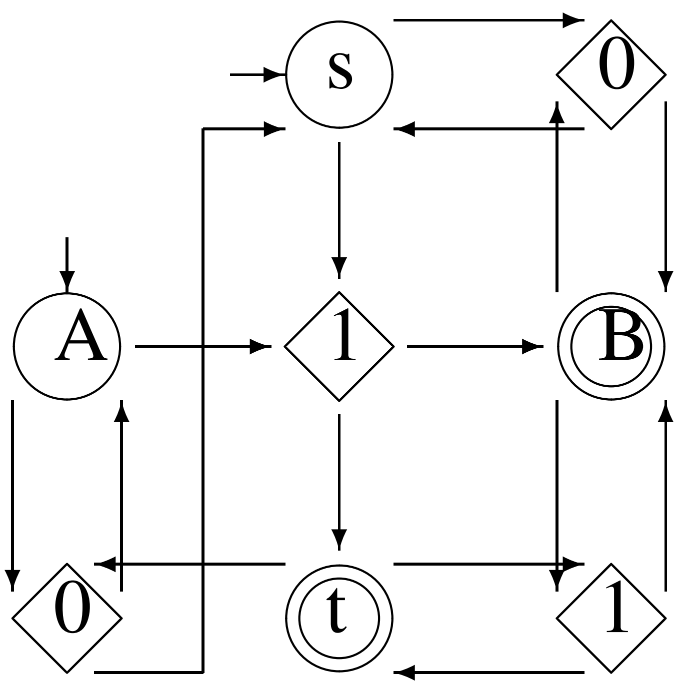


$Sq2$:

Alte exemple: O funcţie exponenţială

Un FIS *Pow* pentru limbajul

1	10	100	...
	01	010	
	11	110	
		001	
		101	
		011	
		111	



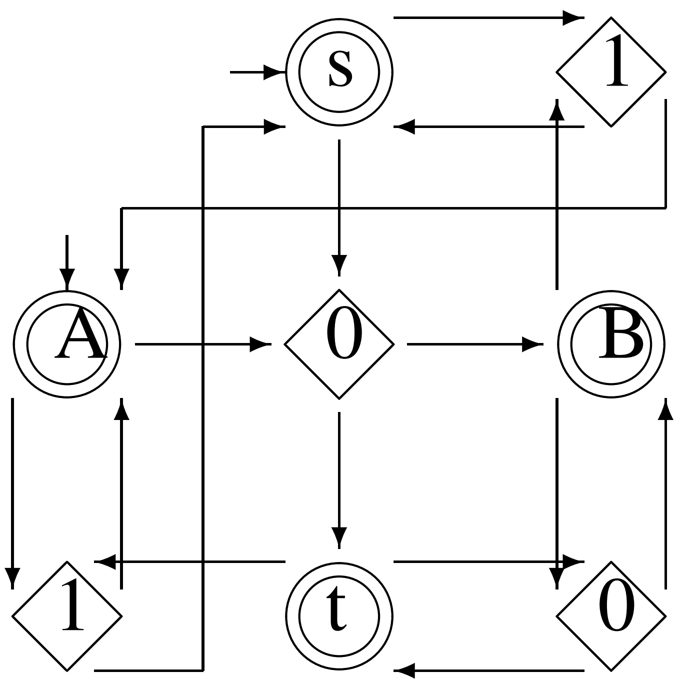
Pow

un scenariu de succes:

s s s
*A*1*B*0*B*0*B*
t s s
*A*0*A*1*B*0*B*
s t s
*A*1*B*1*B*0*B*
t t s
*A*0*A*0*A*1*B*
s s t
*A*1*B*0*B*1*B*
t s t
*A*0*A*1*B*1*B*
s t t
*A*1*B*1*B*1*B*
t t t

Alte exemple: Triunghiul lui Pascal

Un FIS *Pas* care recunoaşte mulţimea de griduri peste $\{0,1\}$ care
– au pe laturile de nord şi vest secvenţe alternante de 0 şi 1 (începând cu 0) şi,
– de-a lungul diagonalei secundare, satisfac regula de recurenţă din triunghiul lui Pascal, modulo 2



Pas

un scenariu de succes:

$$
\begin{array}{c}
\;\; s \;\; s \;\; s \;\; s \;\; s \;\; s \\
A0B1A0B1A0B1A \\
\;\; t \;\; s \;\; t \;\; s \;\; t \;\; s \\
A1A0B0B1A1A0B \\
\;\; s \;\; t \;\; t \;\; s \;\; s \;\; t \\
A0B0B0B1A0B0B \\
\;\; t \;\; t \;\; t \;\; s \;\; t \;\; t \\
A1A1A1A0B0B0B \\
\;\; s \;\; s \;\; s \;\; t \;\; t \;\; t \\
A0B1A0B0B0B0B \\
\;\; t \;\; s \;\; t \;\; t \;\; t \;\; t \\
A1A0B0B0B0B0B \\
\;\; s \;\; t \;\; t \;\; t \;\; t \;\; t
\end{array}
\quad .
$$

FIS-**uri vs. expresii regulate**

Să notăm că:

- automatele proiecţie $\mathsf{state}(Pas)$ şi $\mathsf{class}(Pas)$ pentru FIS-ul de mai sus sunt automate reset care au toate stările/clasele finale, deci recunosc *limbaje complete*.
- intersecţia lor este limbajul complet de griduri, *foarte departe* de gridurile cu proprietatea dată de triunghiul lui Pascal

Rezultate:

Teorema A: Dacă toate tranziţiile dintr-un FIS *au litere distincte, atunci* $L(S) = L(\mathsf{state}(S))^{\dagger} \cap L(\mathsf{class}(S))^{\star}$

Teorema B: Expresiile regulate 2-dim, îmbogăţite cu homomorfisme, sunt echivalente cu FIS*-urile.*

