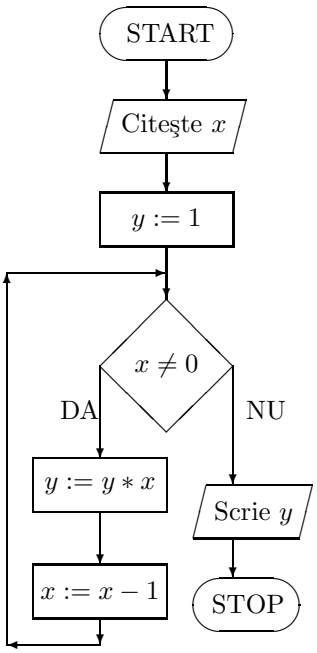
1. Puţină ISTORIE privind conceptul de PROGRAM şi SEMANTICA lui

Virgil Emil Căzănescu

February 14, 2010

Vom începe cu un exemplu de schemă logică şi anume schema logică pentru factorial



1 Programe abstracte în stil Ianov

Observăm că datele la care se referă schema logică de mai sus sunt numere întregi. Întrebîndu-ne care sunt elementele specifice numerelor întregi care intervin în construirea schemei, observăm că sunt utilizate constantele 0 şi 1, operaţia binară de înmulţire şi operaţia unară de scădere a unităţii.

Abstractizăm înlocuind:

- mulţimea numerelor întregi cu o mulţime arbitrară D,
- constantele 0 şi 1 cu două elemente a şi b din D,
- operaţia de înmulţire cu o operaţie binară f pe mulţimea D,
- scăderea unităţii cu o operaţie unară g pe mulţimea D.

Obţinem în acest fel *programul abstract* din Figura 1. Pentru orice număr natural n notăm

$$[\, n \,] = \{1, 2, 3, \ldots, n\}.$$

Schema logică a factorialului se obţine din programul abstract din Figura 1 interpretînd în mod adecvat simbolurile: a, b, f şi g.

Vom prefera să dăm acestora o altă interpretare:

- $D = A^*$ cu A mulţime arbitrară,

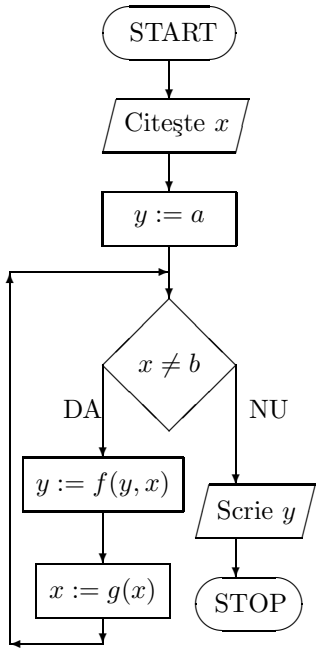


Figure 1: Program abstract

- a şi b sunt şirul de lungime nulă(cuvîntul vid),
- $g(a_1a_2 \dots a_n) = a_2a_3 \dots a_n$ unde $(\forall i \in [\, n\,])a_i \in A$,
- $f(y, a_1a_2 \dots a_n) = a_1y$ unde $y \in A^*$ şi $(\forall i \in [\, n\,])a_i \in A$.

Prin această interpretare programul abstract de mai sus se transformă într-o schemă logică a programului care primind la intrare $x = a_1a_2 \dots a_n$ furnizează la ieşire $y = a_n \dots a_2a_1$ unde $(\forall i \in [\, n\,])a_i \in A$.

Un program abstract reprezintă de fapt o familie de programe, fiecare dintre acestea obţinându-se printr-o interpretare. *Un program este format dintr-un program abstract şi o interpretare.*

În continuare ne vom ocupa de conţinutul căsuţelor(chenarelor) din figură care includ atribuiri sau teste. Pentru definirea noţiunii de atribuire şi test este preferabil să ne situăm la acest *prim nivel de abstractizare*, unde uitlizând un limbaj de ordinul întâi(adică al calculului cu predicate) se vede că o atribuire este formată dintr-o variabilă individuală şi un termen, iar un test este o formulă fără cuantificatori. Vom prefera să nu utilizăm limbajul din calculul cu predicate ci să utilizăm limbajul folosit de informaticieni redefinind, pentru claritate, toate noţiunile de care avem nevoie.

1.1 Sintaxă

Fie *Id* o mulţime ale cărei elemente le numim *identificatori*, O o mulţime ale cărei elemente le numim *simboluri de operaţii*, R o mulţime ale cărei elemente le numim *simboluri de relaţii* şi două funcţii cu valori numere naturale

$$a : O \longrightarrow \omega \quad \text{şi} \quad c : R \longrightarrow \omega.$$

Fie Σ signatura cu două sorturi $\{e, b\}$ unul pentru valori numerice şi altul pentru valori logice: e provine de la expresie şi b de la boolean. Operaţiile şi rangul lor sunt:

- $o : e^{a(o)} \longrightarrow e$ pentru orice $o \in O$,
- $p : e^{c(p)} \longrightarrow b$ pentru orice $p \in R$,
- $\wedge : bb \longrightarrow b$ este conjucţia şi $\neg : b \longrightarrow b$ este negaţia.

Σ-algebra liber generată de $\{Id, \emptyset\}$ este notată cu $T_\Sigma(Id)$. Observăm că identificatorii sunt variabile de sort e şi că nu am introdus variabile de sort b. Elementele lui $T_\Sigma(Id)_e$ le numim **expresii** şi elementele lui $T_\Sigma(Id)_b$ le numim **teste**. Fie

$$\mathbf{Atrib} = \{\langle x,\, e\rangle \;\mid\; x \in Id \;\text{ şi }\; e \in T_\Sigma(Id)_e\}.$$

Elementele lui **Atrib** le numim *atribuiri*.

1.2 Semantică

Pentru **interpretarea testelor** considerăm algebra valorilor logice cu suportul $[\,2\,]$, 1 pentru *adevăr* şi 2 pentru *fals*, precum şi operaţiile $\wedge_b$ şi $\neg_b$ definite prin

$$f \wedge_b g \;=\; \begin{cases} 1 & \text{dacă } f = g = 1 \\ 2 & \text{dacă } f = 2 \text{ sau } g = 2 \end{cases}$$

$$\neg_b f \;=\; \begin{cases} 1 & \text{dacă } f = 2 \\ 2 & \text{dacă } f = 1 \end{cases}$$

Definiţie 1 *O* **interpretare** *este pur şi simplu o Σ-algebră*

$$\mathcal{I} = \langle D, [\,2\,], \{o_D\}_{o\in O}, \{p_D\}_{p\in R}, \wedge_b, \neg_b\rangle,$$

în care partea de logică este fixată.

D este o mulţime, pentru orice $o \in O$ şi orice $p \in R$

$$o_D\colon D^{a(o)} \longrightarrow D \text{ şi } p_D\colon D^{c(p)} \longrightarrow [\,2\,]$$

sunt funcţii.

Interpretarea dă semnificaţie numai simbolurilor de operaţii şi simbolurilor de relaţii. Arătăm în continuare cum interpretăm(dăm semnificaţie) celorlalte elemente care pot apare într-un program abstract.

Înainte de a trece mai departe vom da câteva explicaţii intuitive privind interpretarea expresiilor. Mulţimea D este mulţimea de date cu care se fac calculele. O funcţie $s\colon Id \longrightarrow D$ arată valorile pe care le au identificatorii într-un anumit moment, adică o **stare a memoriei**. Dacă $s \in D^{Id}$ şi $x \in Id$ atunci $s(x)$ este valoarea identificatorului x în starea s a memoriei. Mulţimea D^{Id}, a funcţiilor de la Id la D, este numită **mulţimea stărilor memoriei** şi va fi notată cu S.

Pentru orice stare a memoriei $s \in D^{Id}$ notăm cu

$$s^{\#}\colon T_\Sigma(Id) \longrightarrow \mathcal{I}$$

unicul morfism de Σ-algebre care-l extinde pe s.

Funcţia

$$\mathcal{S}\colon T_\Sigma(Id)_e \longrightarrow D^S$$

dă **semnificaţia expresiilor**.

Pentru orice expresie $E \in T_\Sigma(Id)_e$, funcţia

$$\mathcal{S}(E)\colon D^{Id} \longrightarrow D$$

definită prin $\mathcal{S}(E)(s) = s^{\#}_e(E)$ dă pentru orice stare $s \in D^{Id}$ valoarea $s^{\#}_e(E)$ a expresiei E calculată în starea s.

1.2.1 Semantica atribuirilor

Ca orice instrucţiune cu o ieşire, atribuirile trebuiesc interpretate ca funcţii parţiale de la mulţimea stărilor la ea însăşi. Prin urmare definim o funcţie

$$\mathcal{V}\colon \mathbf{Atrib} \longrightarrow S^S.$$

Dacă $x \in Id$ şi $e \in E$ atunci funcţia parţială

$$\mathcal{V}(x,e)\colon S \longrightarrow S$$

este definită pentru orice $s \in D^{Id}$ şi $y \in Id$ prin

$$\mathcal{V}(x,e)(s)(y) = \begin{cases} s(y) & \text{dacă } y \neq x \\ \mathcal{S}(e)(s) & \text{dacă } y = x. \end{cases}$$

În egalitatea de mai sus, s reprezintă starea memoriei la începutul execuţiei atribuirii (x,e) şi $\mathcal{V}(x,e)(s)$ este starea memoriei la terminarea execuţiei atribuirii.

1.2.2 Semantica testelor

Pentru orice test $t \in T_\Sigma(Id)_b$ funcţia

$$\mathcal{V}'(t)\colon D^{Id} \longrightarrow [\,2\,]$$

de evaluare a testului, definită pentru orice $s \in S$ prin $\mathcal{V}'(t)(s) = s_b^{\#}(t)$ ne arată pentru orice stare s, concluzia testării şi anume $\mathcal{V}'(t)(s) = 1$ arată că testul este adevărat în s iar $\mathcal{V}'(t)(s) = 2$ arată că testul este fals în s.

2 Spre al doilea nivel de abstractizare

După ce am terminat de interpretat elementele care apar într-un program abstract observăm neconcordanţa dintre modul de tratare a acestora. Pentru a putea trece la al doilea nivel de abstractizare este necesar să uniformizăm modul de tratare a atribuirilor şi testelor.

În mare un program abstract este un graf orientat şi etichetat.

Observăm că din fiecare nod care este etichetat cu o instrucţiune de atribuire pleacă o singură săgeată, iar din fiecare nod care este etichetat cu un test pleacă două săgeţi. Numărul săgeţilor care pleacă dintr-un nod etichetat cu o instrucţiune trebuie să fie în concordanţă cu **numărul ieşirilor instrucţiunii** care etichetează nodul. Menţionăm că deşi în modelul nostru nu acceptăm decît instrucţiuni cu o ieşire(atribuirile) şi cu două ieşiri(testele), practica programării ne dovedeşte existenţa instrucţiunilor cu număr arbitrar de ieşiri. Ca exemplu de instrucţiune cu trei ieşiri putem menţiona IF-ul aritmetic din FORTRAN. Notăm cu ω mulţimea numerelor naturale şi cu

$$r : \mathbf{Atrib} \cup \mathbf{Teste} \longrightarrow \omega$$

funcţia **ieşire** care ia valoarea 1 pentru orice atribuire şi 2 pentru orice test.

Observăm că cele două săgeţi care pleacă dintr-un nod etichetat cu un test, sunt etichetate cu DA sau NU pentru a menţiona modul de continuare al execuţiei în funcţie de valoarea de adevăr a testului(DA pentru adevărat, respectiv NU pentru fals). Pentru eliminarea etichetelor de pe săgeţi este preferabil să considerăm că cele două săgeţi care pleacă dintr-un nod etichetat cu un test sunt ordonate făcând, de exemplu, alegerea ca prima să fie săgeata etichetată cu DA, a doua fiind săgeata etichetată cu NU.

Notaţia valorilor de adevăr este în concordanţă cu convenţia de mai sus adică valoarea de adevăr 1 corespunde adevărului deoarece continuarea execuţiei pentru un rezultat adevărat al testului se face pe prima săgeată şi valoarea de adevăr 2 corespunde falsului deoarece continuarea execuţiei pentru un rezultat fals al testului se face pe a doua săgeată.

Astfel de convenţii trebuiesc făcute pentru toate instrucţiunile cu mai mult de două ieşiri, eliminându-se astfel marcajele de pe săgeţi. Cu alte cuvinte **mulţimea săgeţilor care pleacă dintr-un nod etichetat cu o instrucţiune este o mulţime total ordonată**. Pentru reprezentarea grafică a schemelor logice, ordonarea săgeţilor care pleacă din acelaşi nod se face de la stânga la dreapte.

Pentru fiecare atribuire sau test σ ne interesează în funcţie de starea în care este executat:

1. starea în care ajunge după execuţie,
2. după care dintre săgeţile care pleacă dintr-un nod etichetat cu σ trebuie continuată execuţia.

Prin urmare vom defini pentru orice $\sigma \in \mathbf{Atrib} \cup \mathbf{Teste}$ o funcţie parţială

$$\mathcal{M}(\sigma)\colon D^{Id} \multimap D^{Id} \times [\,r(\sigma)\,] \tag{1}$$

astfel încât pentru orice s şi s' din D^{Id} şi $j \in [\,r(\sigma)\,]$ egalitatea $\mathcal{M}(\sigma)(s) = (s', j)$ să aibă următoarea semnificaţie :

executând instrucţiunea σ în starea s, execuţia lui σ se termină în starea s' şi execuţia trebuie continuată după a j-a săgeată dintre cele $r(\sigma)$ săgeţi ce pleacă din nodul etichetat cu σ.

Pentru $\sigma \in \mathbf{Atrib}$ şi $s \in D^{Id}$ definim

$$\mathcal{M}(\sigma)(s) = (\mathcal{V}(\sigma)(s), 1).$$

Observăm că $\mathcal{M}(\sigma)(s)$ este definit dacă şi numai dacă $\mathcal{V}(\sigma)(s)$ este definit.

Pentru $\sigma \in \mathbf{Teste}$ şi $s \in D^{Id}$ definim

$$\mathcal{M}(\sigma)(s) = (s, \mathcal{V}'(\sigma)(s)).$$

Observăm că $\mathcal{M}(\sigma)(s)$ este definit dacă şi numai dacă $\mathcal{V}'(\sigma)(s)$ este definit.

Cu aceste definiţii am încheiat pregătirile necesare pentru a înţelege trecerea la *al doilea* nivel de abstractizare.

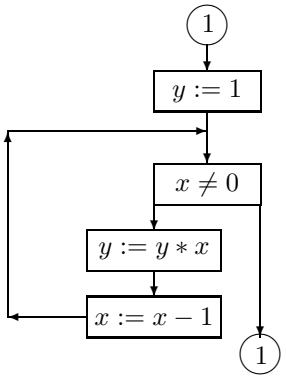


Figure 2: Program pentru $y := x$!

Abstractizarea se face atât sub aspect sintactic, adică privind programele, cât şi sub aspect semantic, adică privind interpretările.

În ceea ce priveşte sintaxa, înlocuim mulţimea $\mathbf{Atrib} \cup \mathbf{Teste}$ cu o mulţime arbitrară de instrucţiuni Σ pentru care este dată o funcţie de la Σ la mulţimea numerelor naturale care asociază fiecărei instrucţiuni numărul ei de ieşiri.

În ceea ce priveşte semantica, înlocuim mulţimea D^{Id} cu o mulţime arbitrară S iar interpretarea o dăm prin nişte funcţii de forma (1).

Definiţiile propriu zise le dăm în secţiunea următore, care abstracţie făcând de unele notaţii va fi independentă de motivarea care o precede.

Pentru ilustrarea ideilor de care ne-am ocupat, am lucrat cu programe care nu utilizează decît instrucţiuni de atribuire şi teste pentru ramificaţie. Eliminarea instrucţiunilor de citire-scriere, de exemplu, nu este o restricţie semnificativă deoarece putem alcătui programele astfel încît toate instrucţiunile de citire să fie făcute la început şi toate instrucţiunile de scriere să fie făcute la sfîrşit, urmînd ca noi să ne ocupăm de ceea ce se găseşte între acestea. O restricţie semnificativă inclusă aici este eliminarea procedurilor recursive.

Etichetele START şi STOP au o situaţie privilegiată. În general ne ocupăm de programe abstracte care pot reprezenta şi părţi de programe fapt pentru care vom admite posibilitatea mai multor etichete START. Presupunând că schema logică conţine mai multe etichete START, vom nota cu a(în exemplul nostru $a = 1$) numărul lor şi vom înlocui etichetele START cu numere de la 1 la a. Ele reprezintă *noduri de intrare* în program iar a va fi numit *numărul intrărilor programelor*. Nodurile etichetate cu STOP, deoarece ne ocupăm şi de programe abstracte care reprezintă părţi ale unor programe, pot reprezenta în afară de noduri în care execuţia se termină şi puncte de legătură cu alte programe. Vom nota cu b(în exemplul nostru $b = 1$) numărul nodurilor etichetate cu STOP, le vom numerota cu numerele de la 1 la b care vor înlocui etichetele STOP. Ele vor reprezenta *noduri de ieşire* iar b va fi numit *numărul ieşirilor programului*.

Din cele de mai sus rezultă că vom lucra cu scheme logice în care pot apare trei tipuri de noduri:

1. Noduri de intrare în care nu intră săgeţi şi din care pleacă o singură săgeată. Le numerotăm cu numere de la 1 la a.

2. Noduri *interne* etichetate cu instrucţiuni. Din fiecare ieşire de instrucţiune care etichetează un nod pleacă o singură săgeată, prin urmare din fiecare nod intern pleacă un număr de săgeţi egal cu numărul ieşirilor instrucţiunii care etichetează nodul.

3. Noduri de ieşire din care nu pleacă săgeţi. Le numerotăm cu numere de la 1 la b.

Mai menţionăm că fiecare săgeţă trebuie să ajungă la o ieşire a programului sau la una dintre intrările instrucţiunilor care etichetează nodurile interne.

Transformând schema logică de mai sus după aceste idei obţinem Figura 2.

3 Programe deterministe în sens Elgot

Fie Σ o mulţime. Chiar dacă elementele lui Σ sunt numite instrucţiuni, ele sunt de fapt variabile care pot reprezenta fie instrucţiuni, fie programe. Deoarece un program poate avea mai multe intrări vom presupune acelaşi lucru şi despre elementele lui Σ. Fie

$$i : \Sigma \longrightarrow \omega \text{ şi } o : \Sigma \longrightarrow \omega$$

două funcţii. Pentru o instrucţiune $\sigma \in \Sigma$, $i(\sigma)$ dă numărul ei de intrări şi $o(\sigma)$ dă numărul ei de ieşiri. Trebuie făcută o distincţie netă între intrările, respectiv ieşirile programului şi intrările, respectiv ieşirile unei instrucţiuni σ. Mai menţionăm că săgeţile care ajung într-un nod intern trebuie să ajungă pe o anumită intrare a instrucţiunii care etichetează nodul.

Până acum ne-am ocupat de programele numite în literatura de specialitate **complet deterministe**. Pentru a explica necesitatea trecerii la clasa mai largă a programelor **deterministe** vom vorbi puţin despre feedback.

Deşi feedbackul nu apăruse pe vremea lui C.C.Elgot, deşi în titlul acestui text am inclus cuvântul istorie nu ne permitem să scriem un text atât de demodat încât să mai modelăm ciclarea prin operaţia de iterată aşa cum făcea Elgot.

Vom utiliza la început feedbackul unar la stânga. El se poate aplica numai programelor care au cel puţin o intrare şi o ieşire. După o feedbackare(aplicarea feedbackului) unar numărul intrărilor şi al ieşirilor programului ca şi numărul fiecărei intrări sau ieşiri scade cu câte o unitate.

Feedbacul la stânga se efectuează eliminând prima intrare şi prima ieşire, redirijând săgeţile care se îndreptau către prima ieşire spre locul în care era dirijată prima intrare. O problemă apare dacă prima intrare se făcea chiar pe prima ieşire căci săgeţile care mergeau la prima ieşire nu mai au unde să fie redirijate. În acest caz ele nu mai sunt redirijate ci pur şi simplu sunt şterse, fapt care face ca rezultatul feedbackului să iasă din clasa programelor de care ne-am ocupat pâna acum denumite în literatura de specialitate programe deterministe complete.

Aici este un punct cheie în dezvoltarea acestei teorii. Operaţiile parţiale nu sunt dorite. Ele pot fi studiate dar acceptarea lor ca instrument de baza a fost de mult abandonată. O altă idee care s-a dovedit neproductivă a fost încercarea de a completa în mod artificial definiţia de mai sus pentru ca operaţia să devină peste tot definită. Singurul punct de vedere pe care-l considerăm adecvat pentru dezvoltarea acestui studiu este înlocuirea clasei programelor complet deterministe cu clasa mai largă a programelor deterministe. Clasa este lărgită prin impunerea de condiţii mai slabe asupra mulţimii săgeţilor programului:

din fiecare intrare a programului sau din fiecare ieşire a unei instrucţiuni care etichetează nodurile interne pleacă cel mult o săgeată(în cazul complet determinist trebuia să plece exact o săgeată).

Din punct de vedere semantic lipsa unei săgeţi de-alungul căreia să se continue execuţia este interpretată ca o intrerupere a programului care nu se mai termină.

Definiţie 2 *O* interpretare *a lui* Σ *este formată dintr-o mulţime* S *şi câte o funcţie parţială*

$$\mathcal{I}(\sigma) : S \times [i(\sigma)] \multimap S \times [o(\sigma)]$$

pentru fiecare $\sigma \in \Sigma$.

Mulţimea S reprezintă mulţimea stărilor calculatorului care execută progamele. Pentru $\sigma \in \Sigma$, $s, t \in S$, $n \in [i(\sigma)]$ şi $m \in [o(\sigma)]$ egalitatea

$$\mathcal{I}(\sigma)(s, n) = (t, m)$$

are următoarea semnificaţie:

începând execuţia instrucţiunii σ *de la intrarea* n *a ei şi din starea* s *a memoriei, execuţia se termină în starea* t *a memoriei la ieşirea* m *a lui* σ.

Vom nota cu $PFn(S)$ modelul semantic fundamental pentru programe deterministe. S este mulţimea stărilor maşinii care execută programele. $PFn(S)$ este o categorie având numerele naturale ca obiecte. Pentru două numere naturale a şi b, $PFn(S)(a, b)$ este prin definiţie mulţimea funcţiilor parţiale de la $S \times [a]$ la $S \times [b]$.

Dacă P este un program determinist cu a intrări şi b ieşiri vom nota semantica lui prin

$$\mathcal{S}m(P) \in PFn(S)(a, b).$$

Vom prezenta semantica unui program în două variante. Prima variantă se referă la cazul general al unui program sub forma descrisă mai sus a cărui semantică operaţională este dată în secţiunea următoare. A doua variantă prezentată mai târziu se referă la programe “structurate”.

3.1 Semantica operaţională

Vom nota cu E funcţia de etichetare a nodurilor interne, prin urmare dacă n este un nod intern atunci $E(n)$ este instrucţiunea care-l etichetează.

Fie P un program cu a intrări şi b ieşiri pentru care ştim semantica instrucţiunilor care-l compun. Vom defini semantica sa

$$\mathcal{S}m(P) : S \times [a] \nrightarrow S \times [b].$$

Fie $s_1, s \in S$, $i_0 \in [a]$ şi $h \in [b]$. Prin definiţie

$$\mathcal{S}m(P)(s_1, i_0) = (s, h)$$

dacă şi numai dacă

1) există o săgeată de la intrarea i_0 a programului la ieşirea h a programului şi $s = s_1$ SAU

2) există $k \geq 1$ şi un şir finit

$$(n_1, i_1, s_1, r_1)\ (n_2, i_2, s_2, r_2)\ \ldots\ (n_k, i_k, s_k, r_k)$$

unde n_j este un nod intern, $i_j \in [i(E(n_j))]$, $r_j \in [o(E(n_j))]$ şi $s_j \in S$ pentru orice $j \in [k]$ cu proprietăţile:

1. există o săgeată de la intrarea i_0 a programului
   la intrarea i_1 a instrucţiuni care etichetează nodul n_1,
2. $\mathcal{I}(E(n_j))(s_j, i_j) = (s_{j+1}, r_j)$ pentru orice $j \in [k-1]$ şi
   există o săgeată de la ieşirea r_j a instrucţiuni care etichetează nodul n_j
   la intrarea i_{j+1} a instrucţiuni care etichetează nodul n_{j+1} ŞI
3. $\mathcal{I}(E(n_k))(s_k, i_k) = (s, r_k)$ şi există
   o săgeată de la ieşirea r_k a instrucţiunii care etichetează nodul n_k la ieşirea h a programului.

Pentru o mai uşoară înţelegere a acestei definiţii menţionăm următoarele:
- k este numărul nodurilor interne prin care se trece în timpul execuţiei programului,
- $n_1, \ldots, n_k$ sunt nodurile interne prin care se trece în timpul execuţiei programului,
- i_j este numărul intrării de la care începe execuţia instrucţiunii care etichetează nodul n_j,
- s_j este starea memoriei de la care începe execuţia instrucţiunii care etichetează nodul n_j,
- r_j este numărul ieşirii la care se termină execuţia instrucţiunii care etichetează nodul n_j.

Trebuie să recunoaştem că este o definiţie cu care nu se poate lucra uşor, fapt pentru care s-au căuat alte soluţii. Un important pas înainte a fost făcut prin structurarea programelor. Din punct de vedere didactic considerăm că este util să prezentăm şi o definiţie a semanticii unui program bazată pe această idee, fapt pe care-l facem în cele ce urmează.

3.2 Semantica este un morfism

3.2.1 Compunerea

Compunerea între două programe este posibilă numai dacă numărul ieşirilor primului program este egal cu numărul intrărilor programului al doilea. Compunerea se face identificând fiecare ieşire a primului program cu intrarea cu acelaşi număr a programului al doilea şi desfiinţâdu-le ca noduri.

Fie P un program determinist cu a intrări şi b ieşiri şi Q un program determinist cu b intrări şi c ieşiri. Este uşor de observat că

$$\mathcal{S}m(P;Q) = \mathcal{S}m(P); \mathcal{S}m(Q).$$

3.2.2 Suma

Suma a două programe arbitrare se face pur şi simplu alăturând cele două scheme. Intrările, respectiv ieşirile programului al doilea sunt numerotate în continuarea intrărilor, respectiv ieşirilor primului program.

Pentru $f \in PFn(S)(a, b)$ şi $g \in PFn(S)(c, d)$ definim $f + g \in PFn(S)(a + c, b + d)$ prin

$$(f+g)(s,i) = \begin{cases} f(s,i) & \text{dacă } i \in [a] \\ (t, b+j) & \text{dacă } i > a \text{ şi } g(s, i-a) = (t, j) \end{cases}$$

pentru orice $s \in S$ şi $i \in [a + c]$.

Fie P un program determinist cu a intrări şi b ieşiri şi Q un program determinist cu c intrări şi d ieşiri. Este uşor de văzut că

$$\mathcal{S}m(P+Q) = \mathcal{S}m(P) + \mathcal{S}m(Q).$$

3.2.3 Funcţiile parţiale ca programe

Programe obişnuite au ca teorie suport o subteorie a teoriei relaţiilor finite Rel definită de familia de mulţimi

$\mathsf{Rel}(m,n) = \{r | r \subseteq [m] \times [n]$ relaţie $\}$, pentru $m, n \in \omega$

Relaţiile finite formează o categorie în care obiectele alcătuiesc monoidul numerelor naturale $(\omega, +, 0)$. Operaţiile în Rel sunt:
1) Compunerea

$$r \cdot r' = \{(i,k) | (\exists j)[(i,j) \in r \text{ şi } (j,k) \in r']\} \in \mathsf{Rel}(m,p)$$

pentru $r \in \mathsf{Rel}(m,n)$ şi $r' \in \mathsf{Rel}(n,p)$.
2) Identităţi

$$1_m = \{(i,i) | i \in [m]\} \in \mathsf{Rel}(m,m).$$

Funcţiile parţiale finite formează o subcategorie PFn.

Deasemenea funcţiile finite formează o subcategorie Fn. Funţiile finite sunt cele mai simple programe complet deterministe.

Funţiile parţiale finite sunt cele mai simple programe deterministe, mai precis este vorba de programe fără instrucţiuni. Execuţia lor nu modifică starea memoriei, ci se caracterizează printr-o simplă trecere de la o intrare la o ieşire atunci când funcţia duce intrarea în ieşire. Semantica lor va fi reflectată de un functor care păstreză obiectele

$$H : \mathsf{PFn} \longrightarrow PFn(S).$$

Fie $f \in \mathsf{PFn}(m,n)$. Prin definiţie, $H(f) \in PFn(S)(m,n)$ este definit pentru orice $i \in [m]$ şi $s \in S$ prin

$$H(f)(s,i) = (s, f(i)).$$

3.2.4 if_then_else_

Fie t un test. Fie p şi q două programe cu o intrare şi o ieşire. Putem scrie

$$\text{if } t \text{ then } p \text{ else } q = t; (p+q); \mathrm{V}$$

unde V este programul fără instruţiuni cu două intrări şi o ieşire în care fiecare intrare este conectată la unica ieşire. Prin urmare

$$\mathcal{S}m(\text{if } t \text{ then } p \text{ else } q) = \mathcal{S}m(t); (\mathcal{S}m(p) + \mathcal{S}m(q)); H(\mathrm{V}).$$

Un calcul simplu arată că pentru orice stare s

$$\mathcal{S}m(\text{if } t \text{ then } p \text{ else } q)(s,1) = \begin{cases} \mathcal{S}m(p)(s',1) & \text{dacă } \mathcal{S}m(t)(s,1) = (s',1) \\ \mathcal{S}m(q)(s',1) & \text{dacă } \mathcal{S}m(t)(s,1) = (s',2). \end{cases}$$

3.2.5 Feedback

Pentru $f \in PFn(S)(1+a, 1+b)$ definim $\uparrow f \in PFn(S)(a,b)$ pentru orice $s, t \in S$, $i \in [a]$ şi $j \in [b]$ prin

$$\uparrow f(s,i) = (t,j) \text{ dacă şi numai dacă}$$

$f(s, 1+i) = (t, 1+j)$ sau
există $n \geq 1$, există $s_1, s_2, \ldots, s_n \in S$ astfel încât
1) $f(s, 1+i) = (s_1, 1)$ şi
2) $f(s_k, 1) = (s_{k+1}, 1)$ pentru orice $k \in [n-1]$ şi
3) $f(s_n, 1) = (t, 1+j)$.

Această definiţie a fost astfel dată ca pentru orice program P cu $1+a$ intrări şi $1+b$ ieşiri să putem scrie egalitatea

$$\mathcal{S}m(\uparrow P) = \uparrow \mathcal{S}m(P).$$

3.2.6 While

Fie t un test şi p un program cu o intrare şi o ieşire. Putem scrie

$$\text{while } t \text{ do } p = \uparrow (\mathrm{V}; t; (p + 1_1))$$

unde 1_1 este programul fără instrucţiuni cu o intrare conectată la unica lui ieşire.

Prin urmare

$$\mathcal{S}m(\text{while } t \text{ do } p) = \uparrow (H(\mathrm{V}); \mathcal{S}m(t); (\mathcal{S}m(p) + H(1_1))).$$

Proposition 1 Pentru orice stare s_0

$$\mathcal{S}m(\text{while } t \text{ do } p)(s_0, 1) = (s, 1)$$

dacă şi numai dacă există $n \geq 0$ şi stările $t_1, s_1, t_2, s_2, \ldots, s_n, t_n$ cu proprietăţile

1) $\mathcal{S}m(t)(s_{i-1}, 1) = (t_i, 1)$ şi $\mathcal{S}m(p)(t_i, 1) = (s_i, 1)$ pentru orice $i \in [n]$ ŞI

2) $\mathcal{S}m(t)(s_n, 1) = (s, 2)$. □

Proof: Pentru o mai bună înţelegere menţionăm că n este numărul de repetări ale buclei, s_i este starea în care se ajunge la execuţia testului t după i repetări ale buclei şi t_i este starea în care începe execuţia a i-a a instrucţiunii p.

