Rencontre INVAL
Complexité des programmes polygraphiques

Guillaume Bonfante – Yves Guiraud

Lyon – 26 septembre 2008

Complexité des programmes polygraphiques

Introduction

Limitations des interprétations polynomiales de termes

Bornes mixtes. Addition : $0 + x \to x \qquad s(x) + y \to s(x + y)$

Interprétation polynomiale : $(\!|0|\!) = 1 \qquad (\!|s|\!)(X) = X + 1 \qquad (\!|+|\!)(X, Y) = 2X + Y + 1$

Le polynôme $2X + Y + 1$ majore la taille du résultat ($X + Y$) et le temps de calcul (X).

Limitations des interprétations polynomiales de termes

Bornes mixtes. Addition : $0 + x \to x \qquad s(x) + y \to s(x+y)$

Interprétation polynomiale : $(\!|0|\!) = 1 \qquad (\!|s|\!)(X) = X+1 \qquad (\!|+|\!)(X,Y) = 2X+Y+1$

Le polynôme $2X+Y+1$ majore la taille du résultat ($X+Y$) et le temps de calcul (X).

Récursion non structurelle. Division entière : $0/y \to 0 \qquad s(x)/y \to s((x-y)/y)$

Les propriétés de $(\!|\cdot|\!)$ imposent $(\!|\, s(x)/s(x) \,|\!) \leq (\!|\, s((x - s(x))/s(x)) \,|\!)$ au lieu de $>$.

Limitations des interprétations polynomiales de termes

Bornes mixtes. Addition : $\quad 0+x \to x \qquad s(x)+y \to s(x+y)$

Interprétation polynomiale : $\quad \llparenthesis 0 \rrparenthesis = 1 \qquad \llparenthesis s \rrparenthesis(X) = X+1 \qquad \llparenthesis + \rrparenthesis(X,Y) = 2X+Y+1$

Le polynôme $2X+Y+1$ majore la taille du résultat ($X+Y$) et le temps de calcul (X).

Récursion non structurelle. Division entière : $\quad 0/y \to 0 \qquad s(x)/y \to s((x-y)/y)$

Les propriétés de $\llparenthesis \cdot \rrparenthesis$ imposent $\quad \llparenthesis s(x)/s(x) \rrparenthesis \leq \llparenthesis s((x-s(x))/s(x)) \rrparenthesis \quad$ au lieu de $>$.

Fonctions à plusieurs sorties. Séparation de listes (tri fusion) :

$$(x_1, x_2, x_3, x_4, \ldots) \mapsto \big((x_1, x_3, \ldots), (x_2, x_4, \ldots)\big)$$

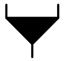
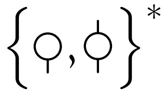
Algorithme de « dézippage » par réécriture de termes (pas du premier ordre) :

$$\delta(\square) \to (\square, \square) \qquad \delta(x :: \square) \to (x :: \square, \square)$$

$$\delta(x :: y :: L) \to \text{ let } (L_1, L_2) = \delta(L) \text{ in } (x :: L_1, y :: L_2)$$

Des systèmes de réécriture de termes aux polygraphes

Traduction des systèmes de réécriture de termes. Exemple : addition et multiplication

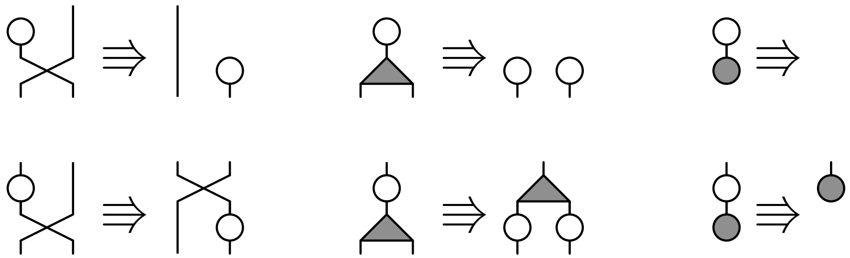
Calcule l'addition  et la multiplication  d'entiers naturels .

Des systèmes de réécriture de termes aux polygraphes

Traduction des systèmes de réécriture de termes. Exemple : addition et multiplication

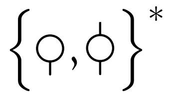
Calcule l'addition et la multiplication d'entiers naturels $\{\;,\;\}^*$.

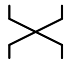
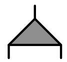
Structure cartésienne explicite. Permutation , duplication et effacement explicites :

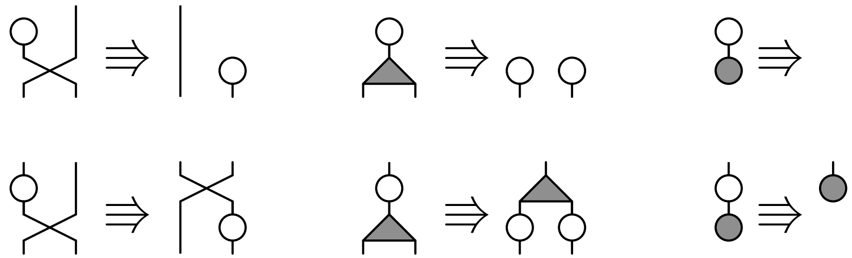


Des systèmes de réécriture de termes aux polygraphes

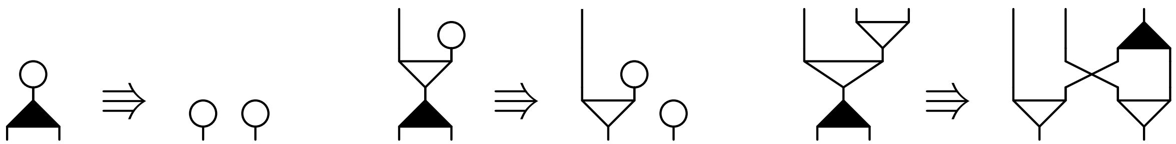
Traduction des systèmes de réécriture de termes. Exemple : addition et multiplication

Calcule l'addition  et la multiplication  d'entiers naturels .