Alte exemple: Griduri spirale

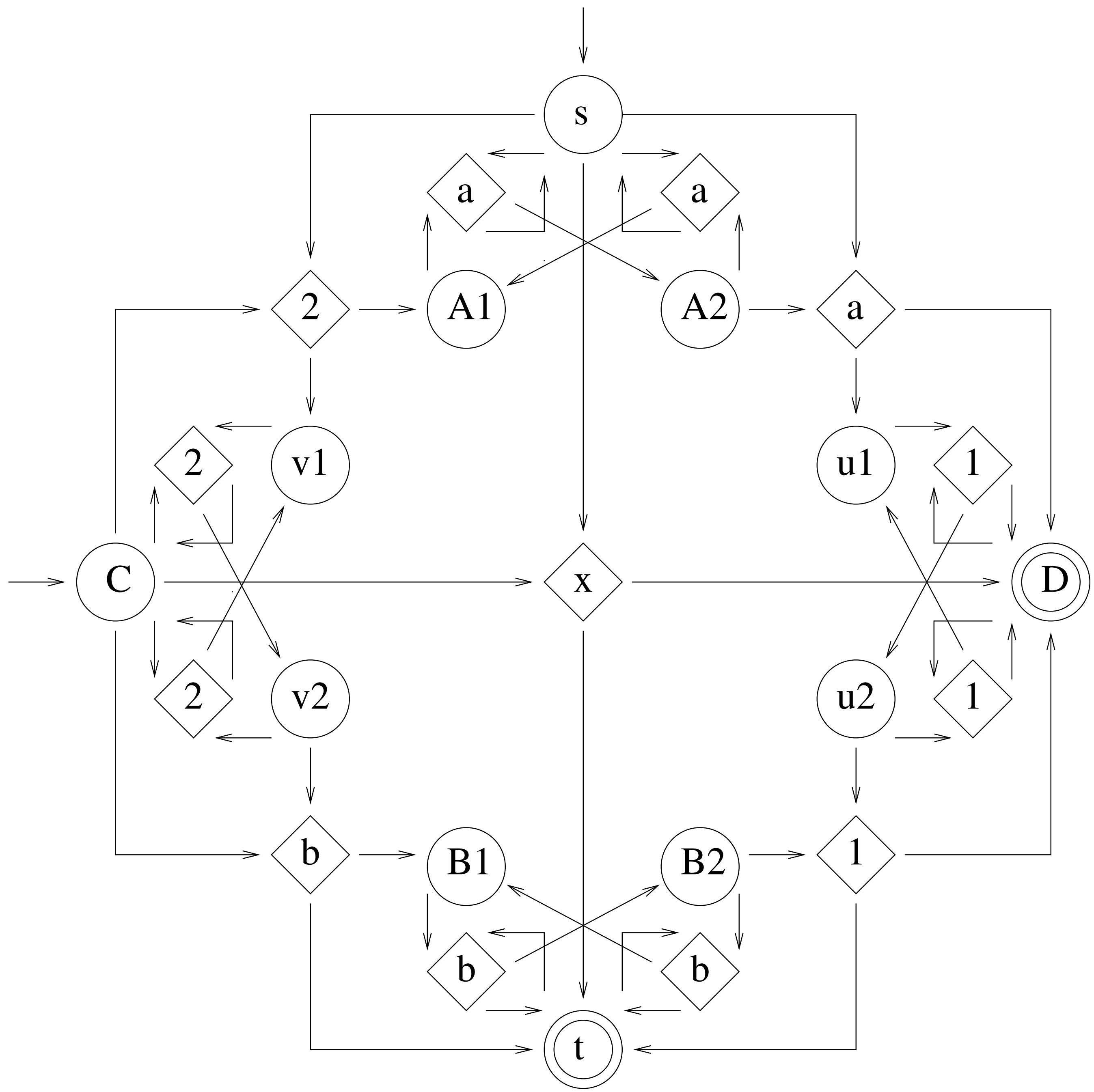
Un FIS pentru griduri "spirale"

```
x

2aa
2x1
bb1

2aaaa
22aa1
22x11
2bb11
bbbb1

:
```



Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- Generalităţi
- Sisteme interactive finite
- *Sisteme de pavare (tiling systems)*
  - *Limbaje locale şi sisteme de pavare*
  - Frontiera I: De la griduri la cuvinte
  - Frontiera II: De la cuvinte la griduri
- Logică monadică de ordinul 2
- Concluzii, diverse, etc.

Limbaje locale şi hv-locale

Limbajele de griduri acceptate de FIS-uri pot fi representate şi cu ajutorul *dominourilor*:

– pentru un grid cu bord $\widehat{w}$, notăm cu $B_{r,s}(\widehat{w})$ mulţimea sub-gridurilor sale de tip $r \times s$;

– un limbaj $L \subseteq V^{\dagger\star}$ este *local* dacă există o mulţime Δ de griduri 2×2 peste $V \cup \{\sharp\}$ astfel ca

$$L = \{w \in V^{\dagger\star} | B_{2,2}(\widehat{w}) \subseteq \Delta\}$$

– un limbaj $L \subseteq V^{\dagger\star}$ este *hv-local* dacă există o mulţime Δ de dominouri orizontale şi verticale peste $V \cup \{\sharp\}$ astfel ca

$$L = \{w \in V^{\dagger\star} | B_{1,2}(\widehat{w}) \cup B_{2,1}(\widehat{w}) \subseteq \Delta\}$$

Limbaje recunoscute

Se poate demonstra că

Teorema:

Există un FIS *care acceptă limbajul de griduri L peste V*

dnd

există un limbaj L' local peste un alfabet V' şi un homomorfism $h : V' \rightarrow V$ cu $h(L') = L$

dnd

există un limbaj L'' hv-local peste un alfabet V'' şi un homomorfism $h : V'' \rightarrow V$ cu $h(L'') = L$.