Observăm că

$$[H(\mathrm{V}); \mathcal{S}m(t); (\mathcal{S}m(p) + 1_1)](s, i) = \begin{cases} \mathcal{S}m(p)(s', 1) & \text{dacă } \mathcal{S}m(t)(s, 1) = (s', 1) \\ \mathcal{S}m(t)(s, 1) & \text{dacă } \mathcal{S}m(t)(s, 1) = (s', 2) \end{cases}$$

Prin urmare notând $F = H(\mathrm{V}); \mathcal{S}m(t); (\mathcal{S}m(p) + 1_1)$ deducem

$\mathcal{S}m(\text{while } t \text{ do } p)(s_0, 1) = (s, 1)$ dacă şi numai dacă

$(\uparrow F)(s_0, 1) = (s, 1)$

dacă şi numai dacă

$$F(s_0, 2) = (s, 2) \quad \text{SAU}$$
$$(\exists n \geq 1)(\exists s_1, s_2, \ldots, s_n \in S) \begin{cases} F(s_0, 2) = (s_1, 1) \\ F(s_k, 1) = (s_{k+1}, 1) \quad \text{dacă} \quad 1 \leq k \leq n-1 \\ F(s_n, 1) = (s, 2) \end{cases}$$

dacă şi numai dacă

$$\mathcal{S}m(t)(s_0, 1) = (s, 2) \quad \text{SAU}$$
$$(\exists n \geq 1)(\exists s_1, s_2, \ldots, s_n \in S) \begin{cases} (\exists t_1 \in S)\mathcal{S}m(t)(s_0, 1) = (t_1, 1) \text{ şi } \mathcal{S}m(p)(t_1, 1) = (s_1, 1) \\ (\forall 1 \leq k \leq n-1)(\exists t_{k+1} \in S)\mathcal{S}m(t)(s_k, 1) = (t_{k+1}, 1) \text{ şi } \mathcal{S}m(p)(t_{k+1}, 1) = (s_{k+1}, 1) \\ \mathcal{S}m(t)(s_n, 1) = (s, 2) \end{cases}$$

dacă şi numai dacă(comasare)

$$\mathcal{S}m(t)(s_0, 1) = (s, 2) \quad \text{SAU}$$
$$(\exists n \geq 1)(\exists s_1, t_1, s_2, t_2, \ldots, s_n, t_n \in S) \begin{cases} (\forall 0 \leq k \leq n-1)\mathcal{S}m(t)(s_k, 1) = (t_{k+1}, 1) \text{ şi } \mathcal{S}m(p)(t_{k+1}, 1) = (s_{k+1}, 1) \\ \mathcal{S}m(t)(s_n, 1) = (s, 2) \end{cases}$$

dacă şi numai dacă(comasare)

$$(\exists n \geq 0)(\exists s_1, t_1, s_2, t_2, \ldots, s_n, t_n \in S) \begin{cases} (\forall 0 \leq k \leq n-1)\mathcal{S}m(t)(s_k, 1) = (t_{k+1}, 1) \text{ şi } \mathcal{S}m(p)(t_{k+1}, 1) = (s_{k+1}, 1) \\ \mathcal{S}m(t)(s_n, 1) = (s, 2) \end{cases}$$

3.3 Semantica unui minilimbaj

Programele de care vorbim acum sunt construite plecând de la o mulţime **Atrib** de atribuiri şi o mulţime **Teste** de teste a căror semantică o cunoaştem. Cu alte cuvinte este dată o mulţime S a stărilor calculatorului care execută programele şi două funcţii notate la fel

$$\mathcal{I} : \mathbf{Atrib} \to Pfn(S)(1,1) \quad \text{şi} \quad \mathcal{I} : \mathbf{Teste} \to Pfn(S)(1,2).$$

Programele sunt construite folosind ideile clasile ale programării structurate. Sintaxa este dată în stilul "abstract" aşa cum se obişnuieşte atunci când subiectul principal este semantica. Gramatica are un singur neterminal P de la program. Producţiile sunt

Aa	P ::= a	pentru orice a ∈ **Atrib**
B	P ::= { P }	
It	P ::= if t then P else P	pentru orice t ∈ **Teste**
C	P ::= P ; P	
Wt	P ::= while t do P	pentru orice t ∈ **Teste**.

Semantica este definită prim metoda semanticii algebrei iniţiale. Suportul algebrei semantice este Pfn(S)(1,1) iar operaţiile sunt

Aa	=	$\mathcal{I}$(a)
B(x)	=	x
It(x, y)	=	$\mathcal{I}(\mathrm{t}); (x + y); H(V)$
C(x, y)	=	$x; y$
Wt(x)	=	$\uparrow (H(V); \mathcal{I}(\mathrm{t}); (x + H(1_1)))$.

2 PROGRAME ÎN SENS ELGOT

Virgil Emil Căzănescu

February 14, 2010

Reamintim cadrul de lucru şi caracteristicile unui program în stil Elgot.
Σ este mulţimea instrucţiunilor. Notând cu ω mulţimea numerelor naturale, funcţiile

$$i : \Sigma \longrightarrow \omega \text{ şi } o : \Sigma \longrightarrow \omega$$

dau pentru fiecare instrucţiune $\sigma \in \Sigma$, numărul intrărilor ei $i(\sigma)$ şi numărul ieşirilor ei $o(\sigma)$.

În mare un program este un graf orientat şi etichetat. Vom lucra cu programe deterministe în care pot apare trei tipuri de noduri:

1. Noduri de intrare în care nu intră săgeţi şi din care pleacă cel mult o singură săgeată. Le numerotăm cu numere de la 1 la a.
2. Noduri *interne* etichetate cu instrucţiuni. Din fiecare ieşire de instrucţiune care etichetează un nod intern pleacă cel mult o singură săgeată.
3. Noduri de ieşire din care nu pleacă săgeţi. Le numerotăm cu numere de la 1 la b.

Reamintim că fiecare săgeată trebuie să ajungă la o ieşire a programului sau la una dintre intrările instrucţiunilor care etichetează nodurile interne.

1 Structurarea programelor Elgot

Ilustrarea ideilor de mai jos o vom face plecând de la programul din Figura 1. Vom trece la o altă reprezentare a programelor în care săgeţile(element grafic) vor fi înlocuite cu o funcţie parţială. Pentru obţinerea acestei reprezentări procedăm în modul descris în continuare. Menţionăm că metoda descrisă mai jos poate fi aplicată oricărui program.

Deşi nu există nici un motiv, vom prefera să introducem o relaţie de ordine totală pe mulţimea nodurilor interne. Această relaţie de ordine este singurul element artificial pe care-l va conţine definiţia noţiunii de program. Înlăturarea acestui element artificial va fi explicată mult mai târziu.

Ordinea aleasă pentru exemplul nostru, se poate vedea în Figura 2 citind instrucţiunile de la stânga la dreapta.

Menţionăm că dacă aceeaşi instrucţiune etichetează mai multe noduri din graf atunci ea trebuie să apară în figura similară Figurii 2 de acelaşi număr de ori. Scriem instrucţiunile pe o linie respectând ordinea aleasă şi sub ele desenăm un dreptunghi

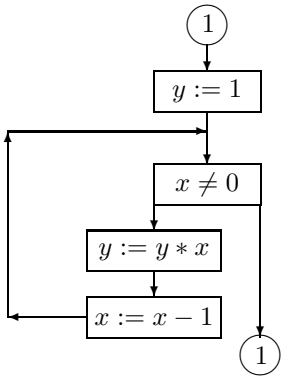


Figure 1: Program pentru $y := x!$

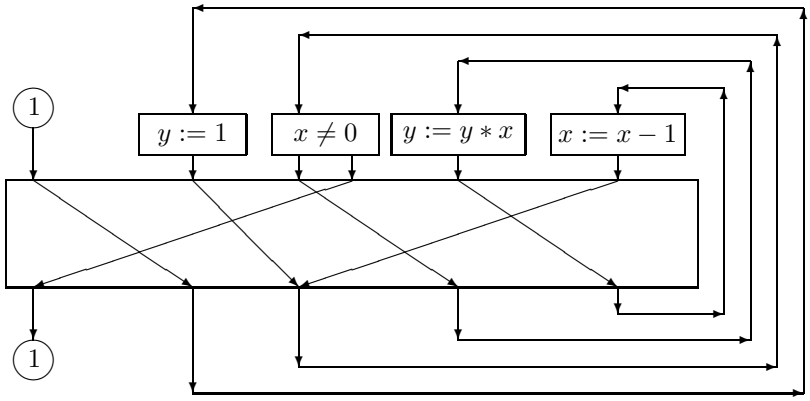


Figure 2: Reprezentarea programului pentru $y := x$!

la care, în partea dreaptă conectăm ieşirile instrucţiunilor. În partea din stânga sus a dreptunghiului conectăm intrările programului, iar în partea din stânga jos a dreptunghiului conectăm ieşirile programului. În partea de jos a dreptunghiului, după conectările ieşirilor programului, punem nişte copii ale intrărilor instrucţiunilor, în ordinea deja aleasă pentru instrucţiuni, pe care le conectăm la intrările corespunzătoare prin săgeţi. În final în dreptunghi punem săgeţi pentru a restabili legăturile date de săgeţile din Figura 1 şi numai acestea.

Transformând Figura 1 după aceste idei obţinem Figura 2.

Aducerea programelor la forma din Figura 2 ne permite să definim mai uşor un concept matematic care să reprezinte noţiunea intuitivă de program.

Înainte de aceasta, vom introduce nişte notaţii. Dacă A este o mulţime notăm cu A^* monoidul liber generat de A. Modelul cel mai uzual pentru A^* este mulţimea şirurilor finite de elemente din A. Fie $m \in A^*$. Notăm cu $|m|$ lungimea lui m. Observăm că putem privi m ca o funcţie

$$m : [\,|m|\,] \longrightarrow A.$$

Produsul(concatenarea) elementelor m şi m' din A^* este definit prin

$$mm'(i) = \begin{cases} m(i) & \text{dacă } i \in [\,|m|\,] \\ m'(i - |m|) & \text{dacă } |m| < i \leq |m| + |m'|. \end{cases}$$

Pentru $i \in [\,|m|\,]$ vom prefera să scriem m_i în loc de $m(i)$. Prin urmare pentru orice $m \in A^*$ putem scrie

$$m = m_1 m_2 \dots m_{|m|}.$$

Vom identifica elementele lui A cu şirurile de lungime 1 din A^*, prin urmare $A \subset A^*$. Reamintim că orice funcţie $f : A \longrightarrow M$ care ia valori într-un monoid M se poate prelungi în mod unic la un morfism de monoizi $f : A^* \longrightarrow M$ prin egalitatea

$$f(m) = f(m_1)f(m_2)\dots f(m_{|m|}).$$

Astfel de extinderi vor fi folosite pentru funcţiile i şi o. Fie

$$i : \Sigma^* \longrightarrow (\omega, +, 0) \text{ şi } o : \Sigma^* \longrightarrow (\omega, +, 0)$$

extinderile lor la monoidul aditiv al numerelor naturale.

Revenind la programe formăm un element $m \in \Sigma^*$ alcătuind un şir finit cu toate etichetele nodurilor interne în ordinea aleasă pentru acestea. Menţionăm că m reprezintă programul format din suma programelor date de literele lui m, prin urmare concatenarea trebuie interpretată ca sumă a programelor. Faptul că numărul intrărilor, respectiv al ieşirilor sumei programelor este suma numărului intrărilor, respectic al ieşirilor programelor care se adună este motivaţia pentru utilizarea mai sus a monoidului aditiv al numerelor naturale.

Pentru exemplul nostru $m = m_1 m_2 m_3 m_4$ unde

$$m_1 = y := 1,\ m_2 = x \neq 0,\ m_3 = y := y * x \text{ şi } m_4 = x := x - 1.$$

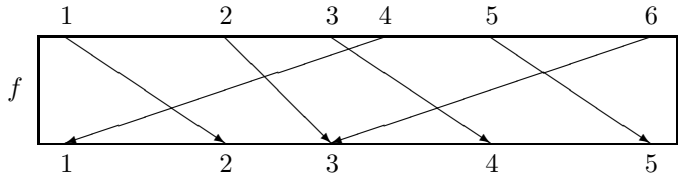


Figure 3: Funcţia $f\colon [\,1+(1+2+1+1)\,] \longrightarrow [\,1+(1+1+1+1)\,]$

Observăm că $i(m) = 1+1+1+1 = 4$ şi că $o(m) = 1+2+1+1 = 5$.

Pentru înlocuirea săgeţilor(element grafic) cu un obiect matematic copiem dreptunghiul din Figura 2 numerotând punctele de conexiune de pe laturile de sus şi de jos de la stânga la dreapta obţinând Figura 3. Observăm că am obţinut o funcţie $f : [\,6\,] \longrightarrow [\,5\,]$ a cărei definiţie este dată în tabelul

j	1	2	3	4	5	6
f(j)	2	3	4	1	5	3

În general se obţine o funcţie parţială

$$f : [\,a+o(m)\,] \multimap [\,b+i(m)\,].$$

Observăm că f este o funcţie dacă şi numai dacă programul este complet determinist.

Analizând reprezentarea din Figura 2 a schemei logice a factorialului observăm că întregul desen poate fi reconstituit plecând de la şirul m al instrucţiunilor şi de la funcţia f cu ajutorul numerelor a, b şi al funcţiilor i şi o.

În concluzie un *program determinist cu a intrări şi b ieşiri* este un cuplu (m, f) unde

- $m \in \Sigma^*$
- $f : [\,a+o(m)\,] \multimap [\,b+i(m)\,]$ este o funcţie parţială.

Definiţie 1 *Fie a şi b două numere naturale. Se numeşte program(abstract) determinist cu a intrări şi b ieşiri un cuplu $P = \langle m,\, f\rangle$ format dintr-un şir de instrucţiuni $m \in \Sigma^*$ şi o funcţie parţială*

$$f\colon [\,a+o(m)\,] \multimap [\,b+i(m)\,].$$

Funcţia parţială f va fi numită *conexiune* deoarece ea include toate legăturile date prin săgeţi între diversele elemente ale programelor. Generalitatea construcţiei de mai sus conduce la:

Theorem 1 Orice program abstract poate fi reprezentat ca o pereche <şir de instrucţiuni, conexiune>. □

Denumirea de reprezentare este justificată de faptul că un program poate avea mai multe reprezentări. Diferenţa dintre ele este o permutare a şirului instrucţiunilor care induce o "permutare între săgeţile conexiunilor". Mai târziu va fi dată o condiţie necesară şi suficientă ca două perechi de forma <şir de instrucţiuni, conexiune> să reprezinte acelaşi program.

Reanalzând Figura 2 observăm că apare un feedback la dreapta. Feedbacul la dreapta acţionează la fel ca cel la stânga cu deosebirea că utilizează ultima ieşire şi ultima intrare. Feedbackul la dreapta al programului P se notează cu $P\uparrow$. Un feedback multiplu este o repetare a unui feedback care afectează o intrare şi o ieşire. Pentru $n \in \omega$ notăm cu $\uparrow^n$ o repetare de n ori a feedbackului.

Theorem 2 (**Structurarea programelor**) Orice program determinist poate fi generat de instrucţiunile din Σ şi de programele fără instrucţiuni folosind ca operaţii compunerea, suma şi feedbackul multiplu la dreapta.

Proof: Fie $P = \langle m,\, f\rangle$ un program determinist cu a intrări şi b ieşiri. Observăm că orice şir de instrucţiuni este suma instrucţiunilor componente şi că

$$P = ((1_a + m); f)\uparrow^{i(m)}. \ \square$$

Ne referim în cele ce urmează la programele nedeterministe. Faptele expuse mai sus în cazul determinist rămân, cu mici modificări, adevărate şi în cazul nedeterminist.

Într-un program nedeterminist din orice nod de intrare al programului şi din orice ieşire de instrucţiune poate pleca un număr arbitrar de săgeţi. În reprezentarea ca pereche <şir de instrucţiuni, conexiune> a unui program nedeterminist, conexiunea este o relaţie.

Definiţie 2 *Fie a şi b două numere naturale. Se numeşte program(abstract) nedeterminist cu a intrări şi b ieşiri un cuplu* $P = \langle m, r\rangle$ *format dintr-un şir de instrucţiuni* $m \in \Sigma^*$ *şi o relaţie finită*

$$r\colon [\, a + o(m)\,] \to [\, b + i(m)\,].$$

În cazul nedeterminist programele fără instrucţiuni sunt relaţiile. Teorema privind structurarea programelor este adevărată şi în cazul nedeterminist.

Reprezentarea programelor ca perechi reprezintă un prim pas spre al treilea nivel de abstarctizare unde nu se mai face diferenţierea între programele deterministe şi nedeterministe. Observăm că singura diferenţă între ele este la nivelul conexiunii. Abstractizarea privind sintaxa se va realiza prin înlocuirea funcţiilor parţiale, respectiv a relaţiilor printr-un concept unic care va putea fi paticularizat atât prin funcţii parţiale finite cât şi prin relaţii finite.

2 Sintaxă

Vom începe a analiza operaţiile cu programe. Pentru aceasta este necesar să reamintim alte operaţii cu relaţiile finite pe care le vom utiliza în continuare.

2.1 Relaţii finite

Programe obişnuite au ca teorie suport o subteorie a teoriei relaţiilor finite Rel.

Alte operaţiile în Rel sunt:

1) Suma

$$r + r' = r \cup \{(m + i, n + j) | (i, j) \in r'\} \in \mathsf{Rel}(m + p, n + q)$$

pentru $r \in \mathsf{Rel}(m, n)$ şi $r' \in \mathsf{Rel}(p, q)$.

2) Transpoziţii bloc

$${}^m\mathrm{X}^n = \{(i, n + i) | i \in [m]\} \cup \{(m + i, i) | i \in [n]\} \in \mathsf{Rel}(m + n, n + m).$$

3) Feedback la dreapta

$$r\uparrow = \{(i, j) | (i, j) \in r \textit{ sau } [(i, n + 1) \in r \textit{ şi } (m + 1, j) \in r]\} \in \mathsf{Rel}(m, n)$$

pentru $r \in \mathsf{Rel}(m + 1, n + 1)$.

4) Feedback la stânga

$$\uparrow r = \{(i, j) | (1 + i, 1 + j) \in r \textit{ sau } [(1 + i, 1) \in r \textit{ şi } (1, 1 + j) \in r]\} \in \mathsf{Rel}(m, n)$$

pentru $r \in \mathsf{Rel}(1 + m, 1 + n)$.

Fiecare din cele două feedbackuri poate fi aplicat de mai multe ori, folosindu-se notaţia $r\uparrow^i$ sau $\uparrow^i r$ unde i este numărul repetărilor.

Teoremă 1 *Funcţiile parţiale finite formează o subcategorie* PFn *închisă la toate operaţiile de mai sus.*

Demonstraţie. Probăm numai pentru feedbackul unar la dreapta.

Fie $r \in \mathsf{PFn}(m + 1, n + 1)$. Presupunem că (i, j) şi (i, k) sunt în $r\uparrow$ şi arătăm că $j = k$.

Evident $i \in [m]$ şi $j, k \in [n]$. Analizăm diversele situaţii posibile.

1. Dacă (i, j) şi (i, k) sunt în r rezultă $j = k$ deoarece r este funcţie parţială.
2. Dacă $(i, n + 1) \in r$ şi $(m + 1, j) \in r$ precum şi $(i, n + 1) \in r$ şi $(m + 1, k) \in r$ din $(m + 1, j) \in r$, $(m + 1, k) \in r$ rezultă $j = k$ deoarece r este funcţie parţială.
3. Dacă $(i, j) \in r$ precum şi $(i, n + 1) \in r$ şi $(m + 1, k) \in r$ din $(i, j) \in r$, $(i, n + 1) \in r$ şi $j \in [n]$ deducem că r nu este funcţie parţială, contradicţie.
4. Dacă $(i, n + 1) \in r$ şi $(m + 1, j) \in r$ precum şi $(i, k) \in r$ obţinem o contradicţie ca şi în cazul precedent. □

Deasemenea funcţiile finite formează o subcategorie Fn închisă la toate operaţiile de mai sus cu excepţia feedbackului.

2.2 Compunerea

Compunerea între două programe este definită numai dacă numărul ieşirilor primului program este egal cu numărul intrărilor programului la doilea. Fie $F = (x, f)$ un program cu a intrări şi b ieşiri şi $G = (y, g)$ un program cu b intrări şi c ieşiri. Compunerea se face legând ieşirile lui F cu intrările corespunzătoareale lui G obţinându-se un program cu a intrări şi c ieşiri.

$$(x, f)(y, g) = (xy, (f + 1_{o(y)})(1_b + {}^{i(x)}\mathrm{X}^{o(y)})(g + 1_{i(x)})(1_c + {}^{i(y)}\mathrm{X}^{i(x)})).$$

2.3 Suma

Suma a două programe arbitrare se face pur şi simplu alăturând cele două programeme. Intrările, respectiv ieşirile programului al doilea sunt numerotate în continuarea intrărilor, respectiv ieşirilor primului program. Fie $F = (x, f)$ un program cu a intrări şi b ieşiri şi $G = (y, g)$ un program cu c intrări şi d ieşiri.

$$(x, f) + (y, g) = (xy, (1_a + {}^{c}\mathrm{X}^{o(x)} + 1_{o(y)})(f + g)(1_b + {}^{i(x)}\mathrm{X}^{d} + 1_{i(y)})).$$

2.4 Feedback

Feedbackul este o operaţie unară, el acţionând asupra unui program cu cel puţin o intrare şi cel puţin o ieşire. Feedbackul se poate face fie la stânga, fie la dreapta. Feedbackul la stânga(dreapta) se efectuează legând cea mai din stânga(dreapta) ieşire la cea mai din stânga(dreapta) intrare. După efectuarea feedbackului numărul intrărilor, respectiv al ieşirilor scade cu o unitate. Fiecare din cele două feedbackuri poate fi repetat. Iată definiţiile pentru feedback de a ori.

Fie $F = (x, f)$ un program cu $a + b$ intrări şi $a + c$ ieşiri şi $G = (y, g)$ o schemă cu $b + a$ intrări şi $c + a$ ieşiri.

$$\uparrow^a(x, f) = (x, \uparrow^a f);$$

$$(y, g)\uparrow^a = (y, [(1_b + {}^{o(y)}\mathrm{X}^{a})g(1_c + {}^{a}\mathrm{X}^{i(y)})]\uparrow^a).$$

2.5 Spre al treilea nivel de abstractizare

Introducem în discuţia noastră şi programele nedeterministe, adică acelea în care conexiunea este dată de o relaţie. Aceasta înseamnă că există intrări ale programului sau ieşiri ale instrucţiunilor din care pleacă mai multe săgeţi. Din punct de vedere semantic, într-un astfel de punct execuţia se continuă alegând nedeterminist una dintre aceste săgeţi.

Abstarctizare se poate face asupra conexiunii. Funcţiile parţiale, respectiv relaţiile pot fi înlocuite printr-un element dintr-o structură algebrică abstractă. Aceasta trebuie aleasă astfel încât:

1) funcţiile parţiale şi relaţiile să fie cazuri particulare,

2) definiţiile de mai sus pentru compunere, sumă şi feedback să aibă sens.

Noua structură algebrică va fi prezentată în capitolulul următor. Cu ajutorul acesteia se realizează unificarea studiului programelor deterministe şi nedeterministe.

3 Semantică

Reamintim că în cazul determinist o interpretare a lui Σ este formată dintr-o mulţime S şi câte o funcţie parţială

$$\mathcal{I}(\sigma) : S \times [i(\sigma)] \multimap S \times [o(\sigma)]$$

pentru fiecare $\sigma \in \Sigma$.

3.1 Programe deterministe

Reamintim că $PFn(S)$ este modelul semantic fundamental pentru programe deterministe. S este mulţimea stărilor maşinii care execută programele. $PFn(S)$ este o categorie având numerele naturale ca obiecte. Pentru două numere naturale a şi b, $PFn(S)(a, b)$ este prin definiţie mulţimea funcţiilor parţiale de la $S \times [a]$ la $S \times [b]$.

Dacă P este un program determinist cu a intrări şi b ieşiri vom nota semantica lui prin

$$\mathcal{S}m(P) \in PFn(S)(a, b).$$

3.1.1 Compunerea

Fie P un program determinist cu a intrări şi b ieşiri şi Q un program determinist cu b intrări şi c ieşiri. Reamintim că

$$\mathcal{S}m(P; Q) = \mathcal{S}m(P)\mathcal{S}m(Q).$$

3.1.2 Suma

Fie P un program determinist cu a intrări şi b ieşiri şi Q un program determinist cu c intrări şi d ieşiri. Reamintim că

$$\mathcal{S}m(P + Q) = \mathcal{S}m(P) + \mathcal{S}m(Q).$$

3.1.3 Feedback

Pentru $f \in PFn(S)(a + c, b + c)$ definim $f\uparrow^c \in PFn(S)(a, b)$ pentru orice $s, t \in S$, $i \in [a]$ şi $j \in [b]$ prin

$$f\uparrow^c (s, i) = (t, j) \text{ dacă şi numai dacă}$$

$f(s, i) = (t, j)$ sau

există $n \geq 1$, există $s_1, s_2, \ldots, s_n \in S$ şi $i_1, i_2, \ldots, i_n \in [c]$ astfel încât

1) $f(s, i) = (s_1, b + i_1)$ şi
2) $f(s_k, a + i_k) = (s_{k+1}, b + i_{k+1})$ pentru orice $k \in [n - 1]$ şi
3) $f(s_n, a + i_n) = (t, j)$.

Fie P un program determinist cu $a + c$ intrări şi $b + c$ ieşiri . Este uşor de văzut că

$$\mathcal{S}m(P\uparrow^c) = (\mathcal{S}m(P))\uparrow^c .$$

3.1.4 Funţiile parţiale ca programe

Reamintim că semantica funcţiilor parţiale gândite ca programe este dată de un functor care păstreză obiectele

$$H : PFn \longrightarrow PFn(S).$$

Fie $f \in \mathsf{PFn}(m, n)$. Prin definiţie, pentru orice $i \in [m]$ şi $s \in S$

$$H(f)(s, i) = (s, f(i)).$$

3.1.5 Concluzii

1. Semantica are proprităţi caracteristice unui morfism
2. Dacă (x, r) este un program determinist cu a intrări, b ieşiri, instrucţiuni din Σ şi conexiuni din PFn atunci semantica lui este dată de formula

   $$\mathcal{S}m(x, r) = [(1_a + \mathcal{S}m(x))H(r)] \uparrow^{i(x)}$$

   unde $\mathcal{S}m(x)$ este suma semanticilor instrucţiunilor lui x şi $i(x)$ este suma numărului intrărilor instrucţiunilor lui x.

3.2 Programe nedeterminite

Vom prefera să schimbăm stilul în care definim conceptul de interpretare pentru a scoate în relief şi alt punct de vedere.

Vom utiliza semiinelul $(\mathcal{P}(S \times S), \cup, \emptyset, ;, 1_S)$ relaţiilor între stări.

Definiţie 3 *O* interpretare *a lui* Σ *este formată dintr-o mulţime* S *şi câte o matrice* $\mathcal{I}(\sigma)$ *cu* $i(\sigma)$ *linii şi* $o(\sigma)$ *coloane cu elemente din semiinelul relaţiilor pentru fiecare* $\sigma \in \Sigma$.

Pentru $\sigma \in \Sigma$, $s, t \in S$, $n \in [i(\sigma)]$ şi $m \in [o(\sigma)]$ apartenenţa

$$(s, t) \in \mathcal{I}(\sigma)_{nm}$$

are următoarea semnificaţie:

începând execuţia instrucţiunii σ *de la intrarea* n *a ei şi din starea* s *a memoriei, execuţia se poate termina în starea* t *a memoriei la ieşirea* m *a lui* σ.

Ideea principală este de a defini semantica ca un morfism. Pentru orice program P vom nota cu $\mathcal{S}m(P)$ matricea sa semantică, adică matricea care ne spune cum se execută P.

3.2.1 Compunerea

Fie P un program nedeterminist cu a intrări şi b ieşiri şi Q un program nedeterminist cu b intrări şi c ieşiri. Vom calcula matricea semantică a compunerii $P; Q$.

Pentru $i \in [a]$, $k \in [c]$ şi $s, t \in S$ observăm că

$$(s, t) \in \mathcal{S}m(P; Q)_{ik}$$

dacă şi numai dacă
începând execuţia lui $P; Q$ de la intrarea i, din starea s a memoriei, execuţia se poate termină în starea t a memoriei la ieşirea k a lui Q

dacă şi numai dacă
există $j \in [b]$ şi $u \in S$ astfel încât începând execuţia lui P de la intrarea i, din starea s a memoriei, execuţia se poate termină în starea u a memoriei la ieşirea j a lui P şi începând execuţia lui Q de la intrarea j, din starea u a memoriei, execuţia se poate termină în starea t a memoriei la ieşirea k a lui Q

dacă şi numai dacă
există $j \in [b]$ şi $u \in S$ astfel încât $(s, u) \in \mathcal{S}m(P)_{ij}$ şi $(u, t) \in \mathcal{S}m(Q)_{jk}$

dacă şi numai dacă există $j \in [b]$ astfel încât $(s, t) \in \mathcal{S}m(P)_{ij}\mathcal{S}m(Q)_{jk}$

dacă şi numai dacă $(s, t) \in \bigcup_{j=1}^{j=b} \mathcal{S}m(P)_{ij}\mathcal{S}m(Q)_{jk}$

dacă şi numai dacă $(s, t) \in (\mathcal{S}m(P)\mathcal{S}m(Q))_{ik}$.

Deci semantica lui $P; Q$ este egală cu produsul matricilor semantice ale lui P şi Q, adică

$$\mathcal{S}m(P; Q) = \mathcal{S}m(P)\mathcal{S}m(Q).$$

3.2.2 Suma

Pentru două matrici A şi B cu elemente din $\mathcal{P}(S \times S)$ definim suma lor $A + B$ prin

$$A + B = \begin{pmatrix} A & \emptyset \\ \emptyset & B \end{pmatrix}$$

unde $\emptyset$ este o matrice în care toate elementele sunt relaţia vidă. Remarcăm că numarul liniilor(coloanelor) lui $A + B$ este egal cu suma numărului liniilor(coloanelor) lui A şi a numărului liniilor(coloanelor) lui B.

Fie P un program nedeterminist cu a intrări şi b ieşiri şi Q un program nedeterminist cu c intrări şi d ieşiri. Este uşor de văzut că matricea semantică a sumei $P + Q$ este suma matricilor semantice ale lui P şi Q, adică

$$\mathcal{S}m(P + Q) = \mathcal{S}m(P) + \mathcal{S}m(Q).$$

3.2.3 Feedback

Pentru orice matrice M pătratică cu elemente din $\mathcal{P}(S \times S)$ prin definiţie

$$M^* = \bigcup_{n \geq 0} M^n.$$

Fie P un program nedeterminist cu $a+b$ intrări şi $a+c$ ieşiri. Vom calcula matricea semantică a feedbackului $\uparrow^a P$. Pentru cele ce urmează este util să scriem matricea $\mathcal{S}m(P)$ sub forma

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

unde A este o matrice pătratică de dimensiune a.

Pentru $i \in [b]$, $k \in [c]$ şi $s, t \in S$ observăm că

$$(s,t) \in \mathcal{S}m(\uparrow^a P)_{ik}$$

dacă şi numai dacă
începând execuţia lui $\uparrow^a P$ de la intrarea i, din starea s a memoriei, execuţia se poate termină în starea t a memoriei la ieşirea k a lui $\uparrow^a P$

dacă şi numai dacă
începând execuţia lui P de la intrarea $a+i$, din starea s a memoriei, execuţia se poate termină în starea t a memoriei la ieşirea $a+k$ a lui P sau
există $n \geq 1$, există $s_1, s_2, \ldots, s_n \in S$ şi $j_1, j_2, \ldots, j_n \in [a]$ astfel încât
1) începând execuţia lui P de la intrarea $a+i$, din starea s a memoriei, execuţia se poate termină în starea s_1 a memoriei la ieşirea j_1 a lui P,
2) pentru orice $u \in [n-1]$ începând execuţia lui P de la intrarea j_u, din starea s_u a memoriei, execuţia se poate termină în starea s_{u+1} a memoriei la ieşirea j_{u+1} a lui P şi
3) începând execuţia lui P de la intrarea j_n, din starea s_n a memoriei, execuţia se poate termină în starea t a memoriei la ieşirea $a+k$ a lui P

dacă şi numai dacă $(s,t) \in \mathcal{S}m(P)_{a+i\, a+k}$ sau
există $n \geq 1$, există $s_1, s_2, \ldots, s_n \in S$ şi $j_1, j_2, \ldots, j_n \in [a]$ astfel încât
1) $(s, s_1) \in \mathcal{S}m(P)_{a+i\, j_1}$,
2) $(s_u, s_{u+1}) \in \mathcal{S}m(P)_{j_u\, j_{u+1}}$ pentru orice $u \in [n-1]$ şi
3) $(s_n, t) \in \mathcal{S}m(P)_{j_n\, a+k}$

dacă şi numai dacă $(s,t) \in D_{ik}$ sau
există $n \geq 1$, există $s_1, s_2, \ldots, s_n \in S$ şi $j_1, j_2, \ldots, j_n \in [a]$ astfel încât
1) $(s, s_1) \in C_{ij_1}$,
2) $(s_u, s_{u+1}) \in A_{j_u j_{u+1}}$ pentru orice $u \in [n-1]$ şi
3) $(s_n, t) \in B_{j_n\, k}$

dacă şi numai dacă $(s,t) \in D_{ik}$ sau
există $s', s'' \in S$ şi $j, j' \in [a]$ astfel încât
$(s, s') \in C_{ij}$, $(s', s'') \in A^*_{jj'}$ şi $(s'', t) \in B_{j'\, k}$

dacă şi numai dacă $(s,t) \in D_{ik}$ sau $(s,t) \in (CA^*B)_{ik}$

dacă şi numai dacă $(s,t) \in (D \cup CA^*B)_{ik}$.