Un astfel de limbaj se numeşte limbaj *recunoscut*.

..Limbaje recunoscute

- Combinaţia *limbaj local + homomorfism* se numeşte *sistem de pavare* (engl. *tiling system*).

- Clasa de limbaje definită de sistemele de pavare rămâne *invariantă* dacă se folosesc *blocuri mai complexe* în Δ, de exemplu griduri 2×3.

Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- Generalităţi
- Sisteme interactive finite
- *Sisteme de pavare (tiling systems)*
  - Limbaje locale şi sisteme de pavare
  - *Frontiera I: De la griduri la cuvinte*
  - Frontiera II: De la cuvinte la griduri
- Logică monadică de ordinul 2
- Concluzii, diverse, etc.

Frontiera limbajelor de griduri

Frontiera:

– *frontiera* unui grid $w \in V^{\dagger\star}$ de tip $m \times n$ este linia de sus, anume $fr(w) = w(1,1)w(1,2)\dots w(1,n)$
– *frontiera* unui limbaj $L \subseteq V^{\dagger\star}$, este $fr(L) = \{fr(w) | w \in L\}$

Demonstrăm următoarea

Teoremă: *Imaginea prin funcţia frontiera fr a clasei limbajelor de griduri recunoscute $\{fr(L) | L\ recunscut\}$ este clasa limbajelor context-senzitive.*

De la griduri la cuvinte

Partea I: *Dacă L este limbaj de griduri recunoscut, atunci $fr(L)$ este context-senzitiv.*

Cum limbajele context-senzitive sunt închise la homomorfisme, e suficient de arătat că pentru un limbaj hv-local $K \subseteq V^{\dagger\star}$, $fr(K)$ e context-senzitiv — în fapt, vom construi o gramatică context-senzitivă pentru $\sharp fr(K)\sharp\sharp$.

Fie Δ mulţimea dominourilor pentru K; presupunem că $\begin{smallmatrix}\# & \#\end{smallmatrix}$ şi $\begin{smallmatrix}\# \\ \#\end{smallmatrix}$ sunt în Δ (altfel, K e vid).

..De la griduri la cuvinte

Idee: Construim o gramatică de tip context-senzitiv care verifică linie-cu-linie dacă gridul este în K; se pleacă cu linia de jos şi, în cursul verificării, se înlocuieşte o linie (curentă) cu linia de deasupra; în final, când testul e gata, rămînem cu linia de sus a lui w.

Construcţia gramaticii $G = (X, Y, P, S)$:

- *terminale* sunt $X = \{a \in V \mid \begin{smallmatrix}\texttt{\#}\\ \texttt{a}\end{smallmatrix} \in \Delta\} \cup \{\#\}$
- *neterminale* sunt $Y = (V \setminus X) \cup \{S, A, L, R\}$ (S este axioma de start)
- *producţiile* sunt grupate pe funcţionalităţi:

..De la griduri la cuvinte

– *generarea liniei de jos*

$S \longrightarrow \#aA$ pentru $\mathtt{\#a}\,,\ \begin{smallmatrix}\mathtt{a}\\ \mathtt{\#}\end{smallmatrix} \in \Delta, \quad a \in V$

$aA \longrightarrow abA$ pentru $\mathtt{ab}\,,\ \begin{smallmatrix}\mathtt{b}\\ \mathtt{\#}\end{smallmatrix} \in \Delta, \quad a, b \in V$

$aA \longrightarrow aL\#$ pentru $\mathtt{a\#} \in \Delta, \quad a \in V$

– *return la prima coloană*

$aL \longrightarrow La$ pentru $a \in V$

$\#L \longrightarrow \#R$

– *verificare orizontală şi verticală (distrugând vechea linie)*

$bRa \longrightarrow bcR$ pentru $\mathtt{bc}\,,\ \begin{smallmatrix}\mathtt{c}\\ \mathtt{a}\end{smallmatrix} \in \Delta, \quad a, c \in V, \quad b \in V \cup \{\#\})$

$aR\# \longrightarrow aL\#$ pentru $\mathtt{a\#} \in \Delta, \quad a \in V$

– *sfârşit de derivare*

$L\# \longrightarrow \#\#$

..De la griduri la cuvinte

- Se vede uşor că pentru $w \in K$, grid de tipul $m \times n$ cu liniile $w_1, \ldots, w_m$, există derivarea

$$\begin{aligned} S &\rightarrow^* \#w_m L\# \rightarrow^* \#Rw_m\# \rightarrow^* \#w_{m-1}L\# \\ &\rightarrow^* \ldots \rightarrow^* \#Rw_2\# \rightarrow^* \#w_1 L\# \rightarrow \#w_1\#\# \end{aligned}$$

  In plus, toate literele din w_1 sunt terminale, deci $\#fr(w)\#\# = \#w_1\#\# \in L(G)$, adică $\#fr(K)\#\# \subseteq L(G)$.

- Reciproc,
  – o derivare terminală în $L(G)$ se încheie doar cu $L\# \rightarrow \#\#$;
  – apariţia lui L în faţa lui $\#$ asigură verificarea condiţiilor de grid pentru toate liniile produse de trecerile lui L prin secvenţă;
  – singurul lucru neverificat este condiţia de grid pentru linia de sus (linia 1), dar asta rezultă din faptul că derivarea e terminală (şi din definiţia simbolurilor terminale)

Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- Generalităţi
- Sisteme interactive finite
- *Sisteme de pavare (tiling systems)*
  - Limbaje locale şi sisteme de pavare
  - Frontiera I: De la griduri la cuvinte
  - *Frontiera II: De la cuvinte la griduri*
- Logică monadică de ordinul 2
- Concluzii, diverse, etc.

De la cuvinte la griduri

Partea II: *Dacă L este limbaj context-senzitiv, există un limbaj de griduri recunoscut K' cu $fr(K') = L$.*

Pentru această inplicaţie folosim o caracterizare *echivalentă* a limbajelor context-senzitive, anume efaptul că ele sunt recunoscute de *automate liniare marginite*, i.e., maşini Turing particulare care funcţionează în acelaşi spaţiu de lucru.

..De la cuvinte la griduri

Un *automat liniar mărginit* M este un 6-uplu $M = (Q, V, Y, \delta, q_0, F)$ format din stări, limbaj de intrare, alfabet de bandă (care conţine intrările), funcţie de tranziţie, stare iniţială, stări finale, respectiv; Funcţia de tranziţie

$$\delta : Q \times Y \to 2^{Q \times Y \times \{-1,0,1\}}$$

se foloseşte spre a defini *paşii de calcul*

$$u.bqa.v \vdash uq'b.a'v, \text{ cu } u, v \in Y^*, a, b \in Y, (q', a', -1) \in \delta(q, a)$$
$$uqa.v \vdash uq'a'.v, \text{ cu } u, v \in Y^*, a \in Y, (q', a', 0) \in \delta(q, a)$$
$$uqa.v \vdash u.a'q'v, \text{ cu } u, v \in Y^*, a \in Y, (q', a', 1) \in \delta(q, a)$$

Un cuvânt w este *acceptat* dacă există o stare finală q şi un calcul

$$q_0 w \vdash \ldots \vdash w'q$$

..De la cuvinte la griduri

Un automat liniar mărginit M care acceptă L va fi simulat cu un limbaj local folosind *griduri de tipul* 2×3, nu simple dominouri (se poate arăta că sunt mecanisme echivalente)

Construcţie: Fie automatul $M=(Q,V,Y,\delta,q_0,F)$;
construim un limbaj local K dat de următorul set de griduri 2×3 peste $V\cup(Q\times Y)$:

- Punem *starea iniţială* pe prima celulă a primei linii
  – $\begin{array}{|c|c|c|}\hline \# & \# & \# \\ \hline \# & (q_0,a) & c \\ \hline \end{array}$, $\begin{array}{|c|c|c|}\hline \# & \# & \# \\ \hline (q_0,a) & b & c \\ \hline \end{array}$, $\begin{array}{|c|c|c|}\hline \# & \# & \# \\ \hline a & b & c \\ \hline \end{array}$
  cu $a,b\in Y, c\in Y\cup\{\#\}$

..De la cuvinte la griduri

- Simulăm *tranziţia automatului* prin trecerea de la o linie la alta
  - $\begin{array}{|c|c|c|}\hline a & (q,b) & c \\ \hline a & (q',b') & c \\ \hline\end{array}$ cu $(q',b',0) \in \delta(q,b), a,c \in Y \cup \{\#\}$
  - $\begin{array}{|c|c|c|}\hline a & (q,b) & c \\ \hline (q',a) & b' & c \\ \hline\end{array}$ cu $(q',b',-1) \in \delta(q,b), a \in Y, c \in Y \cup \{\#\}$
  - $\begin{array}{|c|c|c|}\hline a & (q,b) & c \\ \hline a & b' & (q',c) \\ \hline\end{array}$ cu $(q',b',1) \in \delta(q,b), a \in Y \cup \{\#\}, c \in Y$

..De la cuvinte la griduri

In plus, folosim gridurile $\begin{array}{|c|c|c|}\hline a_1 & a_2 & a_3 \\ \hline a'_1 & a'_2 & a'_3 \\ \hline\end{array}$ cu

(1) $a_1, a'_1, a_3, a'_3, a'_2 \in Y \cup \{\#\}, a_2 \in Y$,
(2) $(a'_2 \in Q \times Y \Rightarrow (a_1, a_3) \notin (Y \cup \{\#\})^2)$
(3) $(a_i \in Y \Rightarrow a'_i \in \{a_i\} \cup Q \times \{a_i\})$

spre a ne asigura că literele fără stări nu se schimbă de la o linie la alta

- In fine, verificăm că ultima linie reprezintă o configuraţie finală
  - $\begin{array}{|c|c|c|}\hline a & b & c \\ \hline \# & \# & \# \\ \hline\end{array}$ cu $a \in Y \cup \{\#\}, b \in Y, c \in V \cup (Q \times Y)$
  - $\begin{array}{|c|c|c|}\hline a & (q,b) & \# \\ \hline \# & \# & \# \\ \hline\end{array}$ cu $a \in Y \cup \{\#\}, b \in Y, (q', c', 1) \in \delta(q, b)$, $q' \in F, c' \in Y$

..De la cuvinte la griduri

Din simulare, rezultă că pentru un grid w,

$$\widehat{w} \in K$$

dnd

w are prima linie de forma $(q_0, w_1^1)w_1^2 \dots w_1^n$
cu $w_1^1 w_1^2 \dots w_1^n \in L(A)$

Ca un ultim pas, aplicăm homomorfisumul π care uită starea, anume $\pi : (q,a) \mapsto a, \pi : a \mapsto a$, $\forall a \in V$ şi $\pi : \# \mapsto \#$

Atunci $L(M) = fr(\pi(K))$, deci *un limbaj context-senzitiv este frontiera unui limbaj de griduri recunoscut*.