În concluzie, definind

$$\uparrow^a \begin{pmatrix} A & B \\ C & D \end{pmatrix} = D \cup CA^*B$$

deducem că $\mathcal{S}m(\uparrow^a P) = D \cup CA^*B = \uparrow^a \mathcal{S}m(P)$.

3.2.4 Relaţiile ca programe

Relaţiile sunt cele mai simple programe nedeterministe, mai precis este vorba de programe fără instrucţiuni. Execuţia lor nu modifică starea memoriei, ci se caracterizează printr-o simplă trecere de la o intrare la o ieşire atunci când există o săgeată între acestea. Semantica lor va fi reflectată de un functor care păstrează obiectele

$$H : Rel \longrightarrow [\mathcal{P}(S \times S)]$$

unde notaţia $[R]$ reprezintă categoria matricilor peste semiinelul R.

Fie $r \in \mathsf{Rel}(m, n)$. Prin definiţie, pentru orice $i \in [m]$ şi $j \in [n]$

$$H(r)_{ij} = \begin{cases} \emptyset & \text{dacă } (i,j) \notin r \\ 1_S & \text{dacă } (i,j) \in r \end{cases}$$

3.2.5 Concluzii

1. Semantica are proprităţi caracteristice unui morfism
2. Dacă (x, r) este un program nedeterminist cu a intrări, b ieşiri, instrucţiuni din Σ şi conexiuni din Rel atunci semantica lui este dată de formula
$$\mathcal{S}m(x, r) = [(1_a + \mathcal{S}m(x))H(r)] \uparrow^{i(x)}$$
unde $\mathcal{S}m(x)$ este suma semanticilor instrucţiunilor lui x şi $i(x)$ este suma numărului intrărilor instrucţiunilor lui x.

3.3 Spre al treilea nivel de abstractizare

Observăm asemănarea dintre formulele cu care se defineşte semantica unui program determinist şi nedeterminist. Acest fapt este un argument suplimentar in vederea unificării studiului programelor deterministe cu cele nedeterministe despre care vorbeam in expunerea despre sintaxă.

Observăm în plus că operaţiile utilizate în formulele cu care se defineşte semantica unui program şi anume compunerea, suma şi feedbackul sunt în linii mari aceleaşi ca cele utilizate în expunerea despre sintaxă. Aceste fapt permite ca structura algebrică care va fi utilizată pentru abstractizarea sintaxei să fie folosită şi pentru abstractizarea semanticii. Prin urmare la al treilea nivel de abstarctizare se va folosi o singură structură algebrică atât pentru studiul sintaxei cât şi pentru studiul semanticii.

3 Introducere în Algebra informaticii

VEC

April 20, 2004

1 Categorii strict monoidale

1.1 Cazul nepermutabil

Definiţia 1 *Se numeste* categorie strict monoidala nepermutabila *(csmn) o categorie N care satisface următoarele proprietăţi:*

- *obiectele formează un monoid $(Ob(N), \cdot, \lambda)$;*
- *oricare ar fi obiectele a, b, c, d se defineşte o operaţie adiţională suma, $+ \colon N(a,b) \times N(c,d) \longrightarrow N(ac, bd)$ care verifică următoarele axiome:*
  1. $f + (g + h) = (f + g) + h$;
  2. $f + 1_\lambda = 1_\lambda + f = f$;
  3. $1_a + 1_b = 1_{ab}$;
  4. $(f + g)(1_b + v) = f + gv$ *pentru* $f \colon a \longrightarrow b$, $g \colon c \longrightarrow d$ *şi* $v \colon d \longrightarrow m$;
  5. $(f + 1_d)(u + v) = fu + v$ *pentru* $f \colon a \longrightarrow b$, $u \colon b \longrightarrow c$ *şi* $v \colon d \longrightarrow m$.

Observaţia 1 *Proprietăţile 4 şi 5 din definiţia csmn sunt echivalente cu:*

i) $(f + 1_m)(g + 1_m) = fg + 1_m$ pentru $f \colon a \longrightarrow b$ şi $g \colon b \longrightarrow c$;

ii) $(f + 1_c)(1_b + g) = f + g$ pentru $f \colon a \longrightarrow b$ şi $g \colon c \longrightarrow d$;

iii) $(1_m + f)(1_m + g) = 1_m + fg$ pentru $f \colon a \longrightarrow b$ şi $g \colon b \longrightarrow c$.

Demonstraţie:
$\Longrightarrow$ este evidentă, deoarece i), ii) şi iii) sunt cazuri particulare ale proprietăţilor 4 şi 5.
$\Longleftarrow$ Demonstrăm 4:
$(f + g)(1_b + v) \overset{ii)}{=} (f + 1_c)(1_b + g)(1_b + v) \overset{iii)}{=} (f + 1_c)(1_b + gv) \overset{ii)}{=} f + gv$.
Demonstrăm 5:
$(f + 1_d)(u + v) \overset{ii)}{=} (f + 1_d)(u + 1_d)(1_c + v) \overset{i)}{=} (fu + 1_d)(1_c + v) \overset{ii)}{=} fu + v$. □

Definiţia 2 *O subcategorie a unei csmn N se numeşte* subcsmn *dacă are aceleaşi obiecte ca şi N şi este stabilă faţă de sumă.*

Definiţia 3 *Fie N o csmn şi G o mulţime de morfisme din N. Se numeşte subcsmn generată de G cea mai mică subcsmn a lui N care include G.*

Propoziţia 1 *Fie $\overline{G}$ categoria care are ca obiecte obiectele lui N şi ca morfisme compuneri finite de morfisme de forma $1_a + g + 1_b$ cu $g \in G$. Atunci $\overline{G}$ este subcsmn generată de G.*

Demonstraţie: Demonstrăm că $\overline{G}$ este subcsmn. Deoarece identităţile sunt compuneri de zero factori, $\overline{G}$ conţine toate identităţile lui N. Închiderea la compunere este evidentă, deci $\overline{G}$ este o sub-categorie a lui N. Arătăm că $\overline{G}$ este închisă la sumă.

Fie $t = t_1 \cdots t_n \in \overline{G}(a,b)$ şi $s = s_1 \cdots s_m \in \overline{G}(c,d)$. $t + s = (t + 1_c)(1_b + s) =$
$(t_1 \cdots t_n + 1_c)(1_b + s_1 \cdots s_m) = (t_1 + 1_c) \cdots (t_n + 1_c)(1_b + s_1) \cdots (1_b + s_m)$. Observăm că dacă $h = 1_u + g + 1_v$ atunci $1_b + h = 1_{bu} + g + 1_v$ şi $h + 1_c = 1_u + g + 1_{vc}$. Rezultă că $t + s \in \overline{G}(ab, cd)$.

Demonstrăm că $G \subseteq \overline{G}$. Din faptul că orice morfism g din G se poate scrie sub forma $g = 1_\lambda + g + 1_\lambda$ rezultă g este morfism din $\overline{G}$.

Fie M o subcsmn a lui N care include G. Deoarece M conţine identităţile lui N şi este închisă la sumă rezultă că M conţine morfismele de forma $1_a + g + 1_b$ cu g din G. Din faptul că M este închisă la compunere obţinem că M conţine morfismele lui $\overline{G}$.

Deci $\overline{G}$ este cea mai mică subcsmn a lui N care include G. □

Observaţia 2 *Cea mai mică subcsmn a lui N este subcsmn generată de $\emptyset$ şi are ca morfisme numai identităţile.*

Definiţia 4 *Dacă $f\colon a \longrightarrow b$ şi $g\colon c \longrightarrow d$ sunt două morfisme spunem că f permută cu g şi scriem*

$$f\mathbf{P}g \Leftrightarrow f + g = (1_a + g)(f + 1_d).$$

Observaţia 3 *Î̂n general, $f\mathbf{P}g$ nu este echivalent cu $g\mathbf{P}f$.*

Propoziţia 2 *Principalele proprietăţi ale relaţiei $\mathbf{P}$ sunt:*

1. *$f\mathbf{P}1_c$ şi $1_c\mathbf{P}f$ pentru $f\colon a \longrightarrow b$;*
2. *$f\mathbf{P}g$ implică $1_u + f\mathbf{P}g$ şi $f\mathbf{P}g + 1_u$ pentru $f\colon a \longrightarrow b$ şi $g\colon c \longrightarrow d$;*
3. *$f + 1_u\mathbf{P}g$ este echivalent cu $f\mathbf{P}1_u + g$ pentru $f\colon a \longrightarrow b$ şi $g\colon c \longrightarrow d$;*
4. *$f\mathbf{P}h$ şi $g\mathbf{P}h$ implică $fg\mathbf{P}h$ pentru $f\colon a \longrightarrow b$, $g\colon b \longrightarrow c$ şi $h\colon u \longrightarrow v$;*
5. *$h\mathbf{P}f$ şi $h\mathbf{P}g$ implică $h\mathbf{P}fg$ pentru $f\colon a \longrightarrow b$, $g\colon b \longrightarrow c$ şi $h\colon u \longrightarrow v$.*

Demonstraţie:

1. $(1_a + 1_c)(f + 1_c) = 1_{ac}(f + 1_c) = f + 1_c$;
2. Deoarece $f\mathbf{P}g$ rezultă $f + g = (1_a + g)(f + 1_d)$. Obţinem
   $(1_{ua} + g)(1_u + f + 1_d) = (1_u + (1_a + g))(1_u + (f + 1_d)) = 1_u + (1_b + g)(f + 1_d) = 1_u + (f + g) = (1_u + f) + g$;
3. $f + 1_u\mathbf{P}g \Leftrightarrow (f + 1_u) + g = (1_{au} + g)((f + 1_u) + 1_d) \Leftrightarrow f + (1_u + g) = (1_a + (1_u + g))(f + 1_{ud}) \Leftrightarrow f\mathbf{P}1_u + g$;
4. Deoarece $f\mathbf{P}h$ şi $g\mathbf{P}h$ rezultă $f + h = (1_a + h)(f + 1_v)$ şi $g + h$=$(1_b + h)(g + 1_v)$. Obţinem $(1_a + h)(fg + 1_v) = (1_a + h)(f + 1_v)(g + 1_v) = (f + h)(g + 1_v) = (f + 1_u)(1_b + h)(g + 1_v) = (f + 1_u)(g + h) = fg + h$;
5. Analog. □

1.2 Cazul permutabil

Definiţia 5 *Se numeste* categorie strict monoidala*(csm) o categorie N care satisface următoarele proprietăţi:*

- *obiectele formeaza un monoid $(Ob(N), \cdot, \lambda)$;*
- *pentru orice obiecte a, b, c, d se defineşte o operaţie adiţională suma,*
  $+\colon N(a,b) \times N(c,d) \longrightarrow N(ac, bd)$ care verifică următoarele axiome:
  1. *$f + (g + h) = (f + g) + h$;*
  2. *$f + 1_\lambda = 1_\lambda + f = f$;*
  3. *$1_a + 1_b = 1_{ab}$;*
  4. *$(f + g)(f' + g') = ff' + gg'$ pentru $f\colon a \longrightarrow b$, $g\colon c \longrightarrow d$, $f'\colon b \longrightarrow b'$ şi $g'\colon d \longrightarrow d'$.*

Propoziţia 3 *O csmn este o csm dacă şi numai dacă $f\mathbf{P}g$ oricare ar fi morfismele f şi g.*

Demonstraţie: Fie $f\colon a \longrightarrow b$ şi $g\colon c \longrightarrow d$ doua morfisme.
$\Longrightarrow$ este evidenta deoarece $(1_a + g)(f + 1_d) = 1_a f + g1_d = f + g$.
$\Longleftarrow$ Fie $f'\colon b \longrightarrow b'$ şi $g'\colon d \longrightarrow d'$. Conform axiomei (4) din definitia csmn, $f+g = (f+1_c)(1_b+g)$ şi $f'+g' = (f'+1_d)(1_{c'}+g')$.
Deoarece $f'\mathbf{P}g$ rezultă $f' + g = (1_b + g)(f' + 1_d)$. Obţinem
$(f + g)(f' + g') = (f + 1_c)(1_b + g)(f' + 1_d)(1_{c'} + g') = (f + 1_c)(f' + g)(1_{c'} + g') = (f + 1_c)(f' + gg') = ff' + gg'$. □

1.3 Morfisme

Definiţia 6 *Fie N şi N' două csm(n). Se numeşte* morfism de csm(n) *un functor $F\colon N \longrightarrow N'$ care pe obiecte este un morfism de monoizi şi comută cu suma morfismelor.*

Definiţia 7 *Fie M un monoid.*
*1. Numim M-*csm(n) *orice csm(n) N cu $Ob(N) = M$.*
2. Fie N şi N' două M-csm(n). Un morfism de M-csm(n) este un morfism de csm(n) $F\colon N \longrightarrow N'$ cu proprietatea că $F(a) = a$ pentru orice $a \in M$.

2 Categorii strict monoidale simetrice

2.1 Cazul nepermutabil

Definiţia 8 *O csmn N se numeşte* simetrică *(csmns) dacă oricare ar fi obiectele a şi b se defineşte un morfism distins ${}^a\mathsf{X}^b \in N(ab, ba)$ (*transpoziţia bloc*), care verifică axiomele:*

1. ${}^a\mathsf{X}^{bc} = ({}^a\mathsf{X}^b + 1_c)(1_b + {}^a\mathsf{X}^c)$;
2. ${}^c\mathsf{X}^a(f + 1_c){}^b\mathsf{X}^c = 1_c + f$ pentru $f\colon a \longrightarrow b$;
3. ${}^a\mathsf{X}^b\, \mathbf{P}\, f$ pentru orice morfism f.

Propoziţia 4 *Într-o csmns sunt adevărate următoarele proprietăţi:*

1. ${}^a\mathsf{X}^{bb}\mathsf{X}^a = 1_{ab}$;
2. ${}^c\mathsf{X}^a(f + 1_c) = (1_c + f){}^c\mathsf{X}^b$ pentru $f\colon a \longrightarrow b$;
3. $(f + 1_c){}^b\mathsf{X}^c = {}^a\mathsf{X}^c(1_c + f)$ pentru $f\colon a \longrightarrow b$;
4. ${}^a\mathsf{X}^c(1_c + f){}^c\mathsf{X}^b = f + 1_c$ pentru $f\colon a \longrightarrow b$;
5. ${}^{bc}\mathsf{X}^a = (1_b + {}^c\mathsf{X}^a)({}^b\mathsf{X}^a + 1_c)$;
6. $(1_a + g)(f + 1_d) = {}^a\mathsf{X}^c(g + f){}^d\mathsf{X}^b$ pentru $f\colon a \longrightarrow b$ şi $g\colon c \longrightarrow d$;
7. ${}^a\mathsf{X}^\lambda = 1_a$.

Demonstraţie:

1. Conform axiomei (2) obţinem ${}^a\mathsf{X}^{bb}\mathsf{X}^a = {}^a\mathsf{X}^b 1_{ab}{}^b\mathsf{X}^a = {}^a\mathsf{X}^b(1_b + 1_a){}^b\mathsf{X}^a = 1_a + 1_b = 1_{ab}$;
2. ${}^c\mathsf{X}^a(f + 1_c) = {}^c\mathsf{X}^a(f + 1_c){}^b\mathsf{X}^{cc}\mathsf{X}^b = (1_c + f){}^b\mathsf{X}^c$;
3. $(f + 1_c){}^b\mathsf{X}^c = {}^a\mathsf{X}^{cc}\mathsf{X}^a(f + 1_c){}^b\mathsf{X}^c = {}^a\mathsf{X}^c(1_c + f)$;
4. ${}^a\mathsf{X}^c(1_c + f){}^c\mathsf{X}^b = (f + 1_c){}^b\mathsf{X}^{cc}\mathsf{X}^b = (f + 1_c)1_{bc} = f + 1_c$;
5. $(1_b + {}^c\mathsf{X}^a)({}^b\mathsf{X}^a + 1_c) = (1_b + {}^c\mathsf{X}^a)({}^b\mathsf{X}^a + 1_c){}^a\mathsf{X}^{bcbc}\mathsf{X}^a = (1_b + {}^c\mathsf{X}^a)({}^b\mathsf{X}^a + 1_c)({}^a\mathsf{X}^b + 1_c)(1_b + {}^a\mathsf{X}^c){}^{bc}\mathsf{X}^a = 1_{bca}{}^{bc}\mathsf{X}^a = {}^{bc}\mathsf{X}^a$;
6. $(1_a + g)(f + 1_d) = {}^a\mathsf{X}^c(g + 1_a){}^d\mathsf{X}^{aa}\mathsf{X}^d(1_d + f){}^d\mathsf{X}^b = {}^a\mathsf{X}^c(g + 1_a)(1_d + f){}^d\mathsf{X}^b = {}^a\mathsf{X}^c(g + f){}^d\mathsf{X}^b$.
7. Conform axiomei (1) obţinem ${}^a\mathsf{X}^\lambda = ({}^a\mathsf{X}^\lambda + 1_\lambda)(1_\lambda + {}^a\mathsf{X}^\lambda) = {}^a\mathsf{X}^\lambda{}^a\mathsf{X}^\lambda$
prin urmare compunând cu ${}^\lambda\mathsf{X}^a$ rezultă concluzia. □

Propoziţia 5 *Într-o csmns relaţia de permutare $\mathbf{P}$ are următoarele proprietăţi suplimentare pentru $f\colon a \longrightarrow b$ şi $g\colon c \longrightarrow d$:*

1. *$f\mathbf{P}g$ dacă şi numai dacă $f + g = {}^a\mathsf{X}^c(g + f){}^d\mathsf{X}^b$,*
2. *$f\mathbf{P}g$ implică $g\mathbf{P}f$,*
3. *$f\mathbf{P}g$ implică $f + 1_u\mathbf{P}g$.*

Demonstraţie:

1. Din proprietatea 6 de mai sus rezultă $f\mathbf{P}g \Leftrightarrow f+g=(1_a+g)(f+1_d) \Leftrightarrow f+g={}^a\mathsf{X}^c(g+f)^d\mathsf{X}^b$;
2. Din $f\mathbf{P}g$ deducem $f+g={}^a\mathsf{X}^c(g+f)^d\mathsf{X}^b$. Compunând la stânga cu ${}^c\mathsf{X}^a$ şi la dreapta cu ${}^b\mathsf{X}^d$ obţinem $g+f={}^c\mathsf{X}^a(f+g)^b\mathsf{X}^d$, deci $g\mathbf{P}f$.
3. Deoarece ${}^a\mathsf{X}^u\mathbf{P}g$, $1_u+f\mathbf{P}g$ şi ${}^u\mathsf{X}^b\mathbf{P}g$ deducem ${}^a\mathsf{X}^u(1_u+f)^u\mathsf{X}^b\mathbf{P}g$ deci $f+1_u\mathbf{P}g$. □

2.2 Cazul permutabil

Definiţia 9 *O categorie N se numeşte* categorie strict monoidală simetrică *(csms) dacă este csm şi oricare ar fi obiectele a şi b se defineşte un morfism distins ${}^a\mathsf{X}^b \in N(ab,ba)$ (*transpoziţia bloc*), care verifică axiomele:*

1. ${}^a\mathsf{X}^{bc}=({}^a\mathsf{X}^b+1_c)(1_b+{}^a\mathsf{X}^c)$;
2. ${}^c\mathsf{X}^a(f+g)^b\mathsf{X}^d=g+f$ pentru $f\colon a\to b$ şi $g\colon c\longrightarrow d$.

Observăm că orice csms este o csmns.

Propoziţia 6 *O csmns este o csms dacă şi numai dacă $f\mathbf{P}g$ oricare ar fi morfismele f şi g.*

Demonstraţie: O csmns în care orice pereche de morfisme permută este o csm. În plus proprietatea 2 din definiţia csms este o consecinţă a ipotezei şi a proprietăţii 6 din Propoziţia 4. Deci o csmns în care orice pereche de morfisme permută este o csms.

Pentru reciprocă observăm că deoarece o csms este o csm orice pereche de morfisme permută. □

2.3 $a\alpha$-morfisme

Observaţia 4 *1. Dacă G este o mulţime de morfisme dintr-o csms(n) N, atunci subcsms(n) generată de G coincide cu subcsm(n) generată de $G\cup\{{}^a\mathsf{X}^b|a,b\in Ob(N)\}$.*
2. Cea mai mică subcsms(n) este subcsms(n) generată de $\emptyset$ şi este formată din toate compunerile de morfisme de forma $1_a+{}^b\mathsf{X}^c+1_d$. Aceste morfisme se numesc $a\alpha$-morfisme.

Propoziţia 7 *Într-o csmns N, un $a\alpha$-morfism permută cu orice alt morfism.*

Demonstraţie: Orice $a\alpha$-morfism este o compunere de morfisme de forma $1_a+{}^b\mathsf{X}^c+1_d$. Fie f un morfism oarecare. Conform axiomei (4) din definiţia csmns rezultă ${}^b\mathsf{X}^c\mathbf{P}f$. Aplicând Propoziţia 2, punctul (2) şi Propoziţia 5, punctul (2) rezultă $1_a+{}^b\mathsf{X}^c+1_d\mathbf{P}f$. Folosind Propoziţia 2, punctul (4) rezultă că orice $a\alpha$-morfism permută cu f. □

Propoziţia 8 *Într-o csmns N, $a\alpha$-morfismele formează o csms $N_{a\alpha}$.*

Demonstraţie: Din propoziţia precedentă rezultă că orice două $a\alpha$-morfisme permută. Aplicând Propoziţia 6 obţinem că $a\alpha$-morfismele formează o csms. □

Propoziţia 9 *Orice $a\alpha$-morfism este izomorfism şi inversul său este tot un $a\alpha$-morfism.*

Demonstraţie: Într-o csms dacă f şi g sunt izomorfisme atunci $f+g$ şi fg sunt izomorfisme. În plus, $(f+g)^{-1}=f^{-1}+g^{-1}$ şi $(fg)^{-1}=g^{-1}f^{-1}$. Rezultă că un morfism de forma $1_a+{}^b\mathsf{X}^c+1_d$ este izomorfism şi inversul său este $1_a+{}^c\mathsf{X}^b+1_d$. Propoziţia este demonstrată deoarece $a\alpha$-morfismele sunt compuneri finite de morfisme de această formă. □

2.4 Morfisme

Definiţia 10 *Fie N şi N' două csms(n). Se numeşte* morfism de csms(n) *un morfism de csm(n) $F\colon N\longrightarrow N'$ care comută cu transpoziţia bloc:*

$$F({}^a\mathsf{X}^b)={}^{F(a)}\mathsf{X}^{F(b)}.$$

Definiţia 11 *Fie M un monoid.*
*1. Numim M-*csms(n) *orice csms(n) N cu $Ob(N)=M$.*
2. Fie N şi N' două M-csms(n)-uri. Un morfism de M-csms(n) este un morfism de csms(n) $F\colon N\longrightarrow N'$ cu proprietatea că $F(a)=a$ pentru orice $a\in M$.

3 Fluxuri şi bifluxuri

Definiţia 12 *Se numeşte* flux *o csmns B în care pentru fiecare $a \in B$ se defineşte o operaţie $\uparrow^a\_ \colon B(ab, ac) \longrightarrow B(b, c)$ care satisface următoarele axiome:*

1. $g(\uparrow^a f) = \uparrow^a((1_a + g)f)$ *pentru* $f \colon ab \longrightarrow ac$ *şi* $g \colon d \longrightarrow b$;
2. $(\uparrow^a f)g = \uparrow^a(f(1_a + g))$ *pentru* $f \colon ab \longrightarrow ac$ *şi* $g \colon c \longrightarrow d$;
3. $(\uparrow^a f) + 1_d = \uparrow^a(f + 1_d)$ *pentru* $f \colon ab \longrightarrow ac$;
4. $\uparrow^{ab} f = \uparrow^b \uparrow^a f$ *pentru* $f \colon abc \longrightarrow abd$;
5. $\uparrow^{ab}((^a\mathsf{X}^b + 1_c)f) = \uparrow^{ba}(f(^a\mathsf{X}^b + 1_d))$ *pentru* $f \colon bac \longrightarrow abd$;
6. $\uparrow^a 1_a = 1_\lambda$;
7. $\uparrow^{aa} {}^a\mathsf{X}^a = 1_a$.

Propoziţia 10 *În orice flux sunt adevărate următoarele egalităţi:*

1. $(\uparrow^a f) + g = \uparrow^a(f + g)$ *pentru* $f \colon ab \longrightarrow ac$ *şi* $g \colon u \longrightarrow v$;
2. $g + \uparrow^a f = \uparrow^a[(^a\mathsf{X}^u + 1_b)(g + f)(^v\mathsf{X}^a + 1_c)]$
   pentru $f \colon ab \longrightarrow ac$ *şi* $g \colon u \longrightarrow v$;
3. $\uparrow^\lambda f = f$ *pentru orice morfism* f;
4. $\uparrow^{ab} f = \uparrow^{ba}((^b\mathsf{X}^a + 1_c)f(^a\mathsf{X}^b + 1_d))$ *pentru orice* $f \colon abc \longrightarrow abd$.

Demonstraţie: 1. Folosind axiomele (3) şi (2) din definiţia feedbackului obţinem
$\uparrow^a f + g = (\uparrow^a f + 1_u)(1_c + g) = \uparrow^a(f + 1_u)(1_c + g) = \uparrow^a[(f + 1_u)(1_{ac} + g)] = \uparrow^a(f + g)$;

2. $$\begin{aligned} g + \uparrow^a f &= (g + 1_b)(1_v + \uparrow^a f) = (g + 1_b)^v\mathsf{X}^b(\uparrow^a f + 1_v)^c\mathsf{X}^v \\ &= \uparrow^a[(1_a + g + 1_b)(1_a + {}^v\mathsf{X}^b)(f + 1_v)(1_a + {}^c\mathsf{X}^v)] \\ &= \uparrow^a[({}^a\mathsf{X}^u(g + 1_a)^v\mathsf{X}^a + 1_b)(1_a + {}^v\mathsf{X}^b)^{ab}\mathsf{X}^v(1_v + f)^v\mathsf{X}^{ac}(1_a + {}^c\mathsf{X}^v)] \\ &= \uparrow^a[({}^a\mathsf{X}^u + 1_b)(g + 1_{ab})({}^v\mathsf{X}^a + 1_b)(1_a + {}^v\mathsf{X}^b)^{ab}\mathsf{X}^v(1_v + f)({}^v\mathsf{X}^a + 1_c)(1_a + {}^v\mathsf{X}^c)(1_a + {}^c\mathsf{X}^v)] \\ &= \uparrow^a[({}^a\mathsf{X}^u + 1_b)(g + 1_{ab})^v\mathsf{X}^{abab}\mathsf{X}^v(1_v + f)({}^v\mathsf{X}^a + 1_c)1_{avc}] \\ &= \uparrow^a[({}^a\mathsf{X}^u + 1_b)(g + 1_{ab})(1_v + f)({}^v\mathsf{X}^a + 1_c)] \\ &= \uparrow^a[({}^a\mathsf{X}^u + 1_b)(g + f)({}^v\mathsf{X}^a + 1_c)]; \end{aligned}$$

3. $\uparrow^\lambda f = \uparrow^\lambda(1_\lambda + f) = \uparrow^\lambda 1_\lambda + f = 1_\lambda + f = f$;
4. $\uparrow^{ab} f = \uparrow^{ab}(({}^a\mathsf{X}^b + 1_c)({}^b\mathsf{X}^a + 1_c)f) = \uparrow^{ba}(({}^b\mathsf{X}^a + 1_c)f({}^a\mathsf{X}^b + 1_d))$. □

Propoziţia 11 *Dacă morfismele* $f \colon ab \longrightarrow dc$ *şi* $g \colon d \longrightarrow a$ *permută, atunci*

$$\uparrow^a(f(g + 1_c)) = \uparrow^d((g + 1_b)f)$$

Demonstraţie: $\uparrow^a(f(g + 1_c)) = \uparrow^a(f(\uparrow^{dd}{}^d\mathsf{X}^d + 1_c)(g + 1_c)) =$
$\uparrow^a\uparrow^d((1_d + f)({}^d\mathsf{X}^d + 1_c)(1_d + g + 1_a)) = \uparrow^{da}((1_d + f)(g + 1_{dc})({}^a\mathsf{X}^d + 1_c)) =$
$\uparrow^{ad}(({}^a\mathsf{X}^d + 1_b)(g + f)) = \uparrow^d\uparrow^a(({}^a\mathsf{X}^d + 1_b)(g + 1_{ab})(1_a + f)) = \uparrow^d(\uparrow^a[(1_a + g + 1_b)({}^a\mathsf{X}^a + 1_b)]f) = \uparrow^d((g + 1_b)f)$. □

Definiţia 13 *Fie B şi B' două fluxuri. Un functor $H \colon B \longrightarrow B'$ este* morfism de fluxuri *dacă este morfism de csmns care comută cu feedbackul:*

$$H(\uparrow^a f) = \uparrow^{H(a)} H(f).$$

Definiţia 14 *În orice flux B se poate defini un feedback la dreapta în modul următor:*

$$f\uparrow^a \stackrel{def}{=} \uparrow^a({}^a\mathsf{X}^b f {}^c\mathsf{X}^a) \text{ pentru } f \in B(ba, ca).$$

Feedback-ul la dreapta are următoarele proprietăţi:

1. $[(g + 1_a)f]\uparrow^a = g(f\uparrow^a)$ pentru $f \colon ba \longrightarrow ca$ şi $g \colon u \longrightarrow b$;
2. $[f(h + 1_c)]\uparrow^a = (f\uparrow^a)h$ pentru $f \colon ba \longrightarrow ca$ şi $h \colon c \longrightarrow v$;

3. $g + f\uparrow^a = (g + f)\uparrow^a$ pentru $f\colon ba \longrightarrow ca$ şi $g\colon u \longrightarrow v$;
4. $f\uparrow^b\uparrow^a = f\uparrow^{ab}$ pentru $f\colon cab \longrightarrow dab$;
5. $(f(1_d + g))\uparrow^b = ((1_c + g)f)\uparrow^a$ dacă $f\mathbf{P}g$, $f\colon cb \longrightarrow da$ şi $g\colon a \longrightarrow b$;
6. ${}^a\mathsf{X}^a\uparrow^a = 1_a$;
7. $1_a\uparrow^a = 1_\lambda$;
8. $f\uparrow^a + g = [(1_b + {}^u\mathsf{X}^a)(f + g)(1_c + {}^a\mathsf{X}^v)]\uparrow^a$ pentru $f\colon ba \longrightarrow ca$ şi $g\colon u \longrightarrow v$;
9. $f\uparrow^\lambda = f$ pentru orice morfism.

Conceptul de flux poate fi definit axiomatizând feedbackul la dreapta în locul feedbackului la stânga. În acest caz definiţia feedbackului la stânga este:

$\uparrow^a f = ({}^b\mathsf{X}^a f {}^a\mathsf{X}^c)\uparrow^a$, pentru $f \in B(ab, ac)$.

Propoziţia 12 *Are loc următoarea proprietate de legătură între cele două feedback-uri pentru $f\colon acb \longrightarrow adb$:*

$$(\uparrow^a f)\uparrow^b = \uparrow^a (f\uparrow^b)$$

Demonstraţie:

$$\begin{aligned}
(\uparrow^a f)\uparrow^b &= \uparrow^b({}^b\mathsf{X}^c(\uparrow^a f){}^d\mathsf{X}^b)\\
&= \uparrow^b\uparrow^a[(1_a + {}^b\mathsf{X}^c)f(1_a + {}^d\mathsf{X}^b)]\\
&= \uparrow^{ab}[(1_a + {}^b\mathsf{X}^c)f(1_a + {}^d\mathsf{X}^b)]\\
&= \uparrow^{ba}[({}^b\mathsf{X}^a + 1_c)(1_a + {}^b\mathsf{X}^c)f(1_a + {}^d\mathsf{X}^b)({}^a\mathsf{X}^b + 1_d)]\\
&= \uparrow^a\uparrow^b[{}^b\mathsf{X}^{ac} f {}^{ad}\mathsf{X}^b]\\
&= \uparrow^a(f\uparrow^b).\square
\end{aligned}$$

Definiţia 15 *Se numeşte* biflux *B un flux peste o csms.*

Observaţia 5 *Un morfism de bifluxuri este un morfism de fluxuri între două bifluxuri.*

4 Exemple

Exemplul 1 *Rel*

Relaţiile finite Rel cu operaţiile definite în lecţiile anterioare formează un biflux în care monoidul obiectelor este monoidul numerelor naturale $(N, +, 0)$.