Limbaje 2-dimensionale

- Intre diferitele clase de limbaje 2-dimensionale avem relaţiile

$$4\text{DFA} \subset 4\text{NFA} \subset \text{FIS} = \text{sisteme de pavări}$$

  incluziunile fiind stricte

- Se ştie că proprietatea dacă *un limbaj context-senzitiv este ori nu vid* este *nedecidabilă*. Obţinem următorul rezultat important

Corolar: *Este nedecidabil dacă limbajul de griduri recunoscut un* FIS *este vid ori nu.*

Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- Generalităţi
- Sisteme interactive finite
- Sisteme de pavare (tiling systems)
- *Logică monadică de ordinul 2 pentru griduri*
  - *Logica EMSO*
  - De la FIS-uri la formule logice EMSO
  - De la formule logice EMSO la sisteme de pavare
- Concluzii, diverse, etc.

Griduri ca structuri logice

Griduri ca structuri logice: Un grid $p \in V^{\dagger\star}$ poate fi identificat cu o *stuctură* $\underline{p} = (dom(p), S_1, S_2, (P_a)_{a\in V})$, unde

- $dom(p) = \{1,\ldots,l_1(p)\} \times \{1,\ldots,l_2(p)\}$ ($l_1(p)/l_2(p)$ reprezintă numărul de linii/coloane din p)
- $S_1, S_2 \subseteq dom(p)^2$ reprezintă cei doi "*succesori*":
  – $(x,y) \in S_1$ înseamnă că x,y sunt poziţii vecine *vertical* şi x este deasupra lui y
  – $(x,y) \in S_2$ înseamnă că x,y sunt poziţii vecine *orizontal* şi x este la stânga lui y
- pentru $a \in V$, $P_a \subset dom(p)$ reprezintă *submulţimea poziţiilor* lui p în care se alfă litera a

Sintaxa

Rolul logicii: Submulţimile P_a vor fi specificate cu formule din logica monadică de ordinul 2 folosind:

– $x, y, \ldots$ variabile (de ordinul 1) pentru poziţiile din grid;
– $X, Y, \ldots$ variabile (de ordinul 2) pentru mulţimi de astfel de poziţii

Formule atomice:

– $x = y$ (specificând egalitatea lui x cu y)
– xS_iy cu $i = 1, 2$ (specificând faptul că x, y sunt vecini, anume vertical $(x, y) \in S_1$ ori orizontal $(x, y) \in S_2$),
– $X(x)$ (specificînd că x este în mulţimea X)
– $P_a(x)$ (specificând că în poziţia x este litera a, i.e. $x \in P_a$)

..Sintaxa

Formule: *Formulele* se construiesc din formule atomice folosind

– *conectori booleeni*: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

– *cunatificatori*: $\exists, \forall$ (aplicaţi atât variabileor de ordinul 1, cât şi celor de ordinul 2)

Propoziţiile sunt formule *fără variabile libere*.

Exemple:

– $\phi_{sus}(x) := \neg\exists y(yS_1x)$ (poziţia x este pe prima linie)

– $\phi_{dj}(x) := \neg\exists y(xS_1y) \wedge \neg\exists y(xS_2y)$ (poziţia x este colţul dreapta-jos)

– $\phi_1(x) := \neg\exists y(yS_1x) \wedge (P_a(x) \vee P_b(x))$ (x este pe prima linie şi conţine litera a sau b)

– $\phi_2 := \forall x\, \phi_{sus}(x)$ (gridurile au o singură linie)

Semantica / relaţia de satisfacere

Relaţia de satisfacere: Dat un grid p, submulţimi $Q_1,\ldots,Q_n \subseteq dom(p)$, şi o *formulă* ϕ având cel mult variabilele libere $X_1,\ldots,X_n$, spunem că

$$p \textit{ satisface } \phi \textit{ relativ la interpretarea } Q_1,\ldots,Q_n \subseteq dom(p)$$

şi scriem $(\underline{p}, Q_1,\ldots,Q_n) \models \phi(X_1,\ldots,X_n)$ dacă

structura logică asociată $\underline{p}$ satisface ϕ,
unde $X_1,\ldots,X_n$ se înlocuiesc cu $Q_1,\ldots,Q_n$.

Pentru o *propoziţie* scriem simplu $\underline{p} \models \phi$.

Limbajul $L(\phi)$ specificat de o formulă ϕ este mulţimea gridurilor care satisfac ϕ; formal $L(\phi) = \{p \in V^{\dagger\star} : \underline{p} \models \phi\}$.

Limbaje logic-definibile

Un limbaj de griduri L este:

- *definibil în logica monadică de ordinul 2* (ori *MSO definibil*) dacă există o formulă ϕ în logica monadică de ordinul 2 cu $L = L(\phi)$;

- *definibil în logica de ordinul 1* (ori *FO definibil*) dacă există o formulă ϕ în logica de ordinul 1 cu $L = L(\phi)$;

- *definibil în logica existenţială monadică de ordinul 2* (ori *EMSO definibil*) dacă există o formulă ϕ cu $L = L(\phi)$, unde

$$\phi = \exists X_1 \ldots \exists X_n \, \psi(X_1, \ldots, X_n)$$

şi ψ conţine numai cuantificatori de variabile de ordinul 1.

Exemplu: pătrate de a-uri

Exemplu pătrate de a-uri: Prezentăm o formulă EMSO pentru pătrate de a-uri. Intâi câteva notaţii:

– $\phi_{sus}(x)$ (resp. $\phi_{jos}(x), \phi_{st}(x), \phi_{dr}(x)$) sunt formule pentru a specifica că x este pe *bordul de sus* (resp. *jos*, *stânga*, *dreapta*)

– $\phi_{ss}(x)$ (resp. $\phi_{sj}(x), \phi_{ds}(x), \phi_{dj}(x)$) sunt formule pentru a specifica că x este *colţul stânga-sus* (resp. *stânga-jos*, *dreapta-sus*, *dreapta-jos*)

– Exemple: $\phi_{st}(x) := \neg\exists y(yS_2x)$; $\phi_{dj}(x) := \phi_{dr} \wedge \phi_{jos}$; etc.

Formula este ψ_2, unde:

$$\begin{aligned}\psi_1 := \exists X(&\exists x(X(x) \wedge \phi_{ss}(x))\\ &\wedge \forall x \forall y \forall z((X(x) \wedge xS_1y \wedge yS_2z) \to X(z))\\ &\wedge \forall x(X(x) \wedge (\phi_{jos}(x) \vee \phi_{dr}(x)) \to \phi_{dj}(x)))\end{aligned}$$

$$\psi_2 := \psi_1 \wedge \forall x\, P_a(x)$$

..Exemplu: pătrate de *a*-uri

Explicaţie:

- ψ_1 arată că există o submulţime X de poziţii care definesc diagonala principală; anume:
  – (1) X conţine colţul din stânga-sus;
  – (2) mergând în jos, apoi în dreapta rămân pe diagonală;
  – (3) când ating marginea de jos ori din dreapta, de fapt ajung în colţul din dreapta-jos
- ψ_2 adaugă faptul că orice poziţie conţine litera a

Variaţie:

- Descrieţi o formulă pentru pătratele cu a pe diagonală, b în dreapta-sus, şi c în stînga-jos.

Teorema fundamentală (enunţ)

Teorema fundamentală:

Teoremă: Un limbaj de griduri este EMSO definibil dnd este acceptat de un FIS.

Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- Generalităţi
- Sisteme interactive finite
- Sisteme de pavare (tiling systems)
- *Logică monadică de ordinul 2 pentru griduri*
  - Logica EMSO
  - *De la* FIS*-uri la formule logice EMSO*
  - De la formule logice EMSO la sisteme de pavare
- Concluzii, diverse, etc.

De la FIS-uri la formule logice

Incepem cu implicaţia

Teoremă: Un limbaj de griduri acceptat de un FIS *F este EMSO definibil.*

Dem: Modificăm puţin reprezentarea scenariilor lui F introducând informaţia despre stări/clase în celulele:

- o celulă într-un grid peste noul alfabet $\widehat{V}$ conţine un 5-uplu (s,A,a,B,t) dacă regula aplicată pentru a este $\begin{array}{|c|c|c|}\cline{2-2} \multicolumn{1}{c|}{} & s & \multicolumn{1}{c}{} \\ \hline A & a & B \\ \hline \multicolumn{1}{c|}{} & t & \multicolumn{1}{c}{} \\ \cline{2-2}\end{array}$;
- un grid p este acceptat de F dacă există un grid $\widehat{p}$ peste alfabetul extins care: (1) extinde pe p (restrcţia lui $\widehat{p}$ la a 3-a componenta dă p), (2) în care clasele/stările celulelor vecine sunt egale, şi (3) pe bord au clase/stări iniţiale/finale.

..De la FIS-uri la formule logice

Să zicem că o poziţie x conţine $(s_x, A_x, a_x, B_x, t_x)$. Putem găsi formule pentru următoatele proprietăţi care caracterizează scenariile de acceptare:

- dacă poziţia x este pe linia de sus atunci s_x este stare iniţială; analog pentru bordurile din dreapta, stânga, jos
- dacă x este vecinul din stînga al lui y, atunci $B_x = A_y$
- dacă x este vecinul din sus al lui y, atunci $t_x = s_y$

Problema se complică puţin căci nu avem predicate pentru alfabetul extins, ci doar P_a-urile iniţiale. Folosim următoarele variabile:

– dacă stările sunt $s_1, \ldots, s_k$, variabilele $N_{s_1}, \ldots, N_{s_k}$ vor specifica mulţimile de poziţii care conţin în nord (pe prima componentă) stările $s_1, \ldots, s_k$, respectiv;

– analog avem $W_{A_i}, E_{A_i}, S_{s_i}$ pentru direcţiile de vest, est, sud.

..De la FIS-uri la formule logice

Construim formule pentru:

Orice poziţie conţine exact o stare în nord/sud şi exact o clasă în vest/est: Fie F_s/F_c mulţimea stărilor/claselor lui F. Formula N pentru nord este

$$N := \forall x((\vee_{s\in F_s} N_s(x)) \wedge \wedge_{s,t\in F_s, s\neq t} \neg(N_s(x) \wedge N_t(x)))$$

Analog găsim formule $W/E/S$ pentru vest/est/sud.

Poziţiile de pe bord au stări/clase iniţiale/finale: Fie F_{si}/F_{sf} submulţimile de stări iniţiale/finale; F_{ci}/F_{cf} submulţimile de clase iniţiale/finale. Formula BN pentru bordul de nord este

$$BN = \forall x(\phi_{sus}(x) \rightarrow (\vee_{s\in F_{si}} N_s(x)))$$

Analog avem formulele $BW/BE/BS$ pentru bordurile vest/est/sud.

..De la FIS-uri la formule logice

Fiecare poziţie reprezintă o tranziţie validă: Fie F_t mulţimea tranziţiilor din F; reprezentăm o tranziţie t prin $(t_n, t_w, t_c, t_e, t_s)$ (componentele de nord, vest, centru, est, sud). Formula T pentru tranziţii este

$$T := \forall x(\vee_{t\in F_t}(N_{t_n}(x)\wedge W_{t_w}(x)\wedge P_{t_c}(x)\wedge E_{t_e}(x)\wedge S_{t_s}(x)))$$

Formula finală:

$$\phi(F) := \exists_{s\in F_s} N_s\ \exists_{A\in F_c} W_A\ \exists_{A\in F_c} E_A\ \exists_{s\in F_s} S_s \\ (T\wedge N\wedge W\wedge E\wedge S\wedge BN\wedge BW\wedge BE\wedge BS)$$

Din construcţie, se vede că formula capturează scenariile lui F, i.e., există un scenariu pentru p dnd există submulţimi pentru distribuţia stărilor/claselor lui F într-un grid extins $\widehat{p}$ satisfăcând $\phi(F)$.