Exemplul 2 *Rel_S*

Un alt exemplu de biflux este Rel_S, teoria relaţiilor finite S-sortate. Monoidul obiectelor este S^*. Un cuvânt $a \in S^*$ se scrie $a = a_1 a_2 \cdots a_{|a|}$ unde $|a|$ este lungimea sa şi a_i sunt literele sale. Pentru $a, b \in S^*$ prin definiţie

$$\mathsf{Rel}_S(a, b) = \{f \subseteq [|a|] \times [|b|] \mid (i, j) \in f \text{ implică } a_i = b_j\}$$

Operaţiile în $\mathsf{Rel_S}$ sunt:

$$fg = \{(i, k) | (\exists j)[(i, j) \in f \text{ şi } (j, k) \in g]\} ,$$

$$1_a = \{(i, i) | i \in [|a|]\} ,$$

$$f + g = f \cup \{(|a| + i, |b| + j) \mid (i, j) \in g\}, \text{ unde } f : a \to b ,$$

$${}^a\mathsf{X}^b = \{(i, |b| + i) | i \in [|a|]\} \cup \{(|a| + i, i) | i \in [|b|]\} ,$$

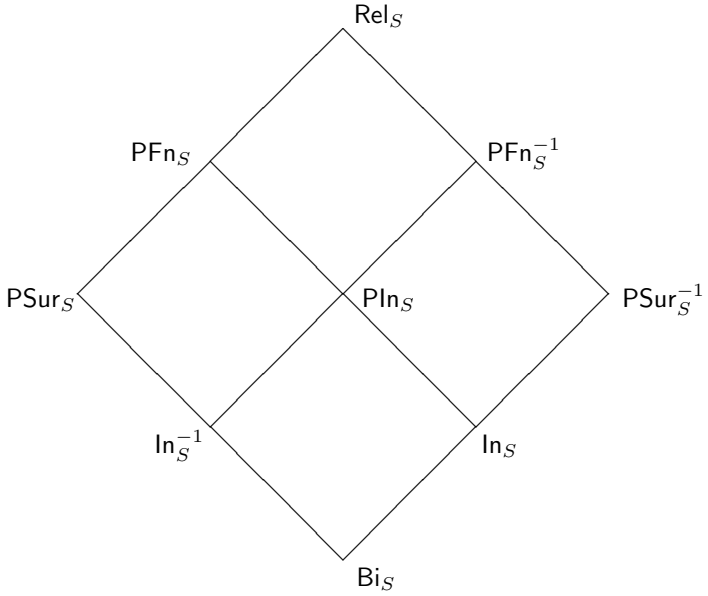


Figure 1: Relaţii închise la feedback

Pentru $s \in S$ şi $f \in \mathsf{Rel}_S(as, bs)$

$$f\uparrow^s = \{(i,j) \in [|a|] \times [|b|] \mid (i,j) \in f \text{ sau } [(i,|b|+1) \in f \text{ şi } (|a|+1,j) \in f]\}\ .$$

În acest caz $\uparrow^a$ este definit prin inducţie folosind $f\uparrow^\lambda = f$ (unde λ este cuvântul vid) şi proprietatea 4 a feedback-ului la dreapta.

Există subbifluxuri interesante ale lui Rel_S. PFn_S este bifluxul funcţiilor parţiale finite S-sortate. Demonstraţia că PFn_S este închis la feedback a fost dată, în cazul fără sorturi, în lecţia precedentă.

Pentru $f \in \mathsf{Rel}_S(a,b)$ notăm

$$f^{-1} = \{(u,v) \mid (v,u) \in f\} \in \mathsf{Rel}_S(b,a).$$

Spunem că f^{-1} este inversa relaţiei f. Atragem atenţia că inversa unei funcţii s-ar putea să nu mai fie o funcţie. Asemănător se arată că inversele de funcţii parţiale formează un biflux notat PFn_S^{-1}. Intersecţia lui PFn_S cu PFn_S^{-1} este bifluxul injecţiilor parţiale notat PIn_S.

Probăm că surjecţiile parţiale PSur_S sunt închise la feedback. Fie $f \in \mathsf{PFn}_S(as, bs)$ unde $s \in S$. Pentru orice $j \in [|b|]$ deoarece f este surjectivă există $i \in [|as|]$ cu $(i,j) \in f$. Dacă $i \in [|a|]$ atunci, $(i,j) \in f\uparrow^s$. Dacă $(|a|+1,j) \in f$ din surjectivitatea lui f deducem exixtenţa lui $k \in [|a|]$ cu proprietatea $(k,|b|+1) \in f$. prin urmare $(k,j) \in f\uparrow^s$.

Asemănător se arată că inversele de surjecţii parţiale formează un biflux notat PSur_S^{-1}.

Restul exemplelor se obţin prin intersecţii. Bifluxul injecţiilor finite S-sortate, notat In_S, este intersecţia lui PIn_S cu PSur_S^{-1}. Bifluxul inverselor de injecţiilor finite S-sortate, notat In_S^{-1}, este intersecţia lui PIn_S cu PSur_S. Bifluxul bijecţiilor finite S-sortate Bi_S este intersecţia lui In_S cu In_S^{-1}.

Csms Sur_S a surjecţiilor finite S-sortate şi csms Fn_S a funcţiilor finite S-sortate nu sunt subbifluxuri deoarece nu sunt închise la feedback.

Se observă că în cazul particular când S are exact un element, identificăm S^* cu monoidul aditiv al întregilor nenegativi şi, prin urmare, Rel_S poate fi identificat cu Rel.

Exemplul 3 Pfn **şi** Pfn_S **pentru programe deterministe.**

Subteoria funcţiilor finite parţiale a lui Rel, notată Pfn şi definită prin familia de mulţimi

$$\mathsf{Pfn}(m,n) = \{f \mid f : [m] \nrightarrow [n] \text{ funcţie parţială}\}, \text{ pentru } m,n \in \mathsf{N}$$

este închisă la operaţiile menţionate mai sus. În schimb, teoria Fn nu este închisă la feedback, prin urmare nu este convenabil să folosim Fn ca teorie suport pentru programele deterministe. Dacă f este unica funcţie din $\mathsf{Fn}(2,1)$, atunci aplicând feedbackul ar trebui ca $f\uparrow \in \mathsf{Fn}(1,0)$, ceea ce este imposibil, deoarece $\mathsf{Fn}(1,0) = \emptyset$. Lucrul cu Pfn ca teorie suport în cazul determinist este echivalent cu trecerea de la programe la programe parţiale. Un program parţial se obţine dintr-un program obişnuit ştergând unele săgeţi. Lipsa unei săgeţi este interpretată ca o conexiune la o buclă fără ieşire. Bifluxul Pfn_S al funcţiilor parţiale finite S-sortate este subbiflux al lui Rel_S. Dacă S are exact un element Pfn_S poate fi identificat cu Pfn.

Exemplul 4 Modelele semantice fundamentale $Rel(S)$ şi $Pfn(S)$.

Modelul de bază pentru studiul semanticii în cazul determinist a fost introdus de C. C. Elgot.

Fie S mulţimea stărilor memoriei dintr-o maşină de calcul. Un program determinist F cu m intrări şi n ieşiri este interpretat cu ajutorul unei interpretări I ca o funcţie parţială

$$F_I : [m] \times S \multimap [n] \times S$$

cu semnificaţia: "$F_I(i,s)$ este definită şi este egală cu (j,s')" dacă şi numai dacă "execuţia programului obţinut interpretând F via I începe la intrarea i a programului cu starea iniţială a memoriei s şi se opreşte la ieşirea j a programului, starea memoriei rezultată fiind s'."

Dacă, pentru $m, n \in \mathsf{N}$, notăm

$$\mathsf{Pfn}(S)(m,n) = \{f | f : [m] \times S \multimap [n] \times S \text{ funcţie parţială }\},$$

obţinem teoria Pfn(S), modelul semantic de bază în cazul determinist.

Se observă că în cazul particular când S are exact un element, Pfn(S) poate fi identificat cu Pfn. În acest caz starea memoriei rămâne neschimbată.

Într-un mod similar se introduce modelul semantic pentru cazul nedeterminist. Un program nedeterminist F cu m intrări şi n ieşiri este interpretat cu ajutorul unei interpretări I ca o relaţie $F_I \subseteq ([m] \times S) \times ([n] \times S)$ cu semnificaţia: "$((i,s),(j,s')) \in F_I$" dacă şi numai dacă "execuţia programului obţinut interpretând F via I începe la intrarea i a programului cu starea iniţială a memoriei s şi poate să se oprească la ieşirea j a programului, starea memoriei rezultată fiind s'."

Dacă, pentru $m, n \in \mathsf{N}$, notăm

$$\mathsf{Rel}(S)(m,n) = \{r | r \subseteq ([m] \times S) \times ([n] \times S) \text{ relaţie }\},$$

obţinem teoria Rel(S), modelul semantic de bază pentru cazul nedeterminist.

Analog, dacă S are exact un element, Rel(S) poate fi identificată cu Rel.

În Rel(S), operaţiile care ne interesează (compunerea, suma şi feedback-ul) au următoarele definiţii.

Pentru $r \in \mathsf{Rel}(S)(m,n)$ şi $r' \in \mathsf{Rel}(S)(n,p)$ compunerea $r \cdot r' \in \mathsf{Rel}(S)(m,p)$ se defineşte astfel:

$$r \cdot r' = \{((i,s),(j,s')) | (\exists (i_0,s_0) \in [n] \times S)[((i,s),(i_0,s_0)) \in r \text{ şi } ((i_0,s_0),(j,s')) \in r']\}.$$

Pentru $r \in \mathsf{Rel}(S)(m,n)$ şi $r' \in \mathsf{Rel}(S)(p,q)$ suma $r + r' \in \mathsf{Rel}(S)(m+p, n+q)$ se defineşte astfel:

$$r + r' = r \cup \{((m+i,s),(n+j,s')) | ((i,s),(j,s')) \in r\}.$$

Pentru a defini feedbackul observăm că orice relaţie $r \in \mathsf{Rel}(S)(m,n)$ este dată de o familie de relaţii $r_{i,j} \subseteq S \times S$, pentru $i \in [m]$ şi $j \in [n]$, unde

$$r_{i,j} = \{(s,s') | ((i,s),(j,s')) \in r\}.$$

În egalitatea de mai sus se vede legătura între relaţiile care definesc semantica şi matricile care definesc semantica utilizate în lecţiile precedente. Cela două forme de prezentare a semanticii pentru programele nedeterministe sunt echivalente.

Notăm cu r^* închiderea reflexivă şi tranzitivă a relaţiei $r \subseteq S \times S$, adică $r^* = 1_S \cup r \cup r^2 \cup \cdots$, unde $1_S = \{(s,s) | s \in S\}$.

Pentru o relaţie $r \in \mathsf{Rel}(S)(m+1, n+1)$ definim feedbackul $r\uparrow \in \mathsf{Rel}(S)(m,n)$ astfel:

$$(r\uparrow)_{i,j} = r_{i,j} \cup (r_{i,n+1} \cdot r^*_{m+1,n+1} \cdot r_{m+1,j}), \text{ pentru } i \in [m], j \in [n].$$

Se obţine astfel că $\mathsf{Rel}(S)$ este biflux şi că $\mathsf{Pfn}(S)$ este subbiflux al lui $\mathsf{Rel}(S)$.

Se observă că se poate defini o scufundare naturală a lui Rel în $\mathsf{Rel}(S)$, dată de aplicaţia

$$H : \mathsf{Rel} \to \mathsf{Rel}(S), H(r) = \{((i,s),(j,s)) | (i,j) \in r, s \in S\}.$$

Evident H este morfism de bifluxuri. Dacă relaţia r este privită ca program nedeterminist fără instrucţiuni atunci, $H(r)$ ne dă comportarea programului.

4 Structura algebrică a modelelor semantice fundamentale

Adela Mihăiţă

February 14, 2010

1 Categoria matricilor cu elemente dintr-un semiinel

Matricile cu elemente din semiinelul $(R, \cup, 0, \cdot, 1)$ sunt organizate ca o categorie $[R]$ după cum urmează. Obiectele categoriei $[R]$ sunt numerele naturale. Pentru orice pereche de numere naturale a şi b mulţimea morfismelor de la a la b notată $[R](a, b)$ este prin definiţie mulţimea matricilor cu a linii şi b coloane. Compunerea morfismelor este prin definiţie produsul obişnuit al matricilor. Matricea unitate cu n linii şi n coloane, notată cu 1_n, este morfismul identitate corespunzător obiectului n.

Observaţia 1.1 *Matricile pătratice peste un semiinel formează semiinel cu adunarea şi produsul de matrici.*

Vom nota cu 0_b^a matricea nulă(plină numai cu zerouri) cu a linii şi b coloane.

Definiţia 1.2 *Se numeşte* categorie strict monoidală *o categorie N care satisface următoarele proprietăţi :*

– *obiectele formează un monoid* $(Ob(N), \cdot, \lambda)$.

– *pentru orice obiecte a, b, c, d se dă o operaţie numită* **sumă**,

$+ : N(a, b) \times N(c, d) \longrightarrow N(ac, bd)$ care verifică următoarele axiome :

1. f + (g + h)= (f + g) + h
2. f + 1_λ = 1_λ + f = f
3. $1_a + 1_b = 1_{ab}$
4. (f +g) $(f' + g') = ff' + gg'$ *pentru* f : a $\longrightarrow$ b , g : c $\longrightarrow$ d , f' : b $\longrightarrow$ b' şi g' : d $\longrightarrow$ d' .

Cuvântul sumă va fi folosit în continuare pentru o altă operaţie cu matrici care este mult diferită de adunarea obişnuită a matricilor.

Definiţia 1.3 Suma matricilor $A \in [R](a, b)$ şi $B \in [R](c, d)$ este matricea $A + B \in [R](a + c, b + d)$ defintă prin

$$A + B = \begin{pmatrix} A & 0_d^a \\ 0_b^c & B \end{pmatrix}$$

Propoziţie. Cu suma definită mai sus categoria $[R]$ a matricilor peste un semiinel R este strict monoidală.

Demonstraţie Deoarece primele 3 axiome din definiţia 1.2 sunt evidente, demonstrăm numai ultima axiomă.

Pentru $A : a \to b, B : c \to d, C : b \to e$ şi $D : d \to f$

$$(A + B)(C + D) = \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} \begin{pmatrix} C & 0 \\ 0 & D \end{pmatrix} = \begin{pmatrix} AC & 0 \\ 0 & BD \end{pmatrix} = AC + BD. \ \square$$

Definiţia 1.4 *O categorie strict monoidală N se numeşte* **simetrică** *dacă oricare ar fi obiectele a şi b este dat un morfism distins* ${}^a\mathbf{X}^b \in N(ab, ba)$, transpoziţia bloc, *care verifică axiomele :*

1. ${}^a\mathbf{X}^{bc} = ({}^a\mathbf{X}^b + 1_c)(1_b + {}^a\mathbf{X}^c)$;

2. ${}^c\mathbf{X}^a(f+1_c)^b\mathbf{X}^c = 1_c + f$ pentru $f : a \longrightarrow b$. □

Pentru orice pereche de numere naturale a şi b definim matricea ${}^a\mathsf{X}^b$ prin

$$ {}^a\mathsf{X}^b = \begin{pmatrix} 0_b^a & 1_a \\ 1_b & 0_a^b \end{pmatrix}. $$

Propoziţie 1.5 *Cu transpunerea bloc definită mai sus categoria $[R]$ a matricilor peste un semiinel R este strict monoidală simetrică .*

Demonstraţie Arătăm că transpunerea bloc definită pentru matrici verifică cele 2 axiome din definiţia categoriilor strict monoidale simetrice :

1. $({}^a\mathbf{X}^b + 1_c)(1_b + {}^a\mathbf{X}^c) = \begin{pmatrix} 0_b^a & 1_a & 0_c^a \\ 1_b & 0_a^b & 0_c^b \\ 0_b^c & 0_a^c & 1_c \end{pmatrix} \begin{pmatrix} 1_b & 0_c^b & 0_a^b \\ 0_b^a & 0_c^a & 1_a \\ 0_b^c & 1_c & 0_a^c \end{pmatrix} = \begin{pmatrix} 0_b^a & 0_c^a & 1_a \\ 1_b & 0_c^b & 0_a^b \\ 0_b^c & 1_c & 0_a^c \end{pmatrix} =$

$$ = \begin{pmatrix} 0_{bc}^a & 1_a \\ 1_{bc} & 0_a^{bc} \end{pmatrix} = {}^a\mathbf{X}^{bc} $$

2. ${}^c\mathbf{X}^a(f+1_c)^b\mathbf{X}^c = \begin{pmatrix} 0_a^c & 1_c \\ 1_a & 0_c^a \end{pmatrix} \begin{pmatrix} f & 0_c^a \\ 0_b^c & 1_c \end{pmatrix} \begin{pmatrix} 0_c^b & 1_b \\ 1_c & 0_b^c \end{pmatrix} = \begin{pmatrix} 0_b^c & 1_c \\ f & 0_c^a \end{pmatrix} \begin{pmatrix} 0_c^b & 1_b \\ 1_c & 0_b^c \end{pmatrix} =$

$$ = \begin{pmatrix} 1_c & 0_b^c \\ 0_c^a & f \end{pmatrix} = 1_c + f \text{ unde } f : a \to b. \square $$

2 Axiomatizarea repetiţiei pentru matrici peste un semiinel

In continuare ne ocupăm de matrici peste un semiinel R în care pentru orice $n \in \mathbf{N}$ este dată o operaţie "repetiţie" :

$$ \_^* : [R](n,n) \longrightarrow [R](n,n) $$

care satisface axiomele de mai jos unde $\cup$ este adunarea uzuală a matricilor.

R1 $(A \cup B)^* = A^*(BA^*)^*$ unde A şi B sunt matrici pătratice cu n linii şi n coloane,

R2 $(AB)^* = 1_n \cup A(BA)^*B$ oricare ar fi $A \in [R](n,m)$ şi $B \in [R](m,n)$. □

2.1 Proprietăţile repetiţiei axiomatizate

Probăm în continuare mai multe consecinţe ale acestor axiome :

R3 $A^* = 1_n \cup AA^*$
R4 $B^* = 1_n \cup B^*B$
R5 $B(AB)^* = (BA)^*B$
R6 $(0_a^a)^* = 1_a$

Demonstraţie Primele două consecinţe se obţin din **R2** pentru B, respectiv A, o identitate.
Probăm **R5**.

$$ B(AB)^* \overset{\mathbf{R2}}{=} B(1_n \cup A(BA)^*B) = B \cup BA(BA)^*B = (1_n \cup BA(BA)^*)B = (BA)^*B $$

cea din urmă egalitate rezultând din **R3** prin înlocuirea lui A cu BA.
Demonstraţia pentru ultima consecinţă **R6** este $(0_a^a)^* \overset{R3}{=} 1_a \cup 0_a^a(0_a^a)^* = 1_a \cup 0_a^a = 1_a$. □

In continuare sunt prezentate două leme ce vor fi folosite în demonstraţia teoremei 2.1.

Lema1. Dacă A este o matrice pătratică atunci $\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix}^* = \begin{pmatrix} A^* & A^*B \\ 0 & 1 \end{pmatrix}$

Demonstraţie.

$$\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix}^* = \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} A & B \end{pmatrix} \right)^* \overset{\mathbf{R2}}{=} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cup \begin{pmatrix} 1 \\ 0 \end{pmatrix} \left(\begin{pmatrix} A & B \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right)^* \begin{pmatrix} A & B \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cup \begin{pmatrix} 1 \\ 0 \end{pmatrix} A^* \begin{pmatrix} A & B \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cup \begin{pmatrix} A^*A & A^*B \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 \cup A^*A & A^*B \\ 0 & 1 \end{pmatrix} \overset{\mathbf{R4}}{=} \begin{pmatrix} A^* & A^*B \\ 0 & 1 \end{pmatrix}$$

Lema2. Dacă D este o matrice pătratică atunci $\begin{pmatrix} 0 & 0 \\ C & D \end{pmatrix}^* = \begin{pmatrix} 1 & 0 \\ D^*C & D^* \end{pmatrix}$

Demonstraţie.

$$\begin{pmatrix} 0 & 0 \\ C & D \end{pmatrix}^* = \left(\begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} C & D \end{pmatrix} \right)^* \overset{\mathbf{R2}}{=} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cup \begin{pmatrix} 0 \\ 1 \end{pmatrix} \left(\begin{pmatrix} C & D \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right)^* \begin{pmatrix} C & D \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cup \begin{pmatrix} 0 \\ 1 \end{pmatrix} D^* \begin{pmatrix} C & D \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cup \begin{pmatrix} 0 & 0 \\ D^*C & D^*D \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ D^*C & 1 \cup D^*D \end{pmatrix} \overset{\mathbf{R4}}{=} \begin{pmatrix} 1 & 0 \\ D^*C & D^* \end{pmatrix}$$

Teorema 2.1 *Dacă A şi D sunt matrici pătratice, atunci* $\begin{pmatrix} A & B \\ C & D \end{pmatrix}^* = \begin{pmatrix} A^* \cup A^*BWCA^* & A^*BW \\ WCA^* & W \end{pmatrix}$ *unde* $W = (CA^*B \cup D)^*$

Demonstraţie.

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}^* = \left(\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix} \cup \begin{pmatrix} 0 & 0 \\ C & D \end{pmatrix} \right)^* \overset{\mathbf{R1}}{=} \begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix}^* \left(\begin{pmatrix} 0 & 0 \\ C & D \end{pmatrix} \begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix}^* \right)^*$$

$$\overset{L1}{=} \begin{pmatrix} A^* & A^*B \\ 0 & 1 \end{pmatrix} \left(\begin{pmatrix} 0 & 0 \\ C & D \end{pmatrix} \begin{pmatrix} A^* & A^*B \\ 0 & 1 \end{pmatrix} \right)^* = \begin{pmatrix} A^* & A^*B \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ CA^* & CA^*B \cup D \end{pmatrix}^*$$

$$\overset{L2}{=} \begin{pmatrix} A^* & A^*B \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ WCA^* & W \end{pmatrix} = \begin{pmatrix} A^* \cup A^*BWCA^* & A^*BW \\ WCA^* & W \end{pmatrix}. \square$$

Propoziţie 2.2 .
Dacă B este o matrice pătratică inversabilă şi A o matrice pătratică atunci $(B^{-1}AB)^* = B^{-1}A^*B$.

Demonstraţie.
$(B^{-1}AB)^* = B^{-1}B(B^{-1}AB)^* = B^{-1}(B((B^{-1}A)B)^*) \overset{R5}{=} B^{-1}((B(B^{-1}A))^*B) = B^{-1}A^*B. \square$

2.2 Feedback

Fie A o matrice cu a linii şi a coloane, $B : a \longrightarrow c$, $C : b \longrightarrow a$ şi $D : b \longrightarrow c$. Prin definiţie

$$\uparrow^a \begin{pmatrix} A & B \\ C & D \end{pmatrix} = D \cup CA^*B$$

se numeşte feedback la stânga.

Definiţia 2.3 *O categorie strict monoidală simetrică se numeşte biflux (basic network algebra, trace monoidal category) dacă pentru fiecare* $a, b, c \in B$ *se defineşte o operaţie* $\uparrow^a \_ : B(ab,ac) \longrightarrow B(b,c)$ *care satisface următoarele axiome :*

1. $g(\uparrow^a f) = \uparrow^a ((1_a + g)f)$ pentru $f : ab \longrightarrow ac$ şi $g : d \longrightarrow b$;
2. $(\uparrow^a f)g = \uparrow^a (f(1_a + g))$ pentru $f : ab \longrightarrow ac$ şi $g : c \longrightarrow d$;
3. $(\uparrow^a f) + 1_d = \uparrow^a (f + 1_d)$ pentru $f : ab \longrightarrow ac$;
4. $\uparrow^{ab} f = \uparrow^b \uparrow^a f$ pentru $f : abc \longrightarrow abd$;
5. $\uparrow^{ab} (({}^a\mathbf{X}^b + 1_c)f) = \uparrow^{ba} (f({}^a\mathbf{X}^b + 1_d))$ pentru $f : bac \longrightarrow abd$;

6. $\uparrow^a 1_a = 1_\lambda$;
7. $\uparrow^a {}^a\mathbf{X}^a = 1_a$.

Propoziţie 2.4 *Dacă repetiţia * este definită şi satisface* **R1** *şi* **R2** *atunci matricile formează un biflux.*

Demonstraţie Verificăm cele 7 axiome din definiţia bifluxlui pentru feedback-ul la stânga $\uparrow^a$ definit pentru matrici cu elemente peste semiinelul R.

1. Prima axiomă, cea a parametrilor la intrare, se demonstrează astfel :
$$E\left(\uparrow^a \begin{pmatrix} A & B \\ C & D \end{pmatrix}\right) = E(CA^*B \cup D) = ECA^*B \cup ED = \uparrow^a \begin{pmatrix} A & B \\ EC & ED \end{pmatrix} =$$
$$\uparrow^a \left(\begin{pmatrix} 1_a & 0 \\ 0 & E \end{pmatrix}\begin{pmatrix} A & B \\ C & D \end{pmatrix}\right) = \uparrow^a \left((1_a + E)\begin{pmatrix} A & B \\ C & D \end{pmatrix}\right)$$
2. A doua axiomă, cea a parametrilor la iesire, se demonstrează astfel :
$$\left(\uparrow^a \begin{pmatrix} A & B \\ C & D \end{pmatrix}\right)E = (CA^*B \cup D)E = CA^*BE \cup DE = \uparrow^a \begin{pmatrix} A & BE \\ C & DE \end{pmatrix} =$$
$$\uparrow^a \left(\begin{pmatrix} A & B \\ C & D \end{pmatrix}\begin{pmatrix} 1_a & 0 \\ 0 & E \end{pmatrix}\right) = \uparrow^a \left(\begin{pmatrix} A & B \\ C & D \end{pmatrix}(1_a + E)\right)$$
3. A treia axiomă sub forma mai generală a parametrilor la sumă se demonstrează astfel :
$$\uparrow^a \left(\begin{pmatrix} A & B \\ C & D \end{pmatrix} + E\right) = \uparrow^a \begin{pmatrix} A & B & 0 \\ C & D & 0 \\ 0 & 0 & E \end{pmatrix} = \begin{pmatrix} C \\ 0 \end{pmatrix} A^* \begin{pmatrix} B & 0 \end{pmatrix} \cup \begin{pmatrix} D & 0 \\ 0 & E \end{pmatrix} =$$
$$= \begin{pmatrix} CA^*B & 0 \\ 0 & 0 \end{pmatrix} \cup \begin{pmatrix} D & 0 \\ 0 & E \end{pmatrix} = \begin{pmatrix} CA^*B \cup D & 0 \\ 0 & E \end{pmatrix} = \uparrow^a \begin{pmatrix} A & B \\ C & D \end{pmatrix} + E$$
4. A patra axiomă:
Notând $W = (DA^*B \cup E)^*$ observăm că
$$\uparrow^b\uparrow^a \begin{pmatrix} A & B & C \\ D & E & F \\ G & H & J \end{pmatrix} = \uparrow^b \left[\begin{pmatrix} D \\ G \end{pmatrix} A^* \begin{pmatrix} B & C \end{pmatrix} \cup \begin{pmatrix} E & F \\ H & J \end{pmatrix}\right] = \uparrow^b \begin{pmatrix} DA^*B \cup E & DA^*C \cup F \\ GA^*B \cup H & GA^*C \cup J \end{pmatrix} =$$
$$= (GA^*B \cup H)W(DA^*C \cup F) \cup GA^*C \cup J$$
$$= GA^*BWDA^*C \cup GA^*BWF \cup HWDA^*C \cup HWF \cup GA^*C \cup J$$ şi că
$$\uparrow^{(a+b)} \begin{pmatrix} A & B & C \\ D & E & F \\ G & H & J \end{pmatrix} = \begin{pmatrix} G & H \end{pmatrix} \begin{pmatrix} A & B \\ D & E \end{pmatrix}^* \begin{pmatrix} C \\ F \end{pmatrix} \cup J$$
(următoarea egalitate se bazează pe teorema 2.1)
$$= \begin{pmatrix} G & H \end{pmatrix} \begin{pmatrix} A^* \cup A^*BWDA^* & A^*BW \\ WDA^* & W \end{pmatrix} \begin{pmatrix} C \\ F \end{pmatrix} \cup J$$
$$= \begin{pmatrix} G(A^* \cup A^*BWDA^*) \cup HWDA^* & GA^*BW \cup HW \end{pmatrix} \begin{pmatrix} C \\ F \end{pmatrix} \cup J$$
$$= (GA^* \cup GA^*BWDA^* \cup HWDA^*)C \cup (GA^*BW \cup HW)F \cup J$$
$$= GA^*C \cup GA^*BWDA^*C \cup HWDA^*C \cup GA^*BWF \cup HWF \cup J$$
Deci $\uparrow^b\uparrow^a \begin{pmatrix} A & B & C \\ D & E & F \\ G & H & J \end{pmatrix} = \uparrow^{(a+b)} \begin{pmatrix} A & B & C \\ D & E & F \\ G & H & J \end{pmatrix}$

5. A cincea axiomă :

$$\uparrow^{(a+b)}\left(({}^aX^b+1_c)\begin{pmatrix}A&B&C\\D&E&F\\G&H&J\end{pmatrix}\right)=\uparrow^{(a+b)}\begin{pmatrix}D&E&F\\A&B&C\\G&H&J\end{pmatrix}=\begin{pmatrix}G&H\end{pmatrix}\begin{pmatrix}D&E\\A&B\end{pmatrix}^*\begin{pmatrix}F\\C\end{pmatrix}\cup J$$

$$\uparrow^{(b+a)}\left(\begin{pmatrix}A&B&C\\D&E&F\\G&H&J\end{pmatrix}({}^aX^b+1_d)\right)=\uparrow^{(b+a)}\begin{pmatrix}B&A&C\\E&D&F\\H&G&J\end{pmatrix}=\begin{pmatrix}H&G\end{pmatrix}\begin{pmatrix}B&A\\E&D\end{pmatrix}^*\begin{pmatrix}C\\F\end{pmatrix}\cup J$$

$$=\begin{pmatrix}H&G\end{pmatrix}{}^aX^b\,{}^bX^a\begin{pmatrix}B&A\\E&D\end{pmatrix}^*{}^bX^a\begin{pmatrix}F\\C\end{pmatrix}\cup J$$

(egalitatea următoare rezultă folosind propoziţia 2.2)

$$=\begin{pmatrix}G&H\end{pmatrix}\left({}^bX^a\begin{pmatrix}B&A\\E&D\end{pmatrix}{}^bX^a\right)^*\begin{pmatrix}F\\C\end{pmatrix}\cup J=\begin{pmatrix}G&H\end{pmatrix}\begin{pmatrix}D&E\\A&B\end{pmatrix}^*\begin{pmatrix}F\\C\end{pmatrix}\cup J$$

6. A şasea axiomă rezultă din faptul că 1_0 este unica matrice cu 0 linii şi 0 coloane.

7. A şaptea axiomă :

$$\uparrow^a{}^aX^a=\uparrow^a\begin{pmatrix}0&1_a\\1_a&0\end{pmatrix}=0\cup 1_a0^*1_a\overset{\mathbf{R6}}{=}1_a$$

3 Matrici cu elemente într-un semiinel de relaţii

3.1 Semiinelul relaţiilor

Pentru orice mulţime S, mulţimea $\mathcal{P}(S\times S)$ a relaţiilor pe S devine semiinel cu reuniunea şi compunerea. În continuare vom utiliza acest semiinel $(\mathcal{P}(S\times S),\bigcup,\emptyset,\cdot,1_S)$. Reamintim că :

$$p\cdot q=\{(s,t)\in S\times S|\text{ există } u\in S \text{ cu proprietatea că } (s,u)\in p \text{ şi } (u,t)\in q\}$$

unde p şi q sunt submulţimi ale lui $S\times S$.