In concluzie, $L(F) = L(\phi(F))$. $\square$

Sisteme interactive finite; Limbaje 2-dimensionale

Cuprins:

- Generalităţi
- Sisteme interactive finite
- Sisteme de pavare (tiling systems)
- *Logică monadică de ordinul 2 pentru griduri*
  - Logica EMSO
  - De la FIS-uri la formule logice EMSO
  - *De la formule logice EMSO la sisteme de pavare*
- Concluzii, diverse, etc.

De la formule logice la sisteme de pavare

Conexiunea se va face prin sisteme de pavare: ele specifică proiecţii de limbaje locale şi sunt echivalente cu FIS-urile. Vom demonstra următoarele fapte:

Lema 1: *Un limbaj EMSO-definibil este proiecţia unui limbaj FO-definibil.*

Lema 2: *Un limbaj FO-definibil este local testabil cu prag.*

(Actalmente, *un limbaj este FO-definibil* dnd *este local testabil cu prag*, dar nouă ne trebuie doar implicaţia din enuţul de mai sus; celălaltă implicaţie e simplă, similară cu Teorema din Slide 4.12.)

Lema 3: *Un limbaj local testabil cu prag este proiecţia unui limbaj local, deci reprezentabil cu un sistem de pavare.*

Limbaje local testabile cu prag

- Fie $t \geq 1$ prag şi $d \geq 1$ dimensiune de grid. Două griduri p_1, p_2 sunt *$\sim_{d,t}$-echivalente* dacă fiecare grid σ de dimensiune cel mult $d \times d$

  – fie apare mai mult de t ori în ambele griduri extinse cu # $\widehat{p_1}$ şi $\widehat{p_2}$
  – fie numărul de apariţii ($\leq t$) este acelaşi în $\widehat{p_1}$ şi $\widehat{p_2}$.

- *$\simeq_{d,t}$-echivalenţa* se referă la cazul când se folosesc doar *pătrate $d \times d$*.

..Limbaje local testabile cu prag

- Un limbaj:

  – este *local d-testabil cu prag t* dacă este reuniune de $\sim_{d,t}$-clase de echivalenţă;
  – este *local d-testabil cu prag* dacă pentru un prag t este local d-testabil cu prag t;
  – în fine, este *local testabil cu prag* dacă pentru o dimensiune d este local d-testabil cu prag.

- Dacă înlocuim $\sim$ cu $\simeq$ (deci folosim teste doar cu pătrate de dimensiune fixă) obţinem limbaje *locale strict testabile cu prag.*

De la EMSO la FO definibilitate

Lema 1: Un limbaj EMSO-definibil este proiecţia unui limbaj FO-definibil.

Dem: Fie $L \subseteq V^{\dagger\star}$ limbaj EMSO-definibil:

- Există o formulă $\phi := \exists X_1 \ldots \exists X_k \quad \psi(X_1,\ldots,X_k)$ cu $\psi(X_1,\ldots,X_k)$ de ordinul 1 şi $L = L(\phi)$; un grid p este în L dnd există $Q_1,\ldots,Q_k \subseteq dom(p)$ şi $(\underline{p},Q_1,\ldots,Q_k)$ satisface $\psi(X_1,\ldots,X_k)$.
- Folosim un alfabet extins $\overline{V} = V \times \{0,1\}^k$ unde a i-a componentă adiţională este 1 pe o poziţie x dnd $x \in Q_i$; fie π proiecţia pe prima componentă (pe V).

..De la EMSO la FO definibilitate

- Limbajul $\overline{L} \subseteq \overline{V}^{\dagger\star}$ definit de $\psi(X_1, \ldots, X_k)$ este FO-definibil, iar $\pi(\overline{L}) = L$.

  Notă: Trebuie ceva atenţie la interpretarea lui $P_a(x)$ şi $X_i(x)$ - ambele rezultă din proiecţii (pe prima, resp. a $i+1$-a componentă) folosind P_a-urile pe alfabetul extins, anume $(P_{\overline{a}})_{\overline{a}\in\overline{V}}$:
  - $P_a(x) \mapsto \bigvee_{\alpha\in\{0,1\}^k} P_{(a,\alpha)}(x)$;
  - $X_i(x) \mapsto \bigvee_{a\in V,\alpha'\in\{0,1\}^{i-1},\alpha''\in\{0,1\}^{k-i}}, P_{(a,\alpha',1,\alpha'')}(x)$; $\square$

De la FO la local testabilitate cu prag

Lema 2: Un limbaj FO-definibil este local testabil cu prag.

Dem. (schiţă):

- Două griduri (privite ca modele) $\underline{p}, \underline{q}$ sunt *n-echivalente* (scris $\underline{p} \equiv_n \underline{q}$) dacă satisfac aceleaşi formule de ordinul 1 cu ordin de cuantificare $\leq n$ (i.e., avînd cel mult n cuantificatori cuibăriţi).

- Relaţia $\equiv_n$ este relaţie de echivalenţă şi limbajul generat de o formulă de ordinul 1 cu ordin de cuantificare $\leq n$ este reuniune de $\equiv_n$-clase.

- In concluzie, pentru Lema 2, este suficient să arătăm că pentru orice $\equiv_n$-clasă C limbajul $\{p | \underline{p} \in C\}$ este local testabil cu prag.

..De la FO la local testabilitate cu prag

Cu comentariile de mai sus, Lema 2 rezultă din următoarul rezultat:

Lema 2': Pentru orice $n \geq 0$, există d, t, anume $d = 2 \cdot 3^n + 1$ şi $t = n \cdot 3^{2n}$, astfel că $p \sim_{d,t} q$ implică $\underline{p} \equiv_n \underline{q}$.