Observaţia 3.1 *Produsul(compunerea) de relaţii este distributiv faţă de reuniuni arbitrare. Fie R şi R_i, cu $i\in I$. Atunci:*

$$R\cdot\bigcup_{i\in I}R_i=\bigcup_{i\in I}R\cdot R_i$$

3.2 Semantica programelor nedeterministe

O matrice A cu a linii şi b coloane cu elemente relaţii dă semantica unui program nedeterminist P cu a intrări şi b ieşiri. Notăm cu S mulţimea stărilor memoriei calculatorului care efectuează calculele. Pentru $1\le i\le a$, $1\le j\le b$, $s\in S$ şi $t\in S$

$(s,t)\in A_{ij}$ dacă şi numai dacă executând programul P de la intrarea i a lui P şi din starea s a memoriei, execuţia se poate termina la ieşirea j a lui P în starea t a memoriei.

3.3 Matrici peste semiinelul $(\mathcal{P}(S\times S),\bigcup,\emptyset,\cdot,1_S)$

Propoziţie 3.2 *Produsul de matrici este distributiv faţă de reuniuni arbitrare.*

Demonstraţie : Fie A o matrice cu a linii şi b coloane şi A^i, cu $i\in I$, matrici cu b linii şi c coloane, cu elemente în semiinelul $\mathcal{P}(S\times S)$. Demonstrăm că

$$A\cdot(\bigcup_{i\in I}A^i)=\bigcup_{i\in I}A\cdot A^i.$$

Fie $l\in[a]$ şi $k\in[c]$. $(A\cdot\bigcup_{i\in I}A^i)_{l,k}=\bigcup_{p\in[b]}(A_{l,p}\cdot(\bigcup_{i\in I}A^i)_{p,k})=\bigcup_{p\in[b]}(A_{l,p}\cdot(\bigcup_{i\in I}A^i_{p,k}))=\bigcup_{p\in[b]}(\bigcup_{i\in I}(A_{l,p}\cdot A^i_{p,k}))=\bigcup_{i\in I}(\bigcup_{p\in[b]}(A_{l,p}\cdot A^i_{p,k}))=\bigcup_{i\in I}(A\cdot A^i)_{l,k}=(\bigcup_{i\in I}A\cdot A^i)_{l,k}$.

Cum l şi k au fost alese arbitrar va rezulta $A\cdot\bigcup_{i\in I}A^i=\bigcup_{i\in I}(A\cdot A^i)$

3.4 Binomul lui Newton într-un semiinel necomutativ

Fie $(R, +, 0, \cdot, 1)$ un semiinel necomutativ cu operaţiile notate aditiv şi multiplicativ. Atunci pentru orice a şi b aparţinând lui R şi pentru orice număr natural $n \geq 1$ are loc egalitatea:

$$(a+b)^n = \sum\{a^{i_0}ba^{i_1}\ldots ba^{i_k} \mid 0 \leq k \leq n, k+i_0+i_1+\ldots+i_k = n\}.$$

Demonstraţie: Prin inducţie după n probăm că $(a+b)^n = \sum\{a,b\}^n$. Cazul $n=1$ este evident.

Presupunem adevărată egalitatea pentru n şi demonstrăm pentru $n+1$.

$(a+b)^{n+1} = (a+b)^n(a+b) = (\sum\{a,b\}^n)(a+b) = (\sum\{a,b\}^n)a + (\sum\{a,b\}^n)b = \sum(\{a,b\}^n a) + \sum(\{a,b\}^n b) = \sum(\{a,b\}^n a \cup \{a,b\}^n b) = \sum\{a,b\}^{n+1}$.

Folosind comutativitatea sumei şi asociativitatea putem grupa după numărul k de factori b ai fiecărui produs. Prin urmare :

$(a+b)^n = \sum\{a,b\}^n = \sum_{0\leq k\leq n}\{a^{i_0}ba^{i_1}\ldots ba^{i_k} \mid i_0+i_1+\ldots+i_k = n-k\}$

$= \sum\{a^{i_0}ba^{i_1}\ldots ba^{i_k} \mid 0 \leq k \leq n,\ k+i_0+i_1+\ldots+i_k = n\}$.□

3.5 Repetiţia

Definiţia 3.3 Pentru orice matrice A pătratică cu elemente din $\mathcal{P}(S\times S)$ prin definiţie

$$A^\star = \bigcup_{n\geq 0} A^n.$$

Proprietăţi Repetiţia definită mai sus are proprităţile:

R1 $A^\star \cdot (B\cdot A^\star)^\star = (A\cup B)^\star$ unde A şi B sunt matrici pătratice cu acelaşi număr n de linii şi coloane

R2 $(A\cdot B)^\star = 1_n \cup A\cdot(B\cdot A)^\star\cdot B$ unde $A : n \longrightarrow m$ şi $B : m \longrightarrow n$

Demonstraţie:

$(A\cdot B)^\star = (A\cdot B)^0 \cup \bigcup_{i\geq 1}(A\cdot B)^i = 1_n \cup \bigcup_{j\geq 0} A\cdot(B\cdot A)^j\cdot B = 1_n \cup A\cdot(B\cdot A)^\star\cdot B$.

$A^\star\cdot(B\cdot A^\star)^\star = \bigcup_{k\geq 0} A^\star\cdot(B\cdot A^\star)^k = \bigcup_{k\geq 0}(\bigcup_{i_0\geq 0} A^{i_0})\cdot B\cdot(\bigcup_{i_1\geq 0} A^{i_1})\ldots B\cdot(\bigcup_{i_k\geq 0} A^{i_k}) = \bigcup\{A^{i_0}\cdot B\cdot A^{i_1}\cdot B\ldots\cdot B\cdot A^{i_k} | k\geq 0, i_0\geq 0, \ldots i_k\geq 0\} = \bigcup_{n\geq 0}\{A^{i_0}\cdot B\cdot A^{i_1}\cdot B\ldots\cdot B\cdot A^{i_k} | i_0+\ldots+i_k+k = n\} = \bigcup_{n\geq 0}(A\bigcup B)^n = (A\bigcup B)^\star$. □

Deoarece repetiţia matricilor cu elemente relaţii satisface toate axiomele impuse repetiţiei în secţiunea precedentă, deducem teorema următoare :

Teorema 3.4 *Matricile cu elemente relaţii formează un biflux .*

4 Bifluxuri de relaţii

Pe lângă matricile formate din relaţii cu elemente din S, o altă categorie Rel(S) a fost folosită ca model semantic pentru cazul nedeterminist. In cele ce urmează vom proba că cele două modele semantice sunt izomorfe. Pentru orice număr natural a notăm $[a] = \{1, 2, .., a\}$.

Fie S o mulţime. Fie a, b două numere naturale. Prin definiţie

$$Rel(S)(a,b) = \{r \mid r \subseteq (S\times[a])\times(S\times[b])\}$$

este mulţimea părţilor lui $(S\times[a])\times(S\times[b])$.

Revenind la semantica programelor nedeterministe, observăm că $r \in Rel(S)(a,b)$ dă semantica unui program P cu a intrări şi b ieşiri dacă şi numai dacă pentru orice $1\leq i\leq a$, $1\leq j\leq b$, $s\in S$ şi $t\in S$,

$((s,i),(t,j)) \in r$ dacă şi numai dacă executând programul P de la intrarea i a lui P şi din starea s a memoriei, execuţia se poate termina la ieşirea j a lui P în starea t a memoriei.

In $Rel(S)$ sunt definite următoarele operaţii:

1. Pentru $r \in Rel(S)(m,n)$ şi $r' \in Rel(S)(n,p)$ compunerea $r \cdot r' \in Rel(S)(m,p)$ se defineşte astfel :

$$r \cdot r' = \{\ ((s,i),(s',j))|(\exists(s_0,i_0) \in [n] \times S)[((s,i),(s_0.i_0)) \in r \text{ şi } ((s_0,i_0),(s',j)) \in r']\ \}.$$

2. Pentru $r \in Rel(S)(m,n)$ şi $r' \in Rel(S)(p,q)$ suma $r + r' \in Rel(S)(m+p,n+q)$ se defineşte astfel :

$$r + r' = r \cup \{\ ((s,m+i),(s',n+j))|((s,i),(s',j)) \in r'\}.$$

3. Pentru $r \in Rel(S)(m+1,n+1)$ definim $\uparrow r \in Rel(S)(m,n)$ pentru orice $s,t \in S$, $1 \leq i \leq m$ şi $1 \leq j \leq n$ prin

$$((s,i),(t,j)) \in \uparrow r \text{ dacă şi numai dacă}$$
$$((s,i+1),(t,j+1)) \in r \text{ sau}$$
$$(\exists k \geq 0)(\exists s_0,...,s_k \in S)\ ((s,i+1),(s_0,1)) \in r,\ (\forall j < k)((s_j,1),(s_{j+1},1)) \in r \text{ şi } ((s_k,1),(t,j+1)) \in r.$$

Pentru orice pereche de numere naturale a,b există o bijecţie între $Rel(S)(a,b)$ şi matricile cu a linii şi b coloane peste semiinelul relaţiilor definită pentru

$$r \subseteq (S \times [a]) \times (S \times [b]) \text{ şi matricea A} \in [\mathcal{P}(S \times S)](a,b)$$

prin

$$((s,i),(t,j)) \in r \text{ dacă şi numai dacă } (s,t) \in A_{ij}.$$

Teorema 4.1 *$Rel(S)$ şi $[\mathcal{P}(S \times S)]$ sunt izomorfe.*

Demonstraţie: Arătăm că există un morfism bijectiv de la $Rel(S)$ la $[\mathcal{P}(S \times S)]$. Bijecţia este cea definită mai sus. Mai trebuie să arătăm că bijecţia este şi un morfism. Definim funcţia f de la $Rel(S)$ la $[\mathcal{P}(S \times S)]$ astfel :

$$f(r)_{ij} = \{(s,t)|((s,i),(t,j)) \in r\}$$

cu $r \subseteq (S \times [a]) \times (S \times [b]), 1 \leq i \leq a, 1 \leq j \leq b$ şi $s,t \in S$. Arătăm că f este morfism.

Probăm că bijecţia f este morfism pentru operaţia de compunere. Pentru $r \in Rel(S)(m,n)$ şi $r' \in Rel(S)(n,q)$ arătăm că $f(r \cdot r') = f(r) \cdot f(r')$, adică $f(r \cdot r')_{ij} = \bigcup_{k=1}^{n} f(r)_{i,k} \cdot f(r')_{k,j}$ pentru orice $1 \leq i \leq m$ şi $1 \leq j \leq q$.

Pentru orice $s,t \in S$ observăm că

$$\begin{aligned}(s,t) \in f(r \cdot r')_{ij} &\Leftrightarrow ((s,i),(t,j)) \in r \cdot r' \\ &\Leftrightarrow (\exists(s_0,k))[((s,i),(s_0,k)) \in r \text{ şi } ((s_0,k),(t,j)) \in r'] \\ &\Leftrightarrow (\exists k)(\exists s_0)[(s,s_0) \in f(r)_{ik} \text{ şi } (s_0,t) \in f(r')_{kj}] \\ &\Leftrightarrow (\exists k)(s,t) \in f(r)_{ik} f(r')_{kj} \\ &\Leftrightarrow (s,t) \in \textstyle\bigcup_{k=1}^{n} f(r)_{ik} f(r')_{kj}\end{aligned}$$

Deci $f(rr') = f(r)f(r')$.

Probăm că f este morfism pentru operaţia de adunare a relaţiilor adică $f(r + r') = f(r) + f(r')$ pentru orice $r \in Rel(S)(m,n)$ şi $r' \in Rel(S)(p,q)$.

Considerăm următoarele cazuri :

• Pentru $i \leq m$ şi $j \leq n$ obţinem $f(r+r')_{ij} = \{(s,t)|((s,i),(t,j)) \in r+r'\}$. Dar din definiţia sumei $r+r'$ observăm că $((s,i),(t,j)) \in r+r'$ dacă şi numai dacă $((s,i),(t,j)) \in r$, prin urmare $f(r+r')_{ij} = \{(s,t)|((s,i),(t,j)) \in r\} = f(r)_{ij}$.

• Pentru $i \leq p$ şi $j \leq q$ obţinem $f(r+r')_{m+i,n+j} = \{(s,t)|((s,m+i),(t,n+j)) \in r+r'\}$. De data aceasta observăm, tot din definiţia sumei $r+r'$, că $f(r+r')_{m+i,n+j} = f(r')_{ij} = (f(r)+f(r'))_{m+i,n+j}$.

• Pentru $i \leq p$ şi $j \leq n$ obţinem $f(r+r')_{m+i,j} = \{(s,t)|((s,m+i),(t,j)) \in r+r'\}$. Din definiţia sumei $r+r'$, rezultă că $f(r+r')_{m+i,j} = \emptyset = (f(r)+f(r'))_{m+i,j}$.

• Pentru $i \leq m$ şi $j \leq q$ obţinem $f(r+r')_{m,n+j} = \{(s,t)|((s,m),(t,n+j)) \in r+r'\}$. Din definiţia sumei $r+r'$, deducem că $f(r+r')_{m,n+j} = \emptyset = (f(r)+f(r'))_{m,n+j}$.

Prin urmare , din cele patru cazuri de mai sus, deducem că $f(r+r')_{ij} = (f(r)+f(r'))_{ij}$ pentru orice $1 \leq i \leq m+p$ şi orice $1 \leq j \leq n+q$, deci $f(r+r') = f(r)+f(r')$.

Notăm cu r^* închiderea reflexivă şi tranzitivă a relaţiei $r \subseteq S \times S$ adică $r^* = 1_S \cup r \cup r^2 \cup ...$ unde $1_S = \{(s,s)|s \in S\}$.

Probăm că bijecţia f este morfism pentru operaţia feedback, adică $f(\uparrow r) = \uparrow f(r)$ pentru orice $r \in Rel(S)(m+1,n+1)$. Pentru orice $s,t \in S$, $1 \leq i \leq m$ şi $1 \leq j \leq n$ observăm că

$$(s,t) \in (f(\uparrow r))_{i,j} \Leftrightarrow ((s,i),(t,j)) \in (\uparrow r) \Leftrightarrow ((s,i+1),(t,j+1)) \in r \text{ sau}$$

$$(\exists k \geq 0)(\exists u_0,...,u_k \in S)[((s,i+1),(u_0,1)) \in r, (\forall j < k)((u_j,1),(u_{j+1},1)) \in r \text{ şi } ((u_k,1),(t,j+1)) \in r] \Leftrightarrow$$

$$\Leftrightarrow (s,t) \in f(r)_{i+1,j+1} \text{ sau}$$

$$(\exists k \geq 0)(\exists u_0,...,u_k \in S) \text{ cu } (s,u_0) \in f(r)_{i+1,1}, (\forall j < k)(u_j,u_j+1) \in f(r)_{11} \text{ şi } (u_k,t) \in f(r)_{1,j+1}$$

$$\Leftrightarrow (s,t) \in f(r)_{i+1,j+1} \text{ sau } (\exists u,v \in S)(s,u) \in f(r)_{i+1,1}, (u,v) \in (f(r)_{11})^* \text{ şi } (v,t) \in f(r)_{1,j+1}$$

$$\Leftrightarrow (s,t) \in f(r)_{i+1,j+1} \cup f(r)_{i+1,1}(f(r)_{11})^* f(r)_{1,j+1}.$$

Prin urmare pentru orice $1 \leq i \leq m$ şi $1 \leq j \leq n$

$$f(\uparrow r)_{ij} = f(r)_{i+1,j+1} \cup f(r)_{i+1,1}(f(r)_{11})^* f(r)_{1,j+1}$$

Scriind $f(r) = \begin{pmatrix} f(r)_{11} & B \\ C & D \end{pmatrix}$ rezultă că $f(\uparrow r)_{ij} = D_{ij} \cup C_{i1}(f(r)_{11})^* B_{1j} = D_{ij} \cup (C(f(r)_{11})^* B)_{ij}$, deci

$f(\uparrow r) = D \cup C(f(r)_{11})^* B = \uparrow f(r)$.

Corolar 1. Rel(S) e biflux.

4.1 Bifluxul relaţiilor finite

Oricare ar fi numerele naturale a şi b prin definiţie

$$Rel(a,b) = \{r|r \subseteq [a] \times [b]\}.$$

Dacă S este o mulţime cu un singur element $S = \{*\}$, se constată existenţa unei bijecţii intre $Rel(a,b)$ şi $Rel(\{*\})(a,b)$ definită prin :

$$b(r) = \{((*,i)(*,j))|(i,j) \in r\} \text{ deoarece } * \text{ nu are decât un rol decorativ.}$$

Debarasându-ne de rolul decorativ al steluţei deducem că relaţiile finite formeaza un biflux Rel. Descriem in continuare operaţiile acestui biflux obţinute prin particularizare din definiţiile lui $Rel(*)$.

• Cu operaţia de compunere şi identităţile $1_m = \{(i,i)|1 \leq i \leq m\}$ Rel este o categorie.

• Cu suma definită pentru $r \in \mathsf{Rel}(a,b)$ şi $r' \in \mathsf{Rel}(c,d)$ prin

$$r + r' = r \cup \{(a+i, b+j) | (i,j) \in r'\}$$

Rel devine o categorie strict monoidală.

• Cu transpoziţia bloc ${}^aX^b \in \mathsf{Rel}(a+b, b+a)$ definită prin

$${}^aX^b = \{(i, b+i) | 1 \leq i \leq a\} \cup \{(a+j, j) | 1 \leq j \leq b\}$$

Rel devine o categorie strict monoidală simetrică.

• Cu feedback-ul definit pentru $r \in \mathsf{Rel}(m+1, n+1)$ prin

$$\uparrow r = \{(i,j) \in [m] \times [n] | (i+1, j+1) \in r \text{ sau } [(i+1, 1) \in r \text{ şi } (1, j+1) \in r]\}$$

Rel este biflux.

5 Determinism

Programele deterministe au semantica într-o subcategorie a lui $Rel(S)$, denumită $PFn(S)$ şi formată din toate funcţiile parţiale.

Teorema 5.1 *PFn(S) este subbiflux al lui Rel(S).*

Demonstraţie In paşii de mai jos ai demonstraţiei, vom arăta că aplicând operaţiile bifluxului unor funcţii partiale rezultatul este tot o funcţie parţială.

1. Dacă $r \in PFn(S)(m,n)$ şi $r' \in PFn(S)(n,q)$, atunci $r; r' \in PFn(S)(m,q)$ deoarece compunerea de funcţii parţiale este funcţie parţială.

2. $1_n(s,i) = (s,i)$ pentru orice $i \in [n]$ şi $s \in S$ este funcţie.
Prin urmare, $PFn(S)$ este subcategorie cu aceleaşi obiecte ca ale lui $Rel(S)$.

3. Pentru $r \in PFn(S)(m,n)$ şi $r' \in PFn(S)(p,q)$ probăm că $r + r' \in PFn(S)(m+p, n+q)$
Fie $((s,i),(s',j)) \in r + r'$ şi $((s,i),(s'',k)) \in r + r'$

Vom trata următoarele 2 cazuri :

• Pentru $i \in [m]$ deducem $((s,i),(s',j)) \in r$ şi $((s,i),(s'',k)) \in r$. Deoarece $r \in PFn(S)$ rezultă că $(s',j) = (s'',k)$.
• Pentru $i > m$ deducem $j > n, ((s, i-m),(s', j-n)) \in r', k > n$ şi $((s, i-m),(s'', k-n)) \in r'$. Deoarece $r' \in PFn(S)$ rezultă că $(s', j-n) = (s'', k-n)$. Prin urmare $j = k$ şi $s' = s''$ deci $(s',j) = (s'',k)$.
Prin urmare $PFn(S)$ este subcategorie strict monoidală a lui $Rel(S)$.

4. ${}^n\mathbf{X}^p(s,i) = \begin{cases} (s, p+i) & \text{dacă } 1 \leq i \leq n \\ (s, i-n) & \text{dacă } n < i. \end{cases}$ este funcţie.
Prin urmare $PFn(S)$ este subcategorie strict monoidală simetrică a lui $Rel(S)$.

5. Pentru $r \in PFn(S)(m+1, n+1)$ aratăm că $\uparrow r \in PFn(S)(m,n)$. Fie $((s,i),(s',j)) \in \uparrow r$ şi $((s,i),(s'',k)) \in \uparrow r$. Tinând cont de definiţia lui $\uparrow r \in Rel(S)$ vom trata următoarele posibilităţi :

• Pentru $((s, i+1),(s', j+1)) \in r$ şi $((s, i+1),(s'', k+1)) \in r$ deoarece r este în $PFn(S)$ deducem $(s', j+1) = (s'', k+1)$ deci $(s', j) = (s'', k)$.

• Pentru $((s, i+1)(s', j+1)) \in r$ şi $((s, i+1),(s'', k+1)) \notin r$ observăm că există $s_0 \in S$ cu proprietatea $((s, i+1),(s_0, 1)) \in r$, prin urmare deoarece r este în $PFn(S)$ deducem $(s', j+1) = (s_0, 1)$ ceea ce este o contradicţie deoarece $j + 1 > 1$.

• Cazul $((s, i+1),(s', j+1)) \notin r$ şi $((s, i+1),(s'', k+1)) \in r$ conduce ca mai sus la o contradicţie.

• Ultimul caz $((s, i+1),(s', j+1)) \notin r$ şi $((s, i+1),(s'', k+1)) \notin r$ implică
$(\exists u \geq 0)(\exists s_0, s_1, ...s_u \in S)$ cu $((s, i+1),(s_0, 1)) \in r$, $(\forall p < u)((s_p, 1),(s_{p+1}, 1)) \in r$ şi $((s_u, 1),(s', j+1)) \in r$

precum şi
$(\exists v \geq 0)(\exists s'_0, s'_1, ...s'_v \in S)$ cu $((s, i+1), (s'_0, 1)) \in r$, $(\forall q < v)((s'_q, 1), (s'_{q+1}, 1)) \in r$ şi
$((s'_v, 1), (s'', k+1)) \in r$.

Din $((s, i+1), (s_0, 1)) \in r$ şi $((s, i+1), (s'_0, 1)) \in r$, deoarece r este în $PFn(S)$ deducem $s_0 = s'_0$. Prin inducţie probăm că $s_p = s'_p$ atâta timp cât $p \leq u$ şi $p \leq v$.
Din $((s_p, 1)(s_p + 1, 1)) \in r$ şi $((s'_p, 1)(s'_{p+1}, 1)) \in r$ deoarece $s_p = s'_p$ conform ipotezei de inducţie şi r este în $PFn(S)$ deducem $s_{p+1} = s'_{p+1}$.

Probăm că $u = v$. Presupunem prin absurd, de exemplu că $u < v$. Din cele de mai sus deducem $s_u = s'_u$. Din

$$((s_u, 1), (s', j+1)) \in r \text{ şi } ((s_u, 1), (s'_{u+1}, 1)) \in r$$

deoarece r este în $PFn(S)$ deducem $j + 1 = 1$, o contradicţie.
Deoarece $u = v$, $s_u = s'_v$ şi r este în $PFn(S)$ din

$$((s_u, 1), (s', j+1)) \in r \text{ şi } ((s'_v, 1), (s'', k+1)) \in r$$

rezultă că $(s', j+1) = (s'', k+1)$, deci $(s', j) = (s'', k)$.
Deci $PFn(s)$ este subiflux al lui $Rel(S)$.

5.1 Bifluxul functiilor parţiale

Funcţiile parţiale finite formează o categorie PFn având numere naturale drept obiecte şi morfismele definite prin :

$$\mathsf{PFn}(m, n) = \{f \mid f : [m] \rightharpoonup [n] \text{ funcţie parţială } \} \text{ pentru } m, n \in \mathsf{N}.$$

Dacă S este o mulţime cu un singur element $S = \{*\}$, se constată existenţa unei bijecţii între $\mathsf{PFn}(m, n)$ şi $\mathsf{PFn}(S)(m, n)$ definită pentru orice $f \in \mathsf{PFn}(\mathsf{m}, \mathsf{n})$ prin :

$$h(f)((*, i)) = (*, f(i)) \text{ pentru orice } 1 \leq i \leq m \text{ deoarece } * \text{ nu are decât rol decorativ.}$$

Debarasându-ne de rolul decorativ al steluţei deducem că funcţiile parţiale formează un biflux. Descriem în continuare operaţiile acestui biflux, operaţii obţinute prin particularizare din definiţiile lui $\mathsf{PFn}(*)$.

• Cu operaţia de compunere a funcţiilor parţiale şi funcţiile $1_m : [m] \to [m]$ cu $1_m(i) = i$ pentru orice $1 \leq i \leq m$, PFn este o categorie.

• Cu suma $f + g \in \mathsf{PFn}(\mathsf{a} + \mathsf{c}, \mathsf{b} + \mathsf{d})$ definită pentru $f \in \mathsf{PFn}(a, b)$ şi $g \in \mathsf{PFn}(c, d)$ prin :

$$(f + g)(i) = \begin{cases} f(i) & \text{dacă } i < a \\ b + g(i - a) & \text{dacă } i > a \end{cases}$$

pentru orice $i \in [a + c]$, PFn devine o categorie strict monoidală.

• Cu transpoziţia bloc ${}^aX^b \in \mathsf{PFn}(a + b, b + a)$ definită prin

$${}^aX^b = \{(i, b + i) | 1 \leq i \leq a\} \cup \{(a + j, j) | 1 \leq j \leq b\}$$

PFn devine o categorie strict monoidală simetrică.

• Cu feedback-ul definit pentru $r \in \mathsf{PFn}(m + 1, n + 1)$ prin

$$(\uparrow r)(i) = \begin{cases} r(i+1) - 1 & \text{dacă } r(i+1) \neq 1 \\ r(1) - 1 & \text{dacă } r(i+1) = 1 \end{cases}$$

PFn este biflux.

6 Relaţii şi funcţii S-sortate

Relaţiile finite S-sortate le vom nota cu $\mathsf{Rel_S}$. Ele formează un biflux. Monoidul obiectelor este S^*. Un cuvant $a \in S^*$ se scrie $a = a_1a_2...a_{|a|}$ unde $|a|$ este lungimea sa şi a_i sunt literele sale pentru $i \in [|a|]$. Pentru $a, b \in S^*$ prin definiţie :

$$\mathsf{Rel_S(a,b)} = \{\mathsf{f} \subseteq [|\mathsf{a}|] \times [|\mathsf{b}|] \ |(\mathsf{i},\mathsf{j}) \in \mathsf{f} \Rightarrow \mathsf{a_i} = \mathsf{b_j}\}$$

Operaţiile in $\mathsf{Rel_S}$ sunt descrise în cele ce urmează.

Compunerea este definită pentru $f \in \mathsf{Rel_S(a,b)}$ şi $g \in \mathsf{Rel_S(b,c)}$ prin :

$$fg = \{(i,k)|(\exists j) \text{ cu } (i,j) \in f \text{ şi } (j,k) \in g\}.$$

Verificăm că $fg \in \mathsf{Rel_S}$. Dacă $(i,k) \in fg$ atunci $(\exists j)(i,j) \in f$ şi $(j,k) \in g$. Din $(i,j) \in f$ rezultă $a_i = b_j$ iar din $(j,k) \in g$ deducem $b_j = c_k$ deci $a_i = c_k$. Rezultă că $fg \in \mathsf{Rel_S}$.

Observăm că $1_a = \{(i,i)|i \in [|a|]\}$ este in $\mathsf{Rel_S}$.

Suma este definită pentru $f \in \mathsf{Rel_S(a,b)}$ şi $g \in \mathsf{Rel_S(c,d)}$ prin:

$$f + g = f \cup \{(|a| + i, |b| + j)|(i,j) \in g\}.$$

Arătăm că $f + g \in \mathsf{Rel_S(ac,bd)}$ pentru $f \in \mathsf{Rel_S(a,b)}$ şi $g \in \mathsf{Rel_S(c,d)}$. Fie $(m,n) \in f + g$. Se arată că $f + g \in \mathsf{Rel_S(ac,bd)}$ demonstrând că $(ac)_m = (bd)_n$.
Există două cazuri : dacă $(m,n) \in f \in \mathsf{Rel_S(a,b)}$ ceea ce implică $m \leq |a|$, atunci $(ac)_m = a_m = b_n = (bd)_n$.
Al doilea caz presupune $(m,n) \in \{(|a| + i, |b| + j|(i,j) \in g)\}$, adică $m = |a| + i$, $n = |b| + j$ şi $(i,j) \in g$.
Prin urmare , $(ac)_m = c_i$ iar $(bd)_n = d_j$. Dar $(i,j) \in g$ implică $c_i = d_j$ ceea ce asigură egalitatea $(ac)_{|a|+i} = (bd)_{|b|+j}$, deci $(ac)_m = (bd)_n$.

Transpoziţia bloc este definită prin :

$${}^a\mathsf{X}^b = \{(i, |b| + i)|i \in [|a|]\} \cup \{(|a| + i, i)|i \in [|b|]\}.$$

Pentru $s \in S$ şi $f \in \mathsf{Rel_S(sa,sb)}$ prin definiţie:

$$\uparrow^s f = \{(i,j) \in [|a|] \times [|b|]|(i+1, j+1) \in f \text{ sau } [(i+1,1) \in f \text{ şi } (1, j+1) \in f]\}$$

In primul caz $(i+1, j+1) \in f$ implică $a_i = b_j$ iar in al doilea caz observăm că
$a_i = (sa)_{1+i} = (sb)_1 = s = (sa)_1 = (sb)_{j+1} = b_j$.

Există un morfism U care acţionează ca un functor atât pe obiecte cât şi pe săgeţi :

$$U : \mathsf{Rel_S} \longrightarrow \mathsf{Rel}.$$

Pe obiecte morfismul de monoizi $U : S^* \longrightarrow N$ este definit prin $U(a) = |a|$ pentru orice $a \in S^*$.
Deoarece $\mathsf{Rel_S(a,b)} \subseteq \mathsf{Rel}(|\mathsf{a}|, |\mathsf{b}|)$, dacă $f \in \mathsf{Rel_S(a,b)}$ atunci $U(f) = f \in Rel(|a|, |b|)$, U acţionând de la $Rel_S(a,b)$ la $Rel(|a|, |b|)$ ca o funcţie incluziune.
Ca morfism, U îndeplineşte condiţiile :

$$U(fg) = U(f)U(g), \qquad U(1_a) = 1_{|a|}, \qquad U(f + g) = U(f) + U(g), ... \text{ etc}$$

Verificarea axiomei $(f + g)(u + v) = fu + gv$ in $\mathsf{Rel_S}$ se face astfel :

$(f + g)(u + v) = U((f + g)(u + v)) = (U(f) + U(g))(U(u) + U(v))$ şi $fu = gv = U(f)U(u) + U(g)U(v)$ şi $fu + gv = U(fu + gv) = U(f)U(u) + U(g)U(v)$. Dar deoarece Rel este categorie strict monoidală $(U(f) + U(g))(U(u) + U(v)) = U(f)U(u) + U(g)U(v)$ prin urmare se obţine $(f + g)(u + v) = fu + gv$.

In acest stil se pot verifica toate axiomele conceptului de biflux pentru $\mathsf{Rel_S}$.

Se observă că în cazul particular când S are exact un element, identificăm S^* cu monoidul aditiv al numerelor naturale şi, prin urmare, $\mathsf{Rel_S}$ poate fi identificat cu Rel (relaţiile finite).

5 Reprezentarea programelor abstracte

VEC

February 14, 2010

Fie B un M-flux şi X un monoid cu elementul neutru ε. Fie $i, o\colon X \longrightarrow M$ două morfisme de monoizi. Cu mici excepţii acesta este cadrul folosit în această lecţie.

În cazurile concrete X este monoidul liber al instrucţiunilor şi M este monoidul aditiv al numerelor naturale.

1 Sintaxa

Definition 1 Fie $a, b \in M$. Se numeşte reprezentant de program cu a intrări şi b ieşiri o pereche ordonată $\langle x,\ f\rangle$ unde $x \in X$ şi $f \in B(ao(x), bi(x))$.

Mulţimea tuturor reprezentanţilor de programe cu a intrări şi b ieşiri se notează cu $Fl_{X,B}(a, b)$. □

Principalele operaţii cu reprezentanţii de programe sunt compunerea, suma şi feedbackul.

Definition 2 Compunerea lui $\langle x,\ f\rangle$ din $Fl_{X,B}(a, b)$ cu $\langle y,\ g\rangle$ din $Fl_{X,B}(b, c)$ este în $Fl_{X,B}(a, c)$ şi este definită prin

$$\langle x,\ f\rangle;\langle y,\ g\rangle = \langle xy,\ (f + 1_{o(y)})(1_b + {}^{i(x)}\mathrm{X}^{o(y)})(g + 1_{i(x)})(1_c + {}^{i(y)}\mathrm{X}^{i(x)})\rangle.\square$$

Definition 3 Suma lui $\langle x,\ f\rangle$ din $Fl_{X,B}(a, b)$ cu $\langle y,\ g\rangle$ din $Fl_{X,B}(c, d)$ este în $Fl_{X,B}(ac, bd)$ şi este definită prin

$$\langle x,\ f\rangle + \langle y,\ g\rangle = \langle xy,\ (1_a + {}^{c}\mathrm{X}^{o(x)} + 1_{o(y)})(f + g)(1_b + {}^{i(x)}\mathrm{X}^{d} + 1_{i(y)})\rangle.\square$$

Definition 4 Feedbackul lui $\langle x,\ f\rangle$ din $Fl_{X,B}(ab, ac)$ este în $Fl_{X,B}(b, c)$ şi este definit prin

$$\uparrow^a \langle x,\ f\rangle = \langle x,\ \uparrow^a f\rangle.\square$$

Pentru $f \in B(a, b)$ definim $E_B(f) \in Fl_{X,B}(a, b)$ prin $E_B(f) = \langle \varepsilon,\ f\rangle$.

Aceste definiţii sunt sursa de inspiraţie a următoarelor fapte.

1.1 M-fluxul instrucţiunilor

M-fluxul instrucţiunilor $Q(X)$ este definit prin

- $Q(X)(a, b) = \{\langle a, x, b\rangle\colon x \in X\}$ pentru $a, b \in M$,
- Compunerea $\langle a, x, b\rangle;\langle b, y, c\rangle = \langle a, xy, c\rangle$,
- Identitate $1_a = \langle a, \varepsilon, a\rangle$
- suma $\langle a, x, b\rangle + \langle c, y, d\rangle = \langle ac, xy, bd\rangle$,
- permutarea ${}^{a}\mathrm{X}^{b} = \langle ab, \varepsilon, ba\rangle$,
- feedbackul la stânga $\uparrow^a \langle ab, x, ac\rangle = \langle b, x, c\rangle$.

Observăm că dacă X este comutativ, atunci $Q(X)$ este biflux.

1.2 Fluxul conexiunilor

1.2.1 Cazul aciclic

În acest paragraf presupunem numai că B este o M-csmns. Definim în cele ce urmează o altă M-csmns $K(B)$.

Pentru $a, b \in M$ definim

$$K(B)(a,b) = \{\langle f, i, o\rangle : \ i, o \in M, \ f \in B(ao, bi)\}.$$

Pentru $\langle f, i, o\rangle$ în $K(B)(a,b)$ şi $\langle f', i', o'\rangle$ în $K(B)(b,c)$ definim **compunerea** lor prin

$$\langle f, i, o\rangle ; \langle f', i', o'\rangle = \langle (f + 1_{o'})(1_b + {}^{i}\mathrm{X}^{o'})(f' + 1_i)(1_c + {}^{i'}\mathrm{X}^{i}), ii', oo'\rangle.$$

Probăm asociativitatea compunerii. Presupunem în plus că $\langle f", i", o"\rangle$ este în $K(B)(c,d)$.

$(\langle f, i, o\rangle ; \langle f', i', o'\rangle) ; \langle f", i", o"\rangle =$

$\langle [(f + 1_{o'})(1_b + {}^{i}\mathrm{X}^{o'})(f' + 1_i)(1_c + {}^{i'}\mathrm{X}^{i}) + 1_{o"}](1_c + {}^{ii'}\mathrm{X}^{o"})(f" + 1_{ii'})(1_d + {}^{i"}\mathrm{X}^{ii'}), (ii')i", (oo')o"\rangle =$

$\langle (f + 1_{o'o"})(1_b + {}^{i}\mathrm{X}^{o'} + 1_{o"})(f' + 1_{io"})(1_c + {}^{i'}\mathrm{X}^{i} + 1_{o"})(1_c + {}^{ii'}\mathrm{X}^{o"})(f" + {}^{i}\mathrm{X}^{i'} ; {}^{i'}\mathrm{X}^{i})(1_d + {}^{i"}\mathrm{X}^{ii'}), i(i'i"), o(o'o")\rangle =$

$\langle (f + 1_{o'o"})(1_b + {}^{i}\mathrm{X}^{o'} + 1_{o"})(f' + {}^{i}\mathrm{X}^{o"} ; {}^{o"}\mathrm{X}^{i})(1_c + {}^{i'i}\mathrm{X}^{o"})(1_{co"} + {}^{i'}\mathrm{X}^{i})$
$(f" + {}^{i}\mathrm{X}^{i'})(1_{di"} + {}^{i'}\mathrm{X}^{i})(1_d + {}^{i"}\mathrm{X}^{i} + 1_{i'})(1_{di} + {}^{i"}\mathrm{X}^{i'}), i(i'i"), o(o'o")\rangle =$

$\langle (f + 1_{o'o"})(1_b + {}^{i}\mathrm{X}^{o'} + 1_{o"})(1_{bo'} + {}^{i}\mathrm{X}^{o"})(f' + 1_{o"i})(1_{ci'} + {}^{o"}\mathrm{X}^{i})(1_c + {}^{i'i}\mathrm{X}^{o"})$
$(f" + 1_{i'i})(1_d + {}^{i"i'}\mathrm{X}^{i})(1_{di} + {}^{i"}\mathrm{X}^{i'}), i(i'i"), o(o'o")\rangle =$

$\langle (f + 1_{o'o"})(1_b + {}^{i}\mathrm{X}^{o'o"})(f' + 1_{o"i})(1_c + {}^{i'}\mathrm{X}^{o"} + 1_i)(f" + 1_{i'i})(1_d + {}^{i"}\mathrm{X}^{i'} + 1_i)(1_d + {}^{i'i"}\mathrm{X}^{i}), i(i'i"), o(o'o")\rangle =$

$\langle (f + 1_{o'o"})(1_b + {}^{i}\mathrm{X}^{o'o"})[(f' + 1_{o"})(1_c + {}^{i'}\mathrm{X}^{o"})(f" + 1_{i'})(1_d + {}^{i"}\mathrm{X}^{i'}) + 1_i](1_d + {}^{i'i"}\mathrm{X}^{i}), i(i'i"), o(o'o")\rangle =$

$\langle f, i, o\rangle ; \langle (f' + 1_{o"})(1_c + {}^{i'}\mathrm{X}^{o"})(f" + 1_{i'})(1_d + {}^{i"}\mathrm{X}^{i'}), i'i", o'o"\rangle = \langle f, i, o\rangle ; (\langle f', i', o'\rangle ; \langle f", i", o"\rangle).$

Pentru orice $f \in B(a,b)$ definim $I_B(f) \in K(B)(a,b)$ prin

$$I_B(f) = \langle f, \lambda, \lambda\rangle.$$

Menţionăm următoarele reguli de calcul

$$\langle f, i, o\rangle ; I_B(f') = \langle f(f' + 1_i), i, o\rangle \qquad \text{şi} \qquad I_B(f) ; \langle f', i', o'\rangle = \langle (f + 1_{o'})f', i', o'\rangle.$$

Deducem că $1_a = I_B(1_a)$ este morfism identitate, deci $K(B)$ este o categorie. Mai observăm că $I_B : \ B \longrightarrow K(B)$ este functor.

Pentru $\langle f, i, o\rangle$ în $K(B)(a,b)$ şi $\langle f', i', o'\rangle$ în $K(B)(c,d)$ definim **suma** lor prin

$$\langle f, i, o\rangle + \langle f', i', o'\rangle = \langle (1_a + {}^{c}\mathrm{X}^{o} + 1_{o'})(f + f')(1_b + {}^{i}\mathrm{X}^{d} + 1_{i'}), ii', oo'\rangle.$$

Probăm asociativitatea sumei. Presupunem în plus că $\langle f", i", o"\rangle$ este în $K(B)(u,v)$.

$(\langle f, i, o\rangle + \langle f', i', o'\rangle) + \langle f", i", o"\rangle =$

$\langle (1_{ac} + {}^{u}\mathrm{X}^{oo'} + 1_{o"})[(1_a + {}^{c}\mathrm{X}^{o} + 1_{o'})(f + f')(1_b + {}^{i}\mathrm{X}^{d} + 1_{i'}) + f"](1_{bd} + {}^{ii'}\mathrm{X}^{v} + 1_{i"}), (ii')i", (oo')o"\rangle =$

$\langle (1_{ac} + {}^{u}\mathrm{X}^{o} + 1_{o'o"})(1_{aco} + {}^{u}\mathrm{X}^{o'} + 1_{o"})(1_a + {}^{c}\mathrm{X}^{o} + 1_{o'uo"})[(f + f') + f"]$
$(1_b + {}^{i}\mathrm{X}^{d} + 1_{i'vi"})(1_{bdi} + {}^{i'}\mathrm{X}^{v} + 1_{i"})(1_{bd} + {}^{i}\mathrm{X}^{v} + 1_{i'i"}), i(i'i"), o(o'o")\rangle =$

$\langle (1_{ac} + {}^{u}\mathrm{X}^{o} + 1_{o'o"})(1_a + {}^{c}\mathrm{X}^{o} + 1_{uo'o"})(1_{aoc} + {}^{u}\mathrm{X}^{o'} + 1_{o"})[f + (f' + f")]$
$(1_{bid} + {}^{i'}\mathrm{X}^{v} + 1_{i"})(1_b + {}^{i}\mathrm{X}^{d} + 1_{vi'i"})(1_{bd} + {}^{i}\mathrm{X}^{v} + 1_{i'i"}), i(i'i"), o(o'o")\rangle =$

$\langle (1_a + {}^{cu}\mathrm{X}^{o} + 1_{o'o"})[f + (1_c + {}^{u}\mathrm{X}^{o'} + 1_{o"})(f' + f")(1_d + {}^{i'}\mathrm{X}^{v} + 1_{i"})](1_b + {}^{i}\mathrm{X}^{dv} + 1_{i'i"}), i(i'i"), o(o'o")\rangle =$

$\langle f, i, o\rangle + \langle (1_c + {}^{u}\mathrm{X}^{o'} + 1_{o"})(f' + f")(1_d + {}^{i'}\mathrm{X}^{v} + 1_{i"}), i'i", o'o"\rangle = \langle f, i, o\rangle + (\langle f', i', o'\rangle + \langle f", i", o"\rangle)$

Menţionăm cazul particular

$$I_B(f) + \langle f', i', o' \rangle = \langle f + f', i', o' \rangle.$$

Demonstrăm că $K(B)$ este o csmn. Este uşor de arătat că 1_λ este element neutru pentru suma morfismelor şi că $1_a + 1_b = 1_{a+b}$.

Pentru $\langle f, i, o \rangle$ în $K(B)(a, b)$ şi $\langle f', i', o' \rangle$ în $K(B)(c, d)$ observăm că

$$(\langle f, i, o \rangle + 1_c); (1_b + \langle f', i', o' \rangle) = \langle (1_a + {}^c\mathrm{X}^o)(f + 1_c)(1_b + {}^i\mathrm{X}^c), i, o \rangle; \langle 1_b + f', i', o' \rangle =$$

$$\langle ((1_a + {}^c\mathrm{X}^o)(f + 1_c)(1_b + {}^i\mathrm{X}^c) + 1_{o'})(1_{bc} + {}^i\mathrm{X}^{o'})(1_b + f' + 1_i)(1_{bd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^c\mathrm{X}^o + 1_{o'})(f + 1_{co'})(1_b + {}^i\mathrm{X}^c + 1_{o'})(1_{bc} + {}^i\mathrm{X}^{o'})(1_b + f' + 1_i)(1_{bd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^c\mathrm{X}^o + 1_{o'})(f + 1_{co'})(1_b + {}^i\mathrm{X}^{co'})(1_b + f' + 1_i)(1_{bd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^c\mathrm{X}^o + 1_{o'})(f + 1_{co'})(1_{bi} + f')(1_b + {}^i\mathrm{X}^{di'})(1_{bd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^c\mathrm{X}^o + 1_{o'})(f + f')(1_b + {}^i\mathrm{X}^d + 1_{i'}), ii', oo' \rangle = \langle f, i, o \rangle + \langle f', i', o' \rangle$$

Pentru $\langle f, i, o \rangle$ în $K(B)(a, b)$ şi $\langle f', i', o' \rangle$ în $K(B)(b, c)$ observăm că

$$(\langle f, i, o \rangle + 1_d); (\langle f', i', o' \rangle + 1_d) = \langle (1_a + {}^d\mathrm{X}^o)(f + 1_d)(1_b + {}^i\mathrm{X}^d), i, o \rangle; \langle (1_b + {}^d\mathrm{X}^{o'})(f' + 1_d)(1_c + {}^{i'}\mathrm{X}^d), i', o' \rangle =$$

$$\langle ((1_a + {}^d\mathrm{X}^o)(f + 1_d)(1_b + {}^i\mathrm{X}^d) + 1_{o'})(1_{bd} + {}^i\mathrm{X}^{o'})((1_b + {}^d\mathrm{X}^{o'})(f' + 1_d)(1_c + {}^{i'}\mathrm{X}^d) + 1_i)(1_{cd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^d\mathrm{X}^o + 1_{o'})(f + 1_{do'})(1_b + {}^i\mathrm{X}^d + 1_{o'})(1_{bd} + {}^i\mathrm{X}^{o'})(1_b + {}^d\mathrm{X}^{o'} + 1_i)(f' + 1_{di})(1_c + {}^{i'}\mathrm{X}^d + 1_i)(1_{cd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^d\mathrm{X}^o + 1_{o'})(f + 1_{do'})(1_b + {}^i\mathrm{X}^{do'})(1_b + {}^d\mathrm{X}^{o'} + 1_i)(f' + {}^d\mathrm{X}^i; {}^i\mathrm{X}^d)(1_c + {}^{i'}\mathrm{X}^d + 1_i)(1_{cd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^d\mathrm{X}^o + 1_{o'})(f + 1_{do'})(1_{bi} + {}^d\mathrm{X}^{o'})(1_b + {}^i\mathrm{X}^{o'd})(1_{bo'} + {}^d\mathrm{X}^i)(f' + 1_{id})(1_c + {}^{i'i}\mathrm{X}^d)(1_{cd} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$\langle (1_a + {}^d\mathrm{X}^o + 1_{o'})(f + 1_{do'})(1_{bi} + {}^d\mathrm{X}^{o'})(1_b + {}^i\mathrm{X}^{o'} + 1_d)(f' + 1_{id})(1_c + {}^{i'}\mathrm{X}^i + 1_d)(1_c + {}^{ii'}\mathrm{X}^d), ii', oo' \rangle =$$

$$\langle (1_a + {}^d\mathrm{X}^o + 1_{o'})(1_{ao} + {}^d\mathrm{X}^{o'})(f + 1_{o'd})(1_b + {}^i\mathrm{X}^{o'} + 1_d)(f' + 1_{id})(1_c + {}^{i'}\mathrm{X}^i + 1_d)(1_{ci} + {}^{i'}\mathrm{X}^d)(1_c + {}^i\mathrm{X}^d + 1_{i'}), ii', oo' \rangle =$$

$$\langle (1_a + {}^d\mathrm{X}^{oo'})((f + 1_{o'})(1_b + {}^i\mathrm{X}^{o'})(f' + 1_i)(1_c + {}^{i'}\mathrm{X}^i) + 1_d)(1_c + {}^{ii'}\mathrm{X}^d), ii', oo' \rangle =$$

$$\langle (f + 1_{o'})(1_b + {}^i\mathrm{X}^{o'})(f' + 1_i)(1_c + {}^{i'}\mathrm{X}^i), ii', oo' \rangle + 1_d = (\langle f, i, o \rangle; \langle f', i', o' \rangle) + 1_d$$

Mai observăm că

$$(1_d + \langle f, i, o \rangle); (1_d + \langle f', i', o' \rangle) = \langle 1_d + f, i, o \rangle; \langle 1_d + f', i', o' \rangle =$$

$$\langle (1_d + f + 1_{o'})(1_{db} + {}^i\mathrm{X}^{o'})(1_d + f' + 1_i)(1_{dc} + {}^{i'}\mathrm{X}^i), ii', oo' \rangle = \langle 1_d + (f + 1_{o'})(1_b + {}^i\mathrm{X}^{o'})(f' + 1_i)(1_c + {}^{i'}\mathrm{X}^i), ii', oo' \rangle =$$

$$1_d + \langle (f + 1_{o'})(1_b + {}^i\mathrm{X}^{o'})(f' + 1_i)(1_c + {}^{i'}\mathrm{X}^i), ii', oo' \rangle = 1_d + \langle f, i, o \rangle; \langle f', i', o' \rangle$$

În concluzie $K(B)$ este o csmn.

Este uşor de arătat că $I_B \colon B \longrightarrow K(B)$ este un morfism de M-csmn.

Pentru $a, b \in M$ definim

$${}^a\mathrm{X}^b = I_B({}^a\mathrm{X}^b)$$

şi probăm că $K(B)$ este o csmns.

Pentru $\langle f, i, o \rangle$ din $K(B)(a, b)$ observăm că

$${}^c\mathrm{X}^a(\langle f, i, o \rangle + 1_c)\,{}^b\mathrm{X}^c = \langle ({}^c\mathrm{X}^a + 1_o)(1_a + {}^c\mathrm{X}^o)(f + 1_c)(1_b + {}^i\mathrm{X}^c)({}^b\mathrm{X}^c + 1_i), i, o \rangle = \langle {}^c\mathrm{X}^{ao}(f + 1_c)\,{}^{bi}\mathrm{X}^c, i, o \rangle =$$

$$\langle 1_c + f, i, o \rangle = 1_c + \langle f, i, o \rangle.$$

Pentru $\langle f, i, o \rangle$ din $K(B)(a, b)$ şi $c, d \in M$ observăm că

$$(1_{cd} + \langle f, i, o \rangle); ({}^c\mathrm{X}^d + 1_b) = \langle (1_{cd} + f)({}^c\mathrm{X}^d + 1_{bi}), i, o \rangle = \langle {}^c\mathrm{X}^d + f, i, o \rangle = {}^c\mathrm{X}^d + \langle f, i, o \rangle.$$

Restul axiomelor fiind uşor de demonstrat, concluzionăm că $K(B)$ este o csmns. Mai observăm că $I_B\colon B \longrightarrow K(B)$ este un morfism de M-csmns.

1.2.2 Cazul ciclic

În ipoteza iniţială că B este un M-flux vom arăta că şi $K(B)$ devine un M-flux.

Pentru $\langle f,i,o\rangle$ din $K(B)(ab,ac)$ feedbackul este definit prin

$$\uparrow^a \langle f,i,o\rangle = \langle \uparrow^a f,i,o\rangle.$$

Pentru $\langle f,i,o\rangle$ din $K(B)(ab,ac)$ şi $\langle f',i',o'\rangle$ din $K(B)(d,b)$ observăm că

$\langle f',i',o'\rangle;\uparrow^a \langle f,i,o\rangle = \langle f',i',o'\rangle;\langle \uparrow^a f,i,o\rangle = \langle (f'+1_o)(1_b + {}^{i'}\mathrm{X}^o)(\uparrow^a f + 1_{i'})(1_c + {}^{i}\mathrm{X}^{i'}), i'i, o'o\rangle =$

$\langle \uparrow^a ((1_a + f' + 1_o)(1_{ab} + {}^{i'}\mathrm{X}^o)(f + 1_{i'})(1_{ac} + {}^{i}\mathrm{X}^{i'})), i'i, o'o\rangle = \uparrow^a \langle (1_a + f' + 1_o)(1_{ab} + {}^{i'}\mathrm{X}^o)(f + 1_{i'})(1_{ac} + {}^{i}\mathrm{X}^{i'}), i'i, o'o\rangle =$

$\uparrow^a (\langle 1_a + f', i', o'\rangle; \langle f,i,o\rangle) = \uparrow^a ((1_a + \langle f',i',o'\rangle); \langle f,i,o\rangle)$

Pentru $\langle f,i,o\rangle$ din $K(B)(ab,ac)$ şi $\langle f',i',o'\rangle$ din $K(B)(c,d)$ observăm că

$(\uparrow^a \langle f,i,o\rangle); \langle f',i',o'\rangle = \langle \uparrow^a f,i,o\rangle; \langle f',i',o'\rangle = \langle (\uparrow^a f + 1_{o'})(1_c + {}^{i}\mathrm{X}^{o'})(f' + 1_i)(1_d + {}^{i'}\mathrm{X}^i), ii', oo'\rangle =$

$\langle \uparrow^a ((f + 1_{o'})(1_{ac} + {}^{i}\mathrm{X}^{o'})(1_a + f' + 1_i)(1_{ad} + {}^{i'}\mathrm{X}^i)), ii', oo'\rangle = \uparrow^a \langle (f + 1_{o'})(1_{ac} + {}^{i}\mathrm{X}^{o'})(1_a + f' + 1_i)(1_{ad} + {}^{i'}\mathrm{X}^i), ii', oo'\rangle =$

$\uparrow^a (\langle f,i,o\rangle; \langle 1_a + f', i', o'\rangle) = \uparrow^a (\langle f,i,o\rangle; (1_a + \langle f',i',o'\rangle))$

Pentru $\langle f,i,o\rangle$ din $K(B)(ab,ac)$ şi $d \in M$ observăm că

$(\uparrow^a \langle f,i,o\rangle) + 1_d = \langle \uparrow^a f,i,o\rangle + 1_d = \langle (1_b + {}^{d}\mathrm{X}^o)(\uparrow^a f + 1_d)(1_c + {}^{i}\mathrm{X}^d), i, o\rangle = \uparrow^a \langle (1_{ab} + {}^{d}\mathrm{X}^o)(f + 1_d)(1_{ac} + {}^{i}\mathrm{X}^d), i, o\rangle =$

$\uparrow^a (\langle f,i,o\rangle + 1_d)$

Pentru $\langle f,i,o\rangle$ din $K(B)(abc,abd)$ observăm că

$\uparrow^b\uparrow^a \langle f,i,o\rangle = \langle \uparrow^b\uparrow^a f,i,o\rangle = \langle \uparrow^{ab} f,i,o\rangle = \uparrow^{ab} \langle f,i,o\rangle$

Pentru $\langle f,i,o\rangle$ din $K(B)(bac,abd)$ observăm că

$\uparrow^{ab} (({}^{a}\mathrm{X}^b + 1_c)\langle f,i,o\rangle) = \langle \uparrow^{ab} [({}^{a}\mathrm{X}^b + 1_{co})f], i, o\rangle = \langle \uparrow^{ba} [f({}^{a}\mathrm{X}^b + 1_{di})], i, o\rangle = \uparrow^{ba} (\langle f,i,o\rangle; ({}^{a}\mathrm{X}^b + 1_d))$.

Deoarece egalitaţile $\uparrow^a 1_a = 1_\lambda$ şi $\uparrow^a {}^{a}\mathrm{X}^a = 1_a$ sunt uşor de probat concluzionăm că $K(B)$ este un M-flux. Mai observăm că $I_B\colon B \longrightarrow K(B)$ este un morfism de M-fluxuri.

1.3 Fluxul reprezentărilor de programe abstracte

Produsul cartezian $Q(X) \times K(B)$ cu operaţiile pe componente este un M-flux. Pentru $a,b \in M$ definim

$$P(a,b) = \{\langle\langle a,x,b\rangle, \langle f, i(x), o(x)\rangle\rangle : x \in X, \langle f,i(x),o(x)\rangle \in K(B)(a,b)\}.$$

Deoarece P este o submulţime a lui $Q(X) \times K(B)$ care este închisă la compunere, sumă, feedback şi include constantele 1_a şi ${}^{a}\mathrm{X}^b$ pentru orice $a,b \in M$, deducem că P este un M-flux.

Funcţia care duce $\langle\langle a,x,b\rangle, \langle f,i(x),o(x)\rangle\rangle \in P(a,b)$ în $\langle x,\ f\rangle$ din $Fl_{X,B}(a,b)$ este un izomorfism cu privire la compunere, sumă, feedback şi constantele 1_a şi ${}^{a}\mathrm{X}^b$. Prin urmare $Fl_{X,B}$ este un M-flux.

Reamintim că pentru orice $f \in B(a,b)$ prin definiţia $E_B(f) = \langle \varepsilon,\ f\rangle$ din $Fl_{X,B}(a,b)$. menţionăm următoarele reguli de calcul

- $\langle x,\ f\rangle; E_B(g) = \langle x,\ f(g + 1_{i(x)})\rangle$ pentru $\langle x,\ f\rangle \in Fl_{X,B}(a,b)$ şi $g \in B(b,c)$,
- $E_B(f); \langle x,\ g\rangle = \langle x,\ (f + 1_{o(x)})g\rangle$ pentru $f \in B(a,b)$ şi $\langle x,\ f\rangle \in Fl_{X,B}(b,c)$,
- $E_B(f) + \langle x,\ g\rangle = \langle x,\ f + g\rangle$ pentru $f \in B(a,b)$ şi $\langle x,\ g\rangle \in Fl_{X,B}(c,d)$.

Deducem că $E_B : B \longrightarrow Fl_{X,B}$ este un morfism de M-fluxuri.

Menţionăm şi regula de calcul a feedbackului la dreapta. Dacă $\langle x,\ f\rangle \in Fl_{X,B}(ba, ca)$, atunci

$$\langle x,\ f\rangle \uparrow^a = \langle x,\ [(1_b + {}^{o(x)}\mathrm{X}^a) f (1_c + {}^{a}\mathrm{X}^{i(x)})] \uparrow^a\rangle = \langle x,\ \uparrow^a [({}^{a}\mathrm{X}^b + 1_{o(x)}) f ({}^{c}\mathrm{X}^a + 1_{i(x)})]\rangle.$$

2 Semantică

Remarcăm că morfismele unei csmn B formează împreună cu operaţia sumă un monoid pe care-l notăm tot cu B.

Se ştie că interpretarea dă semantica instrucţiunilor din care se construesc programele. Interpretarea ataşează fiecărei instrucţiuni $\sigma \in \Sigma$ o funcţie parţială

$$\mathcal{I}(\sigma) \in Pfn(S)(i(\sigma), o(\sigma))$$

unde $i(\sigma)$ şi $o(\sigma)$ sunt numărul intrărilor, respectiv ieşirilor lui σ. Extinzând funcţiile i, o şi $\mathcal{I}$ la morfisme de monoizi $i : \Sigma^* \longrightarrow (\omega, +, 0)$, $o : \Sigma^* \longrightarrow (\omega, +, 0)$ şi $\mathcal{I} : \Sigma^* \longrightarrow Pfn(S)$ se observă că $\mathcal{I}$ devine un morfism de monoizi de la monoidul liber al instrucţiunilor la monoidul morfismelor lui $Pfn(S)$. Mai remarcăm că

$$(\forall x \in \Sigma^*)\mathcal{I}(x) \in Pfn(S)(i(x), o(x)).$$

Aceste fapte constituie motivaţia pentru definiţia următoare.

Definition 5 O interpretare a monoidului X în csmn B este un morfism de monoizi

$$I : X \longrightarrow (B, +, 1_\lambda)$$

cu valori în monoidul morfismelor lui B.

Notând cu $s : (B, +, 1_\lambda) \longrightarrow M$, respectiv cu $c : (B, +, 1_\lambda) \longrightarrow M$ morfismele de monoizi care ataşează fiecărui morfism din B sursa, respectiv cosursa sa, vom spune că I este o interpretare cu privire la Is şi Ic. □

Fie $u, v : X \to Ob(B)$ sunt două morfisme de monoizi. Dacă I este o interpretare a monoidului X în csmn B cu privire la u şi v atunci $(\forall x \in X)I(x) \in B(u(x), v(x))$.

După această introducere revenim la cadrul general al acestei lecţii.
Fie B un M-flux şi $i, o : X \longrightarrow M$ două morfisme de monoizi.

Proposition 6 Pentru orice $x \in X$ definim $E_X(x) \in Fl_{X,B}(i(x), o(x))$ prin

$$E_X(x) = \langle x,\ {}^{i(x)}\mathrm{X}^{o(x)}\rangle.$$

E_X este o interpretare a lui X în $Fl_{X,B}$ cu privire la i şi o care este numită **interpretarea standard** a lui X în $Fl_{X,B}$. În plus $E_B(f)$ permută cu $E_X(x)$ pentru orice $f \in B(a, b)$ şi $x \in X$.

Proof: Probăm că E_X este morfism de monoizi.

Evident $E_X(1_\varepsilon) = \langle \varepsilon,\ {}^{i(\varepsilon)}\mathrm{X}^{o(\varepsilon)}\rangle = \langle \varepsilon,\ 1_\lambda\rangle = 1_\lambda$.

Pentru $x, y \in X$

$E_X(x) + E_X(y) = \langle x,\ {}^{i(x)}\mathrm{X}^{o(x)}\rangle + \langle y,\ {}^{i(y)}\mathrm{X}^{o(y)}\rangle =$

$\langle xy,\ (1_{i(x)} + {}^{i(y)}\mathrm{X}^{o(x)} + 1_{o(y)})({}^{i(x)}\mathrm{X}^{o(x)} + {}^{i(y)}\mathrm{X}^{o(y)})(1_{o(x)} + {}^{i(x)}\mathrm{X}^{o(y)} + 1_{i(y)})\rangle = \langle xy,\ {}^{i(xy)}\mathrm{X}^{o(xy)}\rangle = E_X(xy)$.

Mai observăm că

$(1_a + E_X(x))(E_B(f) + 1_{o(x)}) = \langle x,\ 1_a + {}^{i(x)}\mathrm{X}^{o(x)}\rangle E_B(f + 1_{o(x)}) =$

$\langle x,\ (1_a + {}^{i(x)}\mathrm{X}^{o(x)})(f + 1_{o(x)} + 1_{i(x)})\rangle = \langle x,\ f + {}^{i(x)}\mathrm{X}^{o(x)}\rangle = E_B(f) + \langle x,\ {}^{i(x)}\mathrm{X}^{o(x)}\rangle = E_B(f) + E_X(x)$. □

Proposition 7 Pentru orice $\langle x,\ f\rangle$ din $Fl_{X,B}(a, b)$

$$\langle x,\ f\rangle = ((1_a + E_X(x))E_B(f)) \uparrow^{i(x)}.$$

Proof: $((1_a + E_X(x))E_B(f)) \uparrow^{i(x)} = (\langle x,\ 1_a + {}^{i(x)}\mathrm{X}^{o(x)}\rangle E_B(f)) \uparrow^{i(x)} = \langle x,\ (1_a + {}^{i(x)}\mathrm{X}^{o(x)})(f + 1_{i(x)})\rangle \uparrow^{i(x)} =$

$\langle x,\ [(f + 1_{i(x)})(1_b + {}^{i(x)}\mathrm{X}^{i(x)})] \uparrow^{i(x)}\rangle = \langle x,\ f\rangle$.□

Lemma 8 Fie B un M-flux. Pentru morfismele

$$x : i \to o \text{ şi } y : i' \to o',$$

$$f : ao \to bi \text{ şi } g : bo' \to ci'$$

dacă $f\mathbf{P}y$, atunci

$$((1_a + x)f) \uparrow^i ; ((1_b + y)g) \uparrow^{i'} = ((1_a + x + y)(f + 1_{o'})(1_b + {}^i\mathrm{X}^{o'})(g + 1_i)(1_c + {}^{i'}\mathrm{X}^i)) \uparrow^{ii'}$$

Proof: $((1_a + x + y)(f + 1_{o'})(1_b + {}^i\mathrm{X}^{o'})(g + 1_i)(1_c + {}^{i'}\mathrm{X}^i)) \uparrow^{ii'} = (((1_a + x)f + y)(1_b + {}^i\mathrm{X}^{o'})(g + 1_i)(1_c + {}^{i'}\mathrm{X}^i)) \uparrow^{ii'} =$
$(((1_a + x)f + 1_{i'})(1_b + {}^i\mathrm{X}^{i'})((1_b + y)g + 1_i)(1_c + {}^{i'}\mathrm{X}^i)) \uparrow^{i'} \uparrow^i =$
$((1_a + x)f[(1_b + {}^i\mathrm{X}^{i'})((1_b + y)g + 1_i)(1_c + {}^{i'}\mathrm{X}^i)] \uparrow^{i'}) \uparrow^i =$
$((1_a + x)f[((1_b + y)g) \uparrow^{i'} + 1_i]) \uparrow^i = ((1_a + x)f) \uparrow^i ; ((1_b + y)g) \uparrow^{i'}$. □

Lemma 9 Fie B un M-flux. Pentru morfismele

$$x : i \to o \text{ şi } y : i' \to o',$$

$$f : ao \to bi \text{ şi } g : co' \to di'$$

dacă $f\mathbf{P}y$, atunci

$$((1_a + x)f) \uparrow^i + ((1_c + y)g) \uparrow^{i'} = ((1_{ac} + x + y)(1_a + {}^c\mathrm{X}^o + 1_{o'})(f + g)(1_b + {}^i\mathrm{X}^d + 1_{i'})) \uparrow^{ii'}$$

Proof: $((1_{ac} + x + y)(1_a + {}^c\mathrm{X}^o + 1_{o'})(f + g)(1_b + {}^i\mathrm{X}^d + 1_{i'})) \uparrow^{ii'} =$
$((1_a + {}^c\mathrm{X}^i + 1_{i'})((1_a + x)f + (1_c + y)g)(1_b + {}^i\mathrm{X}^d + 1_{i'})) \uparrow^{i'} \uparrow^i =$
$((1_a + {}^c\mathrm{X}^i)((1_a + x)f + [(1_c + y)g] \uparrow^{i'})(1_b + {}^i\mathrm{X}^d)) \uparrow^i = ((1_a + x)f) \uparrow^i + ((1_b + y)g) \uparrow^{i'}$. □

Theorem 10 Pentru orice morfism de fluxuri $H : B \longrightarrow B'$ şi pentru orice interpretare I a lui X în B' cu privire la $i; H$ şi $o; H$ dacă $H(f)$ permută cu $I(x)$ pentru orice morfism f din B şi orice $x \in X$, atunci există un unic morfism de fluxuri

$$\langle I, H\rangle^f : Fl_{X,B} \longrightarrow B'$$

cu proprietăţile $E_X; \langle I, H\rangle^f = I$ şi $E_B; \langle I, H\rangle^f = H$.