Demonstraţia se bazează pe sisteme dus-întors ("back-and-forth") în sensul lui Ehrenfeucht & Fraïssé. Cum detaliile nu au prea multă relevanţă pentru scopul cursului, le omitem (vezi Giammarresi, Restivo, Seibert, Thomas: *Information and computation* 125(1996):32-45). $\Box$.

De la local testabilitate cu prag la FIS-uri

Lema 3: Un limbaj local testabil cu prag este proiecţia unui limbaj local.

Dem: Demonstraţia constă în doi paşi:

- simulăm calculul apariţiilor subblocurilor $\leq d \times d$ cu un sistem de pavare $d \times d$;
- reducem sistemul de pavare $d \times d$ la unul 2×2

Ultima parte o omitem (se rezumă la o codare adecvată ca în Exerciţiul 4.4).

Pentru prima parte, limbajul se descompune într-o reuniune finită $L = L_0 \cup L_1 \cup \ldots \cup L_{d-2}$ de limbaje L_i în care subblocurile de testat au o dimensiune fixă $(i+2) \times (i+2)$. (Cu ajutorul gridurilor pătrate $j \times j$ se pot manipula şi gridurile $j \times m$ ori $m \times j$ cu $m \geq j$.)

..De la local testabilitate cu prag la FIS-uri

Deci este suficient să demonstrez lema pentru aceste ultime limbaje locale strict d-testabile cu prag, cu $d \geq 3$.

- Fie L limbaj local strict testabil cu prag, deci reuniune de $\simeq_{d,t}$-clase pentru un prag t; fie $k = (|V \cup \{\#\}|)^{d^2}$ şi $\sigma_1, \ldots, \sigma_k$ toate gridurile peste $V \cup \{\#\}$ de dimensiune $d \times d$.

- Fiecare $\simeq_{d,t}$-clasă se poate reprezenta cu un tuplu $(t_1, \ldots, t_k)$ cu $0 \leq t_i \leq t$, unde t_i reprezintă numărul de apariţii al gridului σ_i (redus la t, dacă depăşeşte acest număr).

- Vom defini un limbaj local L' peste un alfabet extins $V \times C$, a cărei proiecţie pe V este L.

..De la local testabilitate cu prag la FIS-uri

- Idee:
  - Dat un grid p, scanăm p cu o fereastră $d \times d$ şi numărăm de câte ori apare fiecare σ_i în p.
  - Scanarea se face orizontal pe fâşii de înălţime d, explorate de la stânga la dreapta; contorizăm rezultatul în partea adiţională a etichetelor (din Σ).
  - Scanăm apoi gridurile din extrema dreaptă de sus în jos şi adunăm rezultatele.
  - In final, $p \in L$ dnd procedura produce un k-uplu acceptabil în colţul dreapta-jos.

..De la local testabilitate cu prag la FIS-uri

- Formalizare (să notăm că poziţia aleasă pentru numărare este $(d-1,d-1)$; ea nu este #, căci $d \geq 3$):
  - Fie $C = \{(t_1,\ldots,t_k) | 0 \leq t_i \leq t\}$; limbajul L' va fi definit peste $V \times C$
  - Gridurile de pavare $\Delta^{(d)}$ au ca elemente generice perechi $\sigma \times c$ cu $\sigma \in (V \cup \{\#\})^{d \times d}$ şi $c \in (C \cup \{\#\})^{d \times d}$ care sunt consonante pe # (dacă o poziţie este # în σ este # şi în c şi vice-versa).
  - A. Pentru gridurile din *centru, sus, şi jos* de forma $\sigma_i \times c$, cerem ca $c(d-1,d-1) = c(d-1,d-2) + (0,\ldots,1,\ldots,0)$, ultima cu 1 pe poziţia i; grilele de mai sus actualizează contorii trecînd cu o poziţie la dreapta;

..De la local testabilitate cu prag la FIS-uri

- B. Gridurile pentru *stânga, stînga-sus, şi stânga-jos* se folosesc pentru iniţializare: poziţia $(d-1,d-1)$ a lui $\sigma_i \times c$ este $(0,\ldots,1,\ldots,0)$, cu 1 pe poziţia i;
- C. Gridurile din *dreapta* conţin sume parţiale, deci combinăm suma pe orizontală cu cea pe verticală: dacă gridul este $\sigma_i \times c$, atunci $c(d-1,d-1) = c(d-2,d-1) + c(d-1,d-2) + (0,\ldots,1,\ldots,0)$, ultima cu 1 pe poziţia i;
- D. Gridurile din *dreapta-sus* iniţializează cu zero suma pe verticală; deci sunt la fel ca cele din grupul A;
- E. Gridurile din *dreapta-jos* calculează suma finală – formal sunt la fel ca cele din grupul C; în plus, ele conţin *condiţia de acceptare*, anume restricţia ca $c(d-1,d-1)$ să conţină doar k-tuplurile admisibile.

..De la local testabilitate cu prag la FIS-uri

- Cu maşina de numărare definită mai sus, se vede uşor că limbajul L' definit astfel conţine un grid p' dnd proiecţia pe prima componentă $\pi(p')$ este în L. $\square$

Cu aceasta este demonstrată următoarea:

Teorema: Un limbaj EMSO-definibil se poate specifica cu un sistem de pavare.

Stim că FIS-urile şi sistemele de pavare sunt echivalente; deci obţinem corolarul:

Corolar: Un limbaj EMSO-definibil se poate specifica cu un FIS.

Echivalenţa: logica EMSO $\Leftrightarrow$ FIS-uri

Combinând rezultatele din secţiunile precedente, obţinem următoarul rezultat:

Teorema fundamentală:

Teoremă: Un limbaj de griduri este EMSO-definibil dnd este acceptat de un FIS. $\square$