Proof: Probăm unicitatea. Presupunem că morfismul $\langle I, H\rangle^f$ din enunţ există şi are proprietăţile cerute. Dacă $a \in M$, atunci $\langle I, H\rangle^f(a) = \langle I, H\rangle^f(E_B(a)) = H(a)$. Pentru orice $\langle x, f\rangle$ din $Fl_{X,B}(a, b)$, utilizând propoziţia precedentă deducem

$$\langle I, H\rangle^f(\langle x, f\rangle) = ((1_{H(a)} + I(x))H(f)) \uparrow^{H(i(x))}$$

ceea ce probează unicitatea morfismului cerut în enunţ.

Probăm existenţa. Definim $\langle I, H\rangle^f$ prin

- pentru orice $a \in M$, $\langle I, H\rangle^f(a) = H(a)$
- pentru orice $\langle x, f\rangle$ din $Fl_{X,B}(a, b)$

$$\langle I, H\rangle^f(\langle x, f\rangle) = ((1_{H(a)} + I(x))H(f)) \uparrow^{H(i(x))}$$

Dacă $\langle x, f\rangle$ este din $Fl_{X,B}(a, b)$ şi $\langle y, g\rangle$ este din $Fl_{X,B}(b, c)$, atunci utilizând lema 8 deducem

$\langle I, H\rangle^f(\langle x, f\rangle\langle y, g\rangle) = ((1_{H(a)} + I(xy))H((f + 1_{o(y)})(1_b + {}^{i(x)}\mathrm{X}^{o(y)})(g + 1_{i(x)})(1_c + {}^{i(y)}\mathrm{X}^{i(x)}))) \uparrow^{H(i(xy))} =$

$((1_{H(a)} + I(x) + I(y))(H(f) + 1_{H(o(y))})(1_{H(b)} + {}^{H(i(x))}\mathrm{X}^{H(o(y))})(H(g) + 1_{H(i(x))})(1_{H(c)} + {}^{H(i(y))}\mathrm{X}^{H(i(x))})) \uparrow^{H(i(x))H(i(y))} =$

$[(1_{H(a)} + I(x))H(f)] \uparrow^{H(i(x))}; [(1_{H(b)} + I(y))H(g)] \uparrow^{H(i(y))} = \langle I, H\rangle^f(\langle x, f\rangle); \langle I, H\rangle^f(\langle y, g\rangle)$.

Dacă $\langle x, f\rangle$ este din $Fl_{X,B}(a, b)$ şi $\langle y, g\rangle$ este din $Fl_{X,B}(c, d)$, atunci utilizând lema 9 deducem

$\langle I, H\rangle^f(\langle x, f\rangle + \langle y, g\rangle) = ((1_{H(ac)} + I(xy))H((1_a + {}^c\mathrm{X}^{o(x)} + 1_{o(y)})(f + g)(1_b + {}^{i(x)}\mathrm{X}^d + 1_{i(y)}))) \uparrow^{H(i(xy))} =$

$((1_{H(a)H(c)} + I(x) + I(y))(1_{H(a)} + {}^{H(c)}\mathrm{X}^{H(o(x))} + 1_{H(o(y))})(H(f) + H(g))(1_{H(b)} + {}^{H(i(x))}\mathrm{X}^{H(d)} + 1_{H(i(y))})) \uparrow^{H(i(x))H(i(y))} =$

$[(1_{H(a)} + I(x))H(f)] \uparrow^{H(i(x))} + [(1_{H(c)} + I(y))H(g)] \uparrow^{H(i(y))} = \langle I, H\rangle^f(\langle x, f\rangle) + \langle I, H\rangle^f(\langle y, g\rangle)$.

Dacă $\langle x, f\rangle$ este din $Fl_{X,B}(ab, ac)$, atunci

$\langle I, H\rangle^f(\uparrow^a \langle x, f\rangle) = \langle I, H\rangle^f(\langle x, \uparrow^a f\rangle) = [(1_{H(b)} + I(x))H(\uparrow^a f)] \uparrow^{H(i(x))} = \uparrow^{H(a)} [(1_{H(ab)} + I(x))H(f)] \uparrow^{H(i(x))} =$

$\uparrow^{H(a)} \langle I, H\rangle^f(\langle x, f\rangle)$.

Dacă $f \in B(a, b)$, atunci

$$\langle I, H\rangle^f(E_B(f)) = [(1_{H(a)} + I(\varepsilon))H(f)] \uparrow^{H(i(\varepsilon))} = H(f),$$

prin urmare $\langle I, H\rangle^f(1_a) = 1_a$ şi $\langle I, H\rangle^f({}^a\mathrm{X}^b) = {}^a\mathrm{X}^b$.

Dacă $x \in X$, atunci

$\langle I, H\rangle^f(E_X(x)) = [(1_{H(i(x))} + I(x))H({}^{i(x)}\mathrm{X}^{o(x)})] \uparrow^{H(i(x))} = ({}^{H(i(x))}\mathrm{X}^{H(i(x))}(I(x) + 1_{H(i(x))})) \uparrow^{H(i(x))} = I(x)$. □

7 PROGRAME ABSTRACTE

VEC

February 14, 2010

Deoarece un program poate avea mai multe reprezentări, vom considera ca echivalente diversele reprezentări ale aceluiaşi program. Mai mult, *două programe reprezentate prin grafuri etichetate se consideră egale dacă cele două grafuri sunt izomorfe.*

Pentru a formaliza aceste idei vom discuta despre programe nedeterministe (conexiunile sunt din Rel) cu instucţiuni din mulţimea Σ. Fie deci două morfisme de monoizi $i\colon \Sigma^* \longrightarrow (N, +, 0)$ şi $o\colon \Sigma^* \longrightarrow (N, +, 0)$ care indica numărul intrărilor, respectiv numarul ieşirilor. Reamintim că aceste morfisme se prelungesc în mod unic la morfisme de csms de la Bi_Σ la Rel, morfisme pe care în cele ce urmeaza le notăm tot cu i şi o. De exemplu pentru $j \in Bi_\Sigma(x, y)$ morfismul $i(j) \in Bi(i(x), i(y))$ poate fi definit prin

$$i(j)(i(x_1x_2 \dots x_{m-1}) + t) = i(y_1y_2 \dots y_{j(m)-1}) + t \tag{1}$$

pentru orice $m \in [|x|]$ şi $t \in [i(x_m)]$.

Programul reprezentat prin $\langle x,\ f\rangle$ din $Fl_{\Sigma,Rel}(a, b)$ poate fi vazut ca un graf etichetat astfel:

1. Sunt a noduri pentru intrările programului, b noduri pentru ieşirile programului şi $|x|$ noduri interne etichetate cu literele lui x.

2. Relaţia $f \in Rel(a + o(x), b + i(x))$ dă sageţile programului după cum urmează

   1. pentru $p \in [\,a\,]$ şi $q \in [\,b\,]$,
   $$\langle p,\ q\rangle \in f$$
   dacă şi numai dacă există o săgeată de la intrarea p a programului la ieşirea q a programului,

   2. pentru $p \in [\,a\,]$, $m \in [\,|\,x\,|\,]$ şi $t \in [\,i(x_m)\,]$,
   $$\langle p,\ b + i(x_1 + x_2 + \cdots + x_{m-1}) + t\rangle \in f$$
   dacă şi numai dacă există o săgeată de la intrarea p a programului la intrarea t a instrucţiunii x_m care etichetează vârful m,

   3. pentru $n \in [\,|\,x\,|\,]$, $s \in [\,o(x_n)\,]$ şi $q \in [\,b\,]$,
   $$\langle a + o(x_1 + x_2 + \cdots + x_{n-1}) + s,\ q\rangle \in f$$
   dacă şi numai dacă există o săgeată de la ieşirea s a instrucţiunii x_n care etichetează vârful n la ieşirea q a programului.

   4. pentru $n \in [\,|\,x\,|\,]$, $s \in [\,o(x_n)\,]$, $m \in [\,|\,x\,|\,]$ şi $t \in [\,i(x_m)\,]$,
   $$\langle a + o(x_1 + x_2 + \cdots + x_{n-1}) + s,\ b + i(x_1 + x_2 + \cdots + x_{m-1}) + t\rangle \in f$$
   dacă şi numai dacă există o săgeată de la ieşirea s a instrucţiunii x_n care etichetează vârful n la intrarea t a instrucţiunii x_m care etichetează vârful m.

Proposition 1 Programele reprezentate prin $\langle x,\ f\rangle$ şi $\langle y,\ g\rangle$ din $Fl_{\Sigma,Rel}(a, b)$ sunt izomorfe dacă şi numai dacă există $j \in Bi_\Sigma(x, y)$ astfel încât

$$f(1_b + i(j)) = (1_a + o(j))g. \tag{2}$$

Proof: Dacă cele două programe sunt izomorfe, atunci există o bijectie j între nodurile interne ale lor astfel încât nodurile care corespund prin j sunt etichetate cu aceeaşi instrucţiune, adică $j \in Bi_\Sigma(x, y)$. În continuare vom presupune existenţa bijecţiei $j \in Bi_\Sigma(x, y)$ şi vom arăta că această bijectie stabileşte un izomorfism între cele două scheme dacă şi numai dacă egalitatea (2) este adevărată.

Pentru început să explicăm care este semnificaţia bijecţiilor

$$i(j) \in Bi(i(x), i(y)) \text{ şi } o(j) \in Bi(o(x), o(y)).$$

În (1) $i(x_1 + x_2 + \cdots + x_{m-1}) + t$ reprezintă intrarea t a instrucţiunii x_m care etichetează nodul m din programul reprezentat prin $\langle x,\ f\rangle$ şi $i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t$ reprezintă intrarea t a instrucţiunii $y_{j(m)} = x_m$ care etichetează vârful corespunzător $j(m)$ din programul reprezentat de $\langle y,\ g\rangle$.
Prin urmare $i(j)$ este o bijecţie între toate intrările instrucţiunilor programului reprezentat prin $\langle x,\ f\rangle$ şi toate intrările instrucţiunilor programului reprezentat prin $\langle y,\ g\rangle$ astfel încât două intrări corespund prin $i(j)$ dacă şi numai dacă sunt intrările cu acelaşi număr a unei instrucţiuni care etichetează două vârfuri care corespund prin j.
Analog, $o(j)$ este o bijecţie între toate ieşirile instrucţiunilor programului reprezentat prin $\langle x,\ f\rangle$ şi toate ieşirile instrucţiunilor programului reprezentat prin $\langle y,\ g\rangle$ astfel încât două ieşiri corespund prin $o(j)$ dacă şi numai dacă sunt ieşirile cu acelaşi număr a unei instrucţiuni care etichetează două vârfuri care corespund prin j.

Egalitatea (2) este echivaventă cu următoarele patru egalităţi:

1. $(1_a + \top_{o(x)})f(1_b + i(j))(1_b + \bot^{i(y)}) = (1_a + \top_{o(x)})(1_a + o(j))g(1_b + \bot^{i(y)})$
2. $(1_a + \top_{o(x)})f(1_b + i(j))(\bot^b + 1_{i(y)}) = (1_a + \top_{o(x)})(1_a + o(j))g(\bot^b + 1_{i(y)})$
3. $(\top_a + 1_{o(x)})f(1_b + i(j))(1_b + \bot^{i(y)}) = (\top_a + 1_{o(x)})(1_a + o(j))g(1_b + \bot^{i(y)})$
4. $(\top_a + 1_{o(x)})f(1_b + i(j))(\bot^b + 1_{i(y)}) = (\top_a + 1_{o(x)})(1_a + o(j))g(\bot^b + 1_{i(y)})$.

Prima egalitate este adevărată dacă şi numai dacă pentru orice $p \in [\,a\,]$ şi $q \in [\,b\,]$ urmatoarele afirmaţii sunt echivalenţe:

- în programul reprezentat prin $\langle x,\ f\rangle$ există o săgeată de la intrarea p a programului la ieşirea q a programului
- în programul reprezentat prin $\langle y,\ g\rangle$ există o săgeată de la intrarea p a programului la ieşirea q a programului.

Într-adevăr, deoarece

$$\langle p,\ q\rangle \in f \Leftrightarrow \langle p,\ q\rangle \in f(1_b + i(j)) \Leftrightarrow \langle p,\ q\rangle \in (1_a + \top_{o(x)})f(1_b + i(j))(1_b + \bot^{i(y)})$$

şi

$$\langle p,\ q\rangle \in g \Leftrightarrow \langle p,\ q\rangle \in (1_a + o(j))g \Leftrightarrow \langle p,\ q\rangle \in (1_a + \top_{o(x)})(1_a + o(j))g(1_b + \bot^{i(y)})$$

rezultă că prima egalitate este echivalentă cu

$$(\forall p \in [\,a\,])(\forall q \in [\,b\,])[\langle p,\ q\rangle \in f \Longleftrightarrow \langle p,\ q\rangle \in g],$$

adică cu enunţul de mai sus.

A doua egalitate este adevărată dacă şi numai dacă pentru orice $p \in [\,a\,]$, $m \in [\,|\,x\,|\,]$ şi $t \in [\,i(x_m)\,]$ următoarele afirmaţii sunt echivalenţe:

- în programul reprezentat prin $\langle x,\ f\rangle$ există o săgeată de la intrarea p a programului la intrarea t a instrucţiunii x_m care etichetează nodul m
- în programul reprezentat prin $\langle y,\ g\rangle$ există o săgeată de la intrarea p a programului la intrarea t a instrucţiunii $y_{j(m)} = x_m$ care etichetează vârful corespunzător $j(m)$.

Într-adevăr, deoarece prima afirmaţie este echivalentă succesiv cu

$$\langle p,\ b + i(x_1 + x_2 + \cdots + x_{m-1}) + t\rangle \in f$$

$$\langle p,\ b + i(j)(i(x_1 + x_2 + \cdots + x_{m-1}) + t)\rangle \in f(1_b + i(j))$$

$$\langle p,\ i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t\rangle \in (1_a + \top_{o(x)})f(1_b + i(j))(\bot^b + 1_{i(y)})$$

şi a doua afirmaţie este echivalentă succesiv cu

$$\langle p,\ b + i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t\rangle \in g$$

$$\langle p,\ b + i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t\rangle \in (1_a + o(j))g$$

$$\langle p,\, i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t\rangle \in (1_a + \top_{o(x)})(1_a + o(j))g(\bot^b + 1_{i(y)})$$

rezultă enunţul de mai sus.

A treia egalitate este adevărată dacă şi numai dacă pentru orice $n \in [\,|x|\,]$, $s \in [\,o(x_n)\,]$ şi $q \in [\,b\,]$ următoarele afirmaţii sunt echivalenţe:

- în programul reprezentat prin $\langle x,\, f\rangle$ există o săgeată de la ieşirea s a instrucţiunii x_n care etichetează vârful n la ieşirea q a programului
- în programul reprezentat prin $\langle y,\, g\rangle$ există o săgeată de la ieşirea s a instrucţiunii $y_{j(n)} = x_n$ care etichetează vârful corespunzător $j(n)$ la ieşirea q a programului.

Într-adevăr, deoarece prima afirmaţie este echivalentă succesiv cu

$$\langle a + o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, q\rangle \in f$$

$$\langle a + o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, q\rangle \in f(1_b + i(j))$$

$$\langle o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, q\rangle \in (\top_a + 1_{o(x)})f(1_b + i(j))(1_b + \bot^{i(y)})$$

şi a doua afirmaţie este echivalentă succesiv cu

$$\langle a + o(y_1 + y_2 + \cdots + y_{j(n)-1}) + s,\, q\rangle \in g$$

$$\langle a + o(j)(o(x_1 + x_2 + \cdots + x_{n-1}) + s),\, q\rangle \in g$$

$$\langle a + o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, q\rangle \in (1_a + o(j))g$$

$$\langle o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, q\rangle \in (\top_a + 1_{o(x)})(1_a + o(j))g(1_b + \bot^{i(y)})$$

rezultă enunţul de mai sus.

A patra egalitate este adevărată dacă şi numai dacă pentru orice $n \in [\,|x|\,]$, $s \in [\,o(x_n)\,]$, $m \in [\,|x|\,]$ şi $t \in [\,i(x_m)\,]$ următoarele afirmaţii sunt echivalenţe:

- în programul reprezentat prin $\langle x,\, f\rangle$ există o săgeată de la ieşirea s a instrucţiunii x_n care etichetează vârful n la intrarea t a instrucţiunii x_m care etichetează nodul m
- în programul reprezentat prin $\langle y,\, g\rangle$ există o săgeată de la ieşirea s a instrucţiunii $y_{j(n)} = x_n$ care etichetează vârful corespunzător $j(n)$ la intrarea t a instrucţiunii $y_{j(m)} = x_m$ care etichetează vârful corespunzător $j(m)$.

Într-adevăr, deoarece prima afirmaţie este echivalentă succesiv cu

$$\langle a + o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, b + i(x_1 + x_2 + \cdots + x_{m-1}) + t\rangle \in f$$

$$\langle a + o(x_1 + \cdots + x_{n-1}) + s,\, b + i(j)(i(x_1 + \cdots + x_{m-1}) + t)\rangle \in f(1_b + i(j))$$

$$\langle o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t\rangle \in (\top_a + 1_{o(x)})f(1_b + i(j))(\bot^b + 1_{i(y)})$$

şi a doua afirmaţie este echivalentă succesiv cu

$$\langle a + o(y_1 + y_2 + \cdots + y_{j(n)-1}) + s,\, b + i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t\rangle \in g$$

$$\langle a + o(j)(o(x_1 + \cdots + x_{n-1}) + s),\, b + i(y_1 + \cdots + y_{j(m)-1}) + t\rangle \in g$$

$$\langle a + o(x_1 + \cdots + x_{n-1}) + s,\, b + i(y_1 + \cdots + y_{j(m)-1}) + t\rangle \in (1_a + o(j))g$$

$$\langle o(x_1 + x_2 + \cdots + x_{n-1}) + s,\, i(y_1 + y_2 + \cdots + y_{j(m)-1}) + t\rangle \in (\top_a + 1_{o(x)})(1_a + o(j))g(\bot^b + 1_{i(y)})$$

rezultă enunţul de mai sus. □

În propoziţia anterioară este vorba despre ceea ce în continuare se va numi *simulare prin bijecţii*. Simularea, aşa cum este prezentată în paragraful următor este un concept abstract rezultat din studiul mai multor tipuri de simulări, printre care şi simularea prin bijecţii.

Studiul simulării prin bijecţii va fi reluat în celălalt paragraf unde va fi prezentată simularea prin bijecţii abstracte, adică simulalea prin $a\alpha$-morfisme.

1 Simulare

Fixăm ipotezele în care lucrăm. Fie X un monoid şi Y o X-csms. Fie B un biflux. Fie $i\colon Y \longrightarrow B$ şi $o\colon Y \longrightarrow B$ două morfisme de csms.

Definition 2 Fie $\langle x,\, f\rangle$ şi $\langle y,\, g\rangle$ din $Fl_{X,B}(a,b)$. Spunem că $\langle x,\, f\rangle$ **simulează** prin $j \in Y(x,y)$ în $\langle y,\, g\rangle$ şi scriem

$$\langle x,\, f\rangle \longrightarrow_j \langle y,\, g\rangle$$

dacă $f(1_b + i(j)) = (1_a + o(j))g$.

Spunem că $\langle x,\, f\rangle$ simulează prin Y în $\langle y,\, g\rangle$ şi scriem

$$\langle x,\, f\rangle \longrightarrow_Y \langle y,\, g\rangle$$

dacă există $j \in Y(x,y)$ astfel încât $\langle x,\, f\rangle \longrightarrow_j \langle y,\, g\rangle$. □

Proposition 3 Relaţia $\longrightarrow_Y$ este o preordine compatibilă cu structura de flux a lui $Fl_{X,B}$.

Proof: Pentru *reflexivitate* observăm că

$$\langle x,\, f\rangle \longrightarrow_{1_x} \langle x,\, f\rangle.$$

Pentru *tranzitivitate* demonstrăm că $\langle x,\, f\rangle \longrightarrow_j \langle y,\, g\rangle$ şi $\langle y,\, g\rangle \longrightarrow_k \langle z,\, h\rangle$ implică $\langle x,\, f\rangle \longrightarrow_{jk} \langle z,\, h\rangle$.
Într-adevăr din $f(1_b + i(j)) = (1_a + o(j))g$ şi $g(1_b + i(k)) = (1_a + o(k))h$ deducem

$$f(1_b + i(jk)) = f(1_b + i(j))(1_b + i(k)) = (1_a + o(j))g(1_b + i(k)) = (1_a + o(jk))h.$$

Pentru *compatibilitatea cu compunerea* demonstrăm că $\langle x,\, f\rangle \longrightarrow_j \langle x',\, f'\rangle$ în $Fl_{X,B}(a,b)$ şi $\langle y,\, g\rangle \longrightarrow_k \langle y',\, g'\rangle$ în $Fl_{X,B}(b,c)$ implică $\langle x,\, f\rangle;\langle y,\, g\rangle \longrightarrow_{j+k} \langle x',\, f'\rangle;\langle y',\, g'\rangle$.

Într-adevăr din $f(1_b + i(j)) = (1_a + o(j))f'$ şi $g(1_c + i(k)) = (1_b + o(k))g'$ deducem

$$\begin{aligned}
&[(f + 1_{o(y)})(1_b + {}^{i(x)}\mathrm{X}^{o(y)})(g + 1_{i(x)})(1_c + {}^{i(y)}\mathrm{X}^{i(x)})](1_c + i(j+k)) = \\
&(f + 1_{o(y)})(1_b + {}^{i(x)}\mathrm{X}^{o(y)})(g + 1_{i(x)})[1_c + {}^{i(y)}\mathrm{X}^{i(x)}(i(j) + i(k))] = \\
&(f + 1_{o(y)})(1_b + {}^{i(x)}\mathrm{X}^{o(y)})[g(1_c + i(k)) + i(j)](1_c + {}^{i(y')}\mathrm{X}^{i(x')}) = \\
&(f + 1_{o(y)})(1_b + {}^{i(x)}\mathrm{X}^{o(y)})[(1_b + o(k))g' + i(j)](1_c + {}^{i(y')}\mathrm{X}^{i(x')}) = \\
&(f + 1_{o(y)})[1_b + {}^{i(x)}\mathrm{X}^{o(y)}(o(k) + i(j))](g' + 1_{i(x')})(1_c + {}^{i(y')}\mathrm{X}^{i(x')}) = \\
&[f(1_b + i(j)) + o(k)](1_b + {}^{i(x')}\mathrm{X}^{o(y')})(g' + 1_{i(x')})(1_c + {}^{i(y')}\mathrm{X}^{i(x')}) = \\
&[(1_a + o(j))f' + o(k)](1_b + {}^{i(x')}\mathrm{X}^{o(y')})(g' + 1_{i(x')})(1_c + {}^{i(y')}\mathrm{X}^{i(x')}) = \\
&(1_a + o(j+k))[(f' + 1_{o(y')})(1_b + {}^{i(x')}\mathrm{X}^{o(y')})(g' + 1_{i(x')})(1_c + {}^{i(y')}\mathrm{X}^{i(x')})].
\end{aligned}$$

Pentru *compatibilitatea cu suma* demonstrăm că $\langle x,\, f\rangle \longrightarrow_j \langle x',\, f'\rangle$ în $Fl_{X,B}(a,b)$ şi $\langle y,\, g\rangle \longrightarrow_k \langle y',\, g'\rangle$ în $Fl_{X,B}(c,d)$ implică $\langle x,\, f\rangle + \langle y,\, g\rangle \longrightarrow_{j+k} \langle x',\, f'\rangle + \langle y',\, g'\rangle$.

Într-adevăr din $f(1_b + i(j)) = (1_a + o(j))f'$ şi $g(1_d + i(k)) = (1_c + o(k))g'$ deducem

$$\begin{aligned}
&[(1_a + {}^{c}\mathrm{X}^{o(x)} + 1_{o(y)})(f + g)(1_b + {}^{i(x)}\mathrm{X}^{d} + 1_{i(y)})](1_{bd} + i(j+k)) = \\
&(1_a + {}^{c}\mathrm{X}^{o(x)} + 1_{o(y)})[f(1_b + i(j)) + g(1_d + i(k))](1_b + {}^{i(x')}\mathrm{X}^{d} + 1_{i(y')}) = \\
&(1_a + {}^{c}\mathrm{X}^{o(x)} + 1_{o(y)})[(1_a + o(j))f' + (1_c + o(k))g'](1_b + {}^{i(x')}\mathrm{X}^{d} + 1_{i(y')}) = \\
&(1_{ac} + o(j+k))[(1_a + {}^{c}\mathrm{X}^{o(x')} + 1_{o(y')})(f' + g')(1_b + {}^{i(x')}\mathrm{X}^{d} + 1_{i(y')})].
\end{aligned}$$

Pentru *compatibilitatea cu feedbackul* demonstrăm că $\langle x,\, f\rangle \longrightarrow_j \langle y,\, g\rangle$ în $Fl_{X,B}(ab, ac)$ implică $\uparrow^a \langle x,\, f\rangle \longrightarrow_j \uparrow^a \langle y,\, g\rangle$.
Într-adevăr din $f(1_{ac} + i(j)) = (1_{ab} + o(j))g$ deducem

$$(\uparrow^a f)(1_c + i(j)) = \uparrow^a [f(1_{ac} + i(j))] = \uparrow^a [(1_{ab} + o(j))g] = (1_b + o(j))(\uparrow^a g).$$ □

În cele ce urmează vom utiliza o simplificare a scrierii bazată pe faptul că E_X şi E_B sunt injective. Pentru $x \in X$ în loc de $E_X(x)$ vom scrie pur şi simplu x şi pentru un morfism f din B în loc de $E_B(f)$ vom scrie pur şi simplu f.

Fact 4 În $Fl_{X,B}$ pentru orice $x, y \in X$

$$(x+y)o({}^{x}\mathrm{X}^{y}) \longrightarrow {}_{x}\mathrm{X}^{y}\ i({}^{x}\mathrm{X}^{y})(y+x)$$

Proof: Observăm că

$$(x+y)o({}^{x}\mathrm{X}^{y}) = \langle x+y,\ {}^{i(x+y)}\mathrm{X}^{o(x+y)}(o({}^{x}\mathrm{X}^{y}) + 1_{i(x+y)})\rangle$$

şi că

$$i({}^{x}\mathrm{X}^{y})(y+x) = \langle y+x,\ (i({}^{x}\mathrm{X}^{y}) + 1_{o(y+x)})\,{}^{i(y+x)}\mathrm{X}^{o(y+x)}\rangle.$$

Prin urmare

$$[{}^{i(x+y)}\mathrm{X}^{o(x+y)}(o({}^{x}\mathrm{X}^{y}) + 1_{i(x+y)})](1_{o(y+x)} + i({}^{x}\mathrm{X}^{y})) = {}^{i(x+y)}\mathrm{X}^{o(x+y)}({}^{o(x)}\mathrm{X}^{o(y)} + {}^{i(x)}\mathrm{X}^{i(y)}) =$$

$$({}^{i(x)}\mathrm{X}^{i(y)} + {}^{o(x)}\mathrm{X}^{o(y)})\,{}^{i(y+x)}\mathrm{X}^{o(y+x)} = (1_{i(x+y)} + o({}^{x}\mathrm{X}^{y}))[(i({}^{x}\mathrm{X}^{y}) + 1_{o(y+x)})\,{}^{i(y+x)}\mathrm{X}^{o(y+x)}].\ \square$$

Lemma 5 Fie $i\colon Y \longrightarrow B$ şi $o\colon Y \longrightarrow B$ două morfisme de csmns şi I o interpretare a monoidului obiectelor lui Y în B cu privire la restricţiile lui i şi o la obiectele lui Y cu proprietatea că orice morfism din imaginea lui I permută cu orice morfism din imaginea lui i şi din imaginea lui o.

Fie $\equiv$ o preordine compatibilă cu operaţiile din B. Dacă

$$I(x+y)o({}^{x}\mathrm{X}^{y}) \equiv i({}^{x}\mathrm{X}^{y})I(y+x) \text{ oricare ar fi obiectele } x, y \text{ din } Y,$$

atunci

$$I(x)o(h) \equiv i(h)I(y) \ \text{ pentru orice } \ h \in Y_{a\alpha}(x,y).$$

Proof: Pentru obiectele arbitrare x şi y din Y notăm

$$S(x,y) = \{h \in Y(x,y)\colon\ I(x)o(h) \equiv i(h)I(y)\}.$$

Vom arăta că S este o subcategorie strict monoidală nepermutabilă simetrică a lui Y.

Dacă $h \in S(x,y)$ şi $f \in S(y,z)$, atunci $I(x)o(h) \equiv i(h)I(y)$ şi $I(y)o(f) \equiv i(f)I(z)$, prin urmare

$$I(x)o(hf) = I(x)o(h)o(f) \equiv i(h)I(y)o(f) \equiv i(h)i(f)I(z) = i(hf)I(z)$$

deci S este stabilă la compunere. Din reflexivitatea lui $\equiv$ rezultă că S include morfismele identitate deci este o subcategorie.

Dacă $h \in S(x,y)$ şi $f \in S(z,w)$, atunci $I(x)o(h) \equiv i(h)I(y)$ şi $I(z)o(f) \equiv i(f)I(w)$, prin urmare

$$I(x+z)o(h+f) = I(x)o(h) + I(z)o(f) \equiv i(h)I(y) + i(f)I(w) = i(h+f)I(y+w)$$

deci S este stabilă la sumă. Folosind şi ipoteza deducem că S este subcategorie strict monoidală nepermutabilă simetrică a lui Y. Deoarece $Y_{a\alpha}$ este cea mai mică subcategorie strict monoidală nepermutabilă simetrică a lui Y rezultă că $Y_{a\alpha}$ este inclusă în S, ceea ce probează concluzia. $\square$

În cazul particular al csms şi al egalităţii enunţul lemei devine:

Corollary 6 Fie $i, o : Y \longrightarrow B$ două morfisme de csms şi I o interpretare a lui $Ob(Y)$ în B cu privire la restricţiile lui i şi o la obiecte.

Dacă $f \in Y_{a\alpha}(a,b)$, atunci $I(a)o(f) = i(f)I(b)$.

Proof: Pentru a deduce concluzia aplicând lema 5 egalităţii din B în rolul preordinei $\equiv$ este suficient să probăm ipoteza lemei. Întradevăr $I(ab)o({}^{a}\mathrm{X}^{b}) = (I(a) + I(b))\,{}^{o(a)}\mathrm{X}^{o(b)} = {}^{i(a)}\mathrm{X}^{i(b)}(I(b) + I(a)) = i({}^{a}\mathrm{X}^{b})I(ba).\ \square$

Lemma 7 Dacă $\equiv$ este o congruenţă în $Fl_{X,B}$ cu proprietatea $(x+y)o({}^{x}\mathrm{X}^{y}) \equiv i({}^{x}\mathrm{X}^{y})(y+x)$ pentru orice $x, y \in X$, atunci

1. Simularea prin $Y_{a\alpha}$ este inclusă în $\equiv$.
2. Câtul lui $Fl_{X,B}$ prin $\equiv$ este un biflux.

Proof: Din lema 5 aplicată interpretării standard E_X si congruenţei $\equiv$ deducem pentru orice $j \in Y_{a\alpha}(x,y)$ că

$$x\ o(j) \equiv i(j)\ y.$$

1. Presupunem că $\langle x,\ f\rangle \longrightarrow_j \langle y,\ g\rangle$ în $Fl_{X,B}(a,b)$ cu $j \in Y_{a\alpha}(x,y)$. Deducem $f = (1_a + o(j))g(1_b + i(j^{-1}))$ prin urmare

$\langle x,\ f\rangle = [(1_a + x)f] \uparrow^{i(x)} = [(1_a + xo(j))g(1_b + i(j^{-1}))] \uparrow^{i(x)} = [(1_a + i(j^{-1}))(1_a + xo(j))g] \uparrow^{i(y)} \equiv [(1_a + i(j)^{-1}i(j)y)g] \uparrow^{i(y)} = \langle y,\ g\rangle$.

2. Este suficient să arătăm pentru orice $\langle x,\ f\rangle$ în $Fl_{X,B}(a,b)$ şi pentru orice $\langle y,\ g\rangle$ în $Fl_{X,B}(c,d)$ că

$$\langle x,\ f\rangle + \langle y,\ g\rangle \equiv {}^a\mathrm{X}^c(\langle y,\ g\rangle + \langle x,\ f\rangle)\,{}^d\mathrm{X}^b.$$

Având în vedere prima parte a demonstraţiei este suficient să arătăm pentru orice $\langle x,\ f\rangle$ în $Fl_{X,B}(a,b)$ şi pentru orice $\langle y,\ g\rangle$ în $Fl_{X,B}(c,d)$ că $\langle x,\ f\rangle + \langle y,\ g\rangle$ simulează prin ${}^x\mathrm{X}^y$ în ${}^a\mathrm{X}^c(\langle y,\ g\rangle + \langle x,\ f\rangle)\,{}^d\mathrm{X}^b$.

Deoarece ${}^a\mathrm{X}^c(\langle y,\ g\rangle + \langle x,\ f\rangle)\,{}^d\mathrm{X}^b = \langle yx,\ ({}^a\mathrm{X}^c + 1_{o(yx)})(1_c + {}^a\mathrm{X}^{o(y)} + 1_{o(x)})(g+f)(1_d + {}^{i(y)}\mathrm{X}^b + 1_{i(x)})({}^d\mathrm{X}^b + 1_{i(yx)})\rangle = \langle yx,\ ({}^a\mathrm{X}^{co(y)} + 1_{o(x)})(g+f)({}^{di(y)}\mathrm{X}^b + 1_{i(x)})\rangle$

afirmaţia de mai sus rezultă din urmatoarele egalităţi

$[(1_a + {}^c\mathrm{X}^{o(x)} + 1_{o(y)})(f+g)(1_b + {}^{i(x)}\mathrm{X}^d + 1_{i(y)})](1_{bd} + i({}^x\mathrm{X}^y)) =$
$(1_a + {}^c\mathrm{X}^{o(x)} + 1_{o(y)})\,{}^{ao(x)}\mathrm{X}^{co(y)}(g+f)\,{}^{di(y)}\mathrm{X}^{bi(x)}(1_b + {}^{i(x)}\mathrm{X}^{di(y)}) =$
$(1_a + {}^c\mathrm{X}^{o(x)} + 1_{o(y)})(1_a + {}^{o(x)}\mathrm{X}^{co(y)})({}^a\mathrm{X}^{co(y)} + 1_{o(x)})(g+f)({}^{di(y)}\mathrm{X}^b + 1_{i(x)}) =$
$(1_{ac} + o({}^x\mathrm{X}^y))[({}^a\mathrm{X}^{co(y)} + 1_{o(x)})(g+f)({}^{di(y)}\mathrm{X}^b + 1_{i(x)})]$. □

Fie $\sim_Y$ congruenţa generată de simularea prin Y. Notăm cu $F_X\colon X \longrightarrow Fl_{X,B}/\sim_Y$ compunerea lui E_X cu morfismul de factorizare canonic. Notăm cu $F_B\colon B \longrightarrow Fl_{X,B}/\sim_Y$ compunerea lui E_B cu morfismul de factorizare canonic.

Theorem 8 $Fl_{X,B}/\sim_Y$ este un biflux, F_B este un morfism de bifluxuri şi F_X este o interpretare a lui X în $Fl_{X,B}/\sim_Y$ cu privire la i şi o.

Proof: Din observaţia 4 deducem că $\sim_Y$ verifică ipoteza lemei 7, deci conform acesteia $Fl_{X,B}/\sim_Y$ este un biflux. □

Definition 9 Într-un flux morfismul $j\colon a \longrightarrow b$ se numeşte **functorial** dacă pentru orice $f\colon ca \longrightarrow da$ şi orice $g\colon cb \longrightarrow db$

$$f(1_d + j) = (1_c + j)g \text{ implică } f \uparrow^a = g \uparrow^b .\square$$

Theorem 10 Fie $H\colon B \longrightarrow B'$ un morfism de bifluxuri şi I o interpretare a lui X în B' cu privire la $i; H$ şi $o; H$. Dacă pentru orice $j \in Y(x,y)$

1. $I(x)H(o(j)) = H(i(j))I(y)$
2. $H(i(j))$ este morfism functorial,

atunci există un unic morfism de bifluxuri

$$\langle I,\ H\rangle\colon Fl_{X,B}/\sim_Y \longrightarrow B'$$

cu proprietăţile $F_X; \langle I,\ H\rangle = I$ şi $F_B; \langle I,\ H\rangle = H$.

Proof: Din proprietatea de universalitate a fluxului $Fl_{X,B}$ ştim că există un unic morfism de fluxuri

$$G\colon Fl_{X,B} \longrightarrow B'$$

cu proprietăţile $E_X; G = I$ şi $E_B; G = H$. Reamintim că pentru $\langle x,\ f\rangle$ din $Fl_{X,B}(a,b)$ are loc egalitatea

$$G(x,f) = ((1_{H(a)} + I(x))H(f)) \uparrow^{H(i(x))} .$$

Pentru a putea aplica teorema de universalitate a algebrei cât este suficient să arătăm că $\sim_Y$ este inclusă în congruenţa nucleară a lui G. Deoarece $\sim_Y$ este generată de simularea prin Y este suficient să dovedim că simularea prin Y este inclusă în congruenţa nucleară a lui G. Pentru $\langle x,\ f\rangle$ şi $\langle y,\ g\rangle$ în $Fl_{X,B}(a,b)$ presupunem că $\langle x,\ f\rangle \longrightarrow_Y \langle y,\ g\rangle$, adică există $j \in Y(x,y)$ astfel încât $f(1_b + i(j)) = (1_a + o(j))g$. Deoarece $H(i(j))$ este functorial din egalităţile

$(1_{H(a)} + H(i(j)))[(1_{H(a)} + I(y))H(g)] = (1_{H(a)} + H(i(j))I(y))H(g) = (1_{H(a)} + I(x)H(o(j)))H(g) =$

$(1_{H(a)} + I(x))(1_{H(a)} + H(o(j)))H(g) = (1_{H(a)} + I(x))H((1_a + o(j))g) = (1_{H(a)} + I(x))H(f(1_b + i(j))) =$
$[(1_{H(a)} + I(x))H(f)](1_{H(b)} + H(i(j)))$

deducem

$$((1_{H(a)} + I(x))H(f)) \uparrow^{H(i(x))} = ((1_{H(a)} + I(y))H(g)) \uparrow^{H(i(y))},$$

deci $G(x, f) = G(y, g)$. Din proprietatea de universalitate a algebrei cât rezultă existenţa morfismului de bifluxuri $\langle I, H\rangle : Fl_{X,B}/\sim_Y \longrightarrow B'$. Celelalte proprietăţi sunt uşor de probat. □

2 Bifluxul programelor abstracte

2.1 Preliminarii

Proposition 11 Într-un flux orice izomorfism care permută cu orice alt morfism este functorial.

Proof: Fie $j : a \longrightarrow b$ un izomorfism care permută cu orice alt morfism. Presupunem $f : ca \longrightarrow da$, $g : cb \longrightarrow db$ şi $f(1_d + j) = (1_c + j)g$. Rezultă că $f \uparrow^a = [(1_c + j)g(1_d + j^{-1})] \uparrow^a = [g(1_d + j^{-1})(1_d + j)] \uparrow^b = g \uparrow^b$. □

Corollary 12 Într-un flux orice $a\alpha$-morfism este functorial. □

Fact 13 Dacă orice morfism din Y este izomorfism, atunci $\longrightarrow_Y$ este o congruenţă.

Proof: Probăm că $\longrightarrow_Y$ este simetrică. Presupunem că $\langle x, f\rangle \longrightarrow_j \langle y, g\rangle$ în $Fl_{X,B}(a, b)$. Deoarece j este izomorfism din $f(1_b + i(j)) = (1_a + o(j))g$ deducem $g(1_b + i(j^{-1})) = (1_a + o(j^{-1}))f$ deci $\langle y, g\rangle \longrightarrow_{j^{-1}} \langle x, f\rangle$. □

2.2 Proprietatea de universalitate

Studiul simulării prin $a\alpha$-morfisme este făcut aplicând teoria din paragraful precedent pentru restricţiile morfismelor de csms i şi o la $Y_{a\alpha}$.

Deoarece orice $a\alpha$-morfism este un izomorfism, din observaţia precedentă deducem că simularea prin $a\alpha$-morfisme $\longrightarrow_{Y_{a\alpha}}$ este o congruenţă.

Fie $P_{X,B}$ câtul lui $Fl_{X,B}$ prin $\longrightarrow_{Y_{a\alpha}}$ şi $p^b : Fl_{X,B} \longrightarrow P_{X,B}$ morfismul de factorizare canonic. Bifluxul $P_{X,B}$ va fi numit în cele ce urmază bifluxul programelor abstracte cu instrucţiuni din X şi conexiuni din B.

Fie $P_X = E_X; p^b$ şi $P_B = E_B; p^b$. Observăm că P_X este o interpretare a lui X în $P_{X,B}$ şi că $P_B : B \longrightarrow P_{X,B}$ este un morfism de bifluxuri.

Theorem 14 Pentru orice biflux B', pentru orice morfism de fluxuri $H : B \longrightarrow B'$ şi pentru orice interpretare I a lui X în B' cu privire la $i; H$ şi $o; H$ există un unic morfism de bifluxuri $\langle I, H\rangle^b : P_{X,B} \longrightarrow B'$ cu proprietăţile $P_X; \langle I, H\rangle^b = I$ şi $P_B; \langle I, H\rangle^b = H$.

Proof: Vom aplica teorema 10. Din corolarele 6 şi 12 rezultă că ipotezele teoremei 10 sunt verificate. □

8 CATEGORII STRICT MONOIDALE SIMETRICE ÎMBOGĂŢITE

VEC

February 14, 2010

Fie B o csms. Structura de csms se îmbogăţeşte pentru fiecare obiect a din B cu nişte constante

$$\begin{array}{ll} \top_a \in B(\lambda, a) & \bot^a \in B(a, \lambda) \\ \vee_a \in B(aa, a) & \wedge^a \in B(a, aa) \end{array}$$

Vom folosi în continuare doi parametri $x \in \{a, b, c, d\}$ şi $y \in \{\alpha, \beta, \gamma, \delta\}$. Pentru fiecare pereche de parametri xy se alege o anumită mulţime de operaţii distinse. În cazul $a\alpha$ nu se adauă operaţii, deoarece acesta coincide cu cazul deja studiat. O $a\alpha$-csms coincide cu o csms.

Definition 1 O xy-csms este o csms la care se adaugă operaţiile distinse alese dintre cele de mai sus în conformitate cu tabelul

x	operaţii	y	operaţii
a		α	
b	$\bot^a$	β	$\top_a$
c	$\wedge^a$	γ	$\vee_a$
d	$\bot^a$ şi $\wedge^a$	δ	$\top_a$ şi $\vee_a$.

Aceste operaţii trebuie să satisfacă nişte axiome. Ele se aleg din tabelul următor pentru fiecare caz în parte după regula: se iau toate axiomele care conţin numai operaţii care se găsesc printre cele corespunzătoare cazului în discuţie.

A	$(\vee_a + 1_a); \vee_a = (1_a + \vee_a); \vee_a$	A^o	$\wedge_a; (\wedge_a + 1_a) = \wedge_a; (1_a + \wedge_a)$
B	${}^a\mathrm{X}^a; \vee_a = \vee_a$	B^o	$\wedge_a; {}^a\mathrm{X}^a = \wedge_a$
C	$(\top_a + 1_a); \vee_a = 1_a$	C^o	$\wedge^a; (\bot^a + 1_a) = 1_a$
D	$\vee_a; \bot^a = \bot^a + \bot^a$	D^o	$\top_a; \wedge^a = \top_a + \top_a$

E	$\top_a; \bot^a = 1_\lambda$
F	$\vee_a; \wedge^a = (\wedge^a + \wedge^a)(1_a + {}^a\mathrm{X}^a + 1_a)(\vee_a + \vee_a)$
G	$\wedge^a; \vee_a = 1_a$

SV1	$\top_\lambda = 1_\lambda$	$SV1^o$	$\bot^\lambda = 1_\lambda$
SV2	$\top_{ab} = \top_a + \top_b$	$SV2^o$	$\bot^{ab} = \bot^a + \bot^b$
SV3	$\vee_\lambda = 1_\lambda$	$SV3^o$	$\wedge^\lambda = 1_\lambda$
SV4	$\vee_{ab} = (1_a + {}^b\mathrm{X}^a + 1_b)(\vee_a + \vee_b)$	$SV4^o$	$\wedge^{ab} = (\wedge^a + \wedge^b)(1_a + {}^a\mathrm{X}^b + 1_b)$. □

Proposition 2 Rel_S este o $d\delta$-csms.

Proof: Relaţiile

$$\begin{array}{ll} \top_a \in Rel_S(\lambda, a) & \bot^a \in Rel_S(a, \lambda) \\ \vee_a \in Rel_S(aa, a) & \wedge^a \in Rel_S(a, aa) \end{array}$$

sunt definite prin

$$\begin{array}{ll} \top_a = \emptyset & \bot^a = \emptyset \\ \vee_a = \{(i,i) | i \in [\,|a|\,]\} \cup \{(|a| + i, i) | i \in [\,|a|\,]\} & \wedge^a = \{(i,i) | i \in [\,|a|\,]\} \cup \{(i, |a| + i) | i \in [\,|a|\,]\}. \end{array}$$

Se arată uşor că sunt verificate axiomele.□

1 Extinderea operaţiilor distinse

Fie B o xy-csms. Pentru orice obiect a din B definim morfismele

$$\wedge_m^a \in B(a, a^m) \text{ şi } \vee_a^n \in B(a^n, a)$$

pentru numele naturale m şi n care satisfac, în funcţie de valorile parametrilor xy, condiţiile date în tabelul următor

x	restricţii	y	restricţii
a	$m = 1$	α	$n = 1$
b	$m \leq 1$	β	$n \leq 1$
c	$m \geq 1$	γ	$n \geq 1$
d		δ	

În cazurile $x \in \{b, d\}$ prin definiţie

$$\wedge_0^a = \perp^a.$$

În cazurile $x \in \{a, b, c\}$ prin definiţie

$$\wedge_1^a = 1_a.$$

În cazurile $x \in \{c, d\}$ prin inducţie

$$\wedge_{m+1}^a = \wedge^a(\wedge_m^a + 1_a).$$

Mai observăm că pentru $x \in \{d\}$ egalitatea $\wedge_1^a = 1_a$ este o consecinţă a definiţiei de mai sus şi a axiomei C^o, deci ea este valabilă în toate cazurile. Mai observăm că pentru $x \in \{c, d\}$

$$\wedge_2^a = \wedge^a.$$

În cazurile $y \in \{\beta, \delta\}$ prin definiţie

$$\vee_a^0 = \top_a.$$

În cazurile $y \in \{\alpha, \beta, \gamma\}$ prin definiţie

$$\vee_a^1 = 1_a.$$

În cazurile $y \in \{\gamma, \delta\}$ prin inducţie

$$\vee_a^{n+1} = (\vee_a^n + 1_a) \vee_a .$$

Mai observăm că pentru $y \in \{\delta\}$ egalitatea $\vee_a^1 = 1_a$ este o consecinţă a definiţiei de mai sus şi a axiomei C, deci ea este valabilă în toate cazurile. Mai observăm că pentru $y \in \{\gamma, \delta\}$

$$\vee_a^2 = \vee_a.$$

Vom demonstra în continuare diverse proprietăţi ale operaţiilor definite mai sus. Pentru a micşora numărul cazurilor analizate fapt ce duce la simplificarea demonstraţiilor vom fave unele observaţii.

1) Putem folosi principiul dualităţii. Conceptul de csms este autodual. Dualizăm schibând ${}^a\mathrm{X}^b$ cu ${}^b\mathrm{X}^a$. Pentru xy-csms dualizăm permutând $\vee_a$ cu $\wedge^a$ şi $\top_a$ cu $\perp^a$.

2) Dacă o egalitate conţine numai operaţiile $\wedge_m$(respectiv $\vee^n$), putem reduce studiul acesteia numai la cazul $x\alpha$(respectiv ay).

3) Dacă într-o egalitate toate constantele $\wedge$ (respectiv $\vee$) apar la aceeaşi putere m (respectiv n) studiul cazului dy (respectiv $x\delta$) poate fi redus la cazurile by şi cy (respectiv $x\beta$ şi $x\gamma$).

Mai menţionăm faptul că vom folosi $\sum$ în cazul necomutativ având grijă să utilizăm o mulţime total ordonată de indici care indică ordinea termenilor.

Proposition 3 Asociativitatea generalizată.

$$\wedge_m^a \left(\sum_{j \in [m]} \wedge_{m_j}^a \right) = \wedge^a_{\sum_{j \in [m]} m_j} \text{ dacă } m_j \text{ şi } m \text{ satisfac } x \text{ şi}$$

$$\left(\sum_{i \in [n]} \vee_a^{n_i} \right) \vee_a^n = \vee_a^{\sum_{i \in [n]} n_i} \text{ dacă } n_i \text{ şi } n \text{ satisfac } y.$$

Proof: Folosind principiul dualităţii este suficient să demonstrăm numai a doua egalitate.

Pentru $n = 0$ remarcăm că suma din membrul stang este 1_e, ceea ce implică adevărul egalităţii. Pentru $n = 1$ egalitatea este evidentă.

Utilizând a doua observaţie de mai sus este suficient să analizăm cazurile ay. Deoarece pentru $n \in \{0, 1\}$ egalitatea este demonstrată rămâne să studiem cazurile $a\gamma$ şi $a\delta$.

În cazul $a\gamma$ demonstraţia se face prin inducţie după n folosind numai axioma A şi este inclusă în demonstraţia cazului $a\delta$ pe care-l demonstrăm în continuare.

Pentru $n = 2$ probăm prin inducţie după j că

$$(\vee_a^i + \vee_a^j)\vee_a = \vee_a^{i+j}.$$

Din axioma C utilizând axioma B deducem $(1_a + \top_a); \vee_a = 1_a$, fapt care implică egalitatea de mai sus pentru $j = 0$. Pentru $j = 1$ egalitatea este evident adevărată.

Pasul inductiv este

$$(\vee_a^i + \vee_a^{j+1})\vee_a = (\vee_a^i + \vee_a^j + 1_a)(1_a + \vee_a)\vee_a = (\vee_a^i + \vee_a^j + 1_a)(\vee_a + 1_a)\vee_a = (\vee_a^{i+j} + 1_a)\vee_a = \vee_a^{i+j+1}.$$

Pentru $n \geq 3$ pasul inductiv este

$$\left(\sum_{i\in[n]} \vee_a^{n_i}\right) \vee_a^n = \left(\sum_{i\in[n-1]} \vee_a^{n_i} + \vee_a^{n_n}\right) (\vee_a^{n-1} + 1_a)\vee_a = (\vee_a^{\sum_{i\in[n-1]} n_i} + \vee_a^{n_n})\vee_a = \vee_a^{\sum_{i\in[n]} n_i}$$

□

Proposition 4 Comutativitatea generalizată rezultă din axiomele A,B şi dualele lor.
Dacă $F : Bi_S \to B$ este un morfism de csms, $s \in S$ şi $F(s) = a$, atunci:

$$\wedge_m^a F(f) = \wedge_m^a, \text{ pentru orice } f \in Bi_S(s^m, s^m), \text{ dacă } m \text{ satisface } x;$$

$$F(f)\vee_a^n = \vee_a^n, \text{ pentru orice } f \in Bi_S(s^n, s^n), \text{ dacă } n \text{ satisface } y.$$

Proof: Utilizând dualitatea probăm numai a doua egalitate. Deoarece orice $f \in Bi_S(s^n, s^n)$ poate fi scris ca o compunere de bijecţii de forma $1_{s^i} + {}^s\mathrm{X}^s + 1_{s^{n-i-2}}$ unde $0 \leq i \leq n-2$, este suficient să probăm egalitatea numai pentru o astfel de bijecţie.

$$F(1_{s^i} + {}^s\mathrm{X}^s + 1_{s^{n-i-2}})\vee_a^n = (1_{a^i} + {}^a\mathrm{X}^a + 1_{a^{n-i-2}})(1_{a^i} + \vee_a + 1_{a^{n-i-2}})\vee_a^{n-1} = (1_{a^i} + \vee_a + 1_{a^{n-i-2}})\vee_a^{n-1} = \vee_a^n$$

□

Pentru generalizarea axiomei F este necesar să introducem bijecţia

$$\varphi_{n,m}^s \in Bi_S((mn)s, (nm)s)$$

definită pentru $s \in S$, $n \geq 0$ şi $m \geq 0$ prin

a) $\varphi_{0,m}^s = \varphi_{n,0}^s = 1_\lambda$

b) pentru $n \geq 1$ şi $m \geq 1$

$$\varphi_{n,m}^s(j) = (r-1)n + k + 1 \text{ dacă şi numai dacă } j = km + r \text{ şi } 1 \leq r \leq m.$$

Observăm că $\varphi_{1,m}^s$ şi $\varphi_{n,1}^s$ sunt identităţi. Mai remarcăm că

$$\varphi_{2,m+1}^s = (1_{ms} + {}^s\mathrm{X}^{ms} + 1_s)(\varphi_{2,m}^s + 1_{2s}).$$

Proposition 5 Axioma F generalizată.
Dacă $F : Bi_S \to B$ este un morfism de csms, $s \in S$ şi $F(s) = a$, atunci:

$$\vee_a^n \wedge_m^a = \left(\sum_{i\in[n]} \wedge_m^a\right) F(\varphi_{n,m}^s) \left(\sum_{i\in[m]} \vee_a^n\right)$$

dacă m satisface x şi n satisface y.

Proof: Observăm că pentru $m = 1$ sau $n = 1$ egalitatea este adevărată. Prin urmare cazurile $x = a$ sau $y = \alpha$ sunt demonstrate. Rămân 9 cazuri. Ţinând cont de observaţia a treia rămân cazurile $b\beta$, $b\gamma$, $c\beta$ şi $c\gamma$. Folosind dualitatea categorială rămân numai cazurile $b\beta$, $b\gamma$ şi $c\gamma$.

Cazul $b\beta$ care trebuie probat numai pentru $n = 0 = m$ este acoperit de axioma E.

Cazul $b\gamma$ trebuie deasemenea probat numai $m = 0$. Vom utiliza numai axioma D. Facem inducţie după n:

$$\vee_a^{n+1}\perp^a = (\vee_a^n + 1_a)\vee_a \perp^a = (\vee_a^n + 1_a)(\perp^a + \perp^a) = \sum_{i\in[n]} \perp^a + \perp^a = \sum_{i\in[n+1]} \perp^a.$$

Cazul $c\gamma$ se demonstrează utilizând axioma F.

Începem prin a introduce o altă funcţie auxiliară

$$\Psi_m \in Bi_S(m(ns) + ms, m((n+1)s)) \text{ unde } m, n \geq 1.$$

Prin definiţie

$$\begin{array}{ll} \Psi_m(in + k) = i(n + 1) + k & \text{dacă } \ 0 \leq i < m \text{ şi } k \in [n] \\ \Psi_m(mn + i) = i(n + 1) & \text{dacă } \ i \in [m] \end{array}$$

Remarcăm că

$$\begin{array}{l} \Psi_1 = 1_{(n+1)s} \\ \Psi_{m+1} = (1_{m(ns)} + {}^{ns}\mathrm{X}^{ms} + 1_s)(\Psi_m + 1_{(n+1)s}) \end{array}$$

Probăm prin inducţie după m că

$$\left(\sum_{i\in[\,m\,]} \vee_a^n + 1_{ma}\right) F(\varphi^s_{2,m}) = F(\Psi_m)\left(\sum_{i\in[\,m\,]} (\vee_a^n + 1_a)\right)$$

Deoarece pentru $m = 1$ egalitatea este evident adevărată trecem la pasul inductiv

$$\left(\sum_{i\in[\,m+1\,]} \vee_a^n + 1_{(m+1)a}\right) F(\varphi^s_{2,m+1}) = \left[\sum_{i\in[\,m\,]} \vee_a^n + (\vee_a^n + 1_{ma})\,{}^a\mathrm{X}^{ma} + 1_a\right](F(\varphi^s_{2,m}) + 1_{2a}) =$$

$$(1_{m(na)} + {}^{na}\mathrm{X}^{ma} + 1_a)\left[\left(\sum_{i\in[\,m\,]} \vee_a^n + 1_{ma}\right) F(\varphi^s_{2,m}) + \vee_a^n + 1_a\right] =$$

$$F((1_{m(ns)} + {}^{ns}\mathrm{X}^{ms} + 1_s)(\Psi_m + 1_{(n+1)s}))\left(\sum_{i\in[\,m+1\,]} (\vee_a^n + 1_a)\right) = F(\Psi_{m+1})\left(\sum_{i\in[\,m+1\,]} (\vee_a^n + 1_a)\right)$$

Demonstraţia generalizării lui F se face prin inducţie. Prin inducţie după m vom face demonstraţia pentru $n = 2$. Deoarece cazul $m = n = 2$ e acoperit de F trecem la pasul inductiv

$$\vee_a^2\wedge^a_{m+1} = \vee_a \wedge^a (\wedge^a_m + 1_a) = (\wedge^a + \wedge^a)(1_a + {}^a\mathrm{X}^a + 1_a)(\vee_a + \vee_a)(\wedge^a_m + 1_a) =$$

$$(\wedge^a + \wedge^a)(1_a + {}^a\mathrm{X}^a + 1_a)\left[(\wedge^a_m + \wedge^a_m)F(\varphi^s_{2,m})\left(\sum_{i\in[\,m\,]} \vee_a\right) + \vee_a\right] =$$

$$(\wedge^a + \wedge^a)(\wedge^a_m + {}^a\mathrm{X}^a(\wedge^a_m + 1_a) + 1_a)(F(\varphi^s_{2,m}) + 1_{2a})\left(\sum_{i\in[\,m+1\,]} \vee_a\right) =$$

$$(\wedge^a + \wedge^a)(\wedge^a_m + 1_a + \wedge^a_m + 1_a)(1_{ma} + {}^a\mathrm{X}^{ma} + 1_a)(F(\varphi^s_{2,m}) + 1_{2a})\left(\sum_{i\in[\,m+1\,]} \vee_a\right) =$$

$$(\wedge^a_{m+1} + \wedge^a_{m+1})F((1_{ms} + {}^s\mathrm{X}^{ms} + 1_s)(\varphi^s_{2,m}) + 1_{2s}))\left(\sum_{i\in[\,m+1\,]} \vee_a\right) = \left(\sum_{i\in[\,2\,]} \wedge^a_{m+1}\right) F(\varphi^s_{2,m+1})\left(\sum_{i\in[\,m+1\,]} \vee_a^2\right).$$

În final facem o inducţie după n pentru m fixat.

$$\vee_a^{n+1}\wedge_m^a = (\vee_a^n + 1_a)\vee_a \wedge_m^a = (\vee_a^n + 1_a)(\wedge_m^a + \wedge_m^a)F(\varphi^s_{2,m})\left(\sum_{i\in[\,m\,]}\vee_a\right) =$$

$$\left[\left(\sum_{i\in[\,n\,]}\wedge_m^a\right)F(\varphi^s_{n,m})\left(\sum_{i\in[\,m\,]}\vee_a^n\right) + \wedge_m^a\right]F(\varphi^s_{2,m})\left(\sum_{i\in[\,m\,]}\vee_a\right) =$$

$$\left(\sum_{i\in[\,n+1\,]}\wedge_m^a\right)[F(\varphi^s_{n,m}) + 1_{ma}]\left[\sum_{i\in[\,m\,]}\vee_a^n + 1_{ma}\right]F(\varphi^s_{2,m})\left(\sum_{i\in[\,m\,]}\vee_a\right) =$$

$$\left(\sum_{i\in[\,n+1\,]}\wedge_m^a\right)[F(\varphi^s_{n,m}) + 1_{ma}]F(\Psi_m)\left(\sum_{i\in[\,m\,]}(\vee_a^n + 1_a)\right)\left(\sum_{i\in[\,m\,]}\vee_a\right) =$$

$$\left(\sum_{i\in[\,n+1\,]}\wedge_m^a\right)F((\varphi^s_{n,m}) + 1_{ms})\Psi_m)\left(\sum_{i\in[\,m\,]}(\vee_a^n + 1_a)\vee_a\right) = \left(\sum_{i\in[\,n+1\,]}\wedge_m^a\right)F(\varphi^s_{n+1,m})\left(\sum_{i\in[\,m\,]}\vee_a^{n+1}\right).$$

□

Proposition 6 Axioma G generalizată.

$$\wedge_m^a\vee_a^m = 1_a \text{ dacă } m \geq 1 \text{ satisface } x \text{ şi } y$$

Prin inducţie după m: $\wedge^a_{m+1}\vee_a^{m+1} = \wedge^a(\wedge_m^a + 1_a)(\vee_a^m + 1_a)\vee_a = \wedge^a(1_a + 1_a)\vee_a = 1_a$. □

2 Morfisme de xy-csms

Definition 7 Fie B şi B' două xy-csms. Un morfisme de csms $H : B \to B'$ se numeşte morfism de xy-csms dacă pentru orice obiect a din B satisface condiţiile:

$$H(\top_a) = \top_{H(a)} \quad \text{dacă } y \in \{\beta, \delta\},$$

$$H(\bot^a) = \bot^{H(a)} \quad \text{dacă } x \in \{b, d\},$$

$$H(\vee_a) = \vee_{H(a)} \quad \text{dacă } y \in \{\gamma, \delta\} \text{ şi}$$

$$H(\wedge^a) = \wedge^{H(a)} \quad \text{dacă } x \in \{c, d\}.\square$$

Proposition 8 Dacă $H : B \to B'$ este un morfism de xy-csms, atunci pentru orice obiect a din B:

$$H(\vee_a^n) = \vee^n_{H(a)} \quad \text{dacă } n \text{ satisface } y \text{ şi}$$

$$H(\wedge_m^a) = \wedge_m^{H(a)} \quad \text{dacă } m \text{ satisface } x.$$

Demonstraţia se face uşor prin inducţie. □

Definition 9 O xy-csms în care monoidul obiectelor este M se numeşte M-xy-csms. Un morfism H de xy-csms între două M-xy-csms se numeşte morfism de M-xy-csms dacă $H(a) = a$ pentru orice $a \in M$. □

Proposition 10 Dacă B este o xy-csms, atunci $Fl_{X,B}/\sim_Y$ este o xy-csms şi F_B este un morfism de xy-csms.

Proof: Constantele distinse din structura cât sunt imaginea prin F_B a constantelor distinse din B. □

3 xy-morfisme într-o xy-csms

Definition 11 O sub-csms a unei xy-csms B se numeşte sub-xy-csms dacă conţine toate morfismele distinse corespunzătoare parametrilor xy. □

Definition 12 Într-o xy-csms B vom nota cu B_{xy} cea mai mică sub-xy-csms a lui B. Morfismele din B_{xy} se vor numi xy-morfisme. □

$$B_{xy} \hookrightarrow B$$

Fact 13 Imaginea unui xy-morfism printr-un morfism de xy-csms este un xy-morfism.

Proof: Se ştie că orice xy-morfism este o compunere de morfisme de forma $1_a + f + 1_d$ unde f este un morfism distins. Prin inducţie după numărul factorilor compunerii. □

Proposition 14 Dacă $H : B \longrightarrow B'$ este un morfism de M-xy-csms atunci, orice xy-morfism din B' este imaginea prin H a unui xy-morfism din B.

Proof: Acelaşi argument ca în propoziţia anterioară. □