Structure cartésienne explicite. Permutation , duplication  et effacement  explicites :



Fonctions de coarité quelconque (à plusieurs sorties).



Calcule la séparation de listes  par « dézippage ».

Interprétations « courants » / « chaleurs »

Intuitions électriques.

● Chaque circuit est traversé par des courants descendants et/ou ascendants.

● Les portes produisent de la chaleurs en fonction de l'intensité des courants incidents.

● Les circuits sont comparés en fonction de la chaleur qu'ils produisent, à intensité égale.

« Théorème. » Si, à chaque application d'une règle de calcul, la chaleur produite diminue strictement, on a la terminaison.

Question :

> Peut-on décrire les programmes par des polygraphes et utiliser des « courants » et des « chaleurs » pour borner leur taille ?

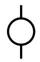
1. Programmes polygraphiques

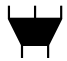
Programmes polygraphiques

Un **programme polygraphique** est un 3-polygraphe Σ *fini* avec :

- Exactement une 0-cellule, notée I.

- Des 2-cellules de deux types :

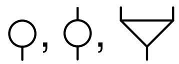
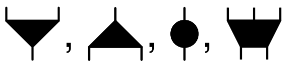
  – Constructeurs (Σ_{2C}) : de coarité 1, comme , , , etc.

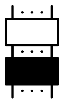
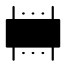
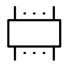
  – Fonctions (Σ_{2F}) : sans restriction, comme , , , , etc.

Programmes polygraphiques

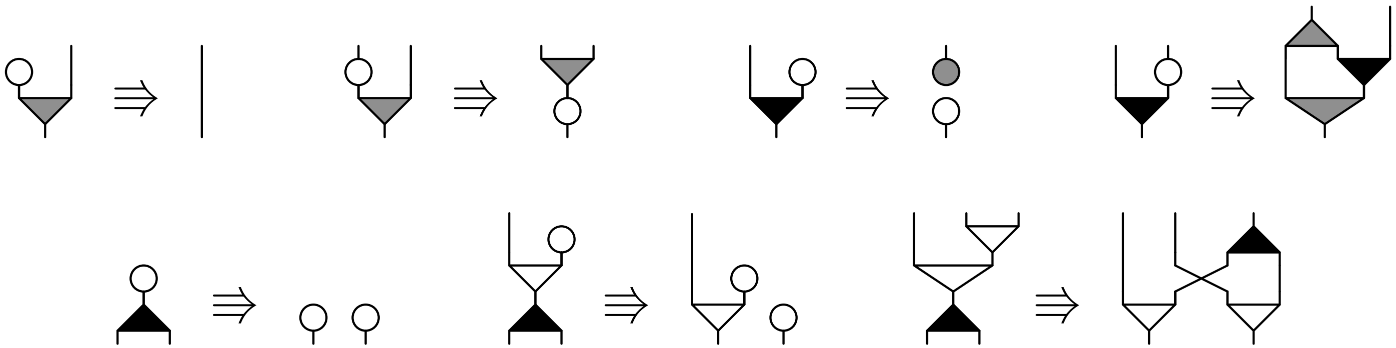
Un **programme polygraphique** est un 3-polygraphe Σ *fini* avec :

- Exactement une 0-cellule, notée I.

- Des 2-cellules de deux types :
  - Constructeurs (Σ_{2C}) : de coarité 1, comme , etc.
  - Fonctions (Σ_{2F}) : sans restriction, comme , etc.

- Des 3-cellules, dont la source est de la forme  avec  $\in \Sigma_{2F}$ et  $\in \Sigma_{2C}^{*}$.

Comme dans les exemples :



Sémantique

Types et valeurs

Les 1-cellules sont les **types**.

Les 2-cellules de Σ^*_{2C} de source I sont les **valeurs**. Le type d'une valeur est son but.
On note $[\![u]\!]$ l'ensemble des valeurs de type u, $u \in \Sigma^*_1$.

Toute valeur $\boxed{\cdots}$ se décompose en valeurs de coarité 1 :

$$\boxed{\cdots} = \bigtriangledown_1 \star_0 \cdots \star_0 \bigtriangledown_n .$$

Sémantique

Types et valeurs

Les 1-cellules sont les **types**.

Les 2-cellules de Σ^*_{2C} de source I sont les **valeurs**. Le type d'une valeur est son but.
On note $[\![u]\!]$ l'ensemble des valeurs de type u, $u \in \Sigma^*_1$.

Toute valeur se décompose en valeurs de coarité 1 :

$$= \triangledown_1 \star_0 \cdots \star_0 \triangledown_n \, .$$

Fonctions calculées

Le programme est **complet** s'il peut réduire toute 2-cellule avec une fonction et une valeur.

Proposition. Si le programme est convergent et complet, $[\![\cdot]\!]$ s'étend en un 2-foncteur $\Sigma^*_2 \to \mathbf{Ens}$:

$$\left[\!\!\left[\;: u \Rightarrow v \right]\!\!\right] \; : \; [\![u]\!] \longrightarrow [\![v]\!]$$

$\mapsto$ unique forme normale de .

Une application $f : [\![u]\!] \to [\![v]\!]$ est **calculée par Σ** s'il existe une fonction $: u \Rightarrow v$ telle que $\left[\!\!\left[\; \right]\!\!\right] = f$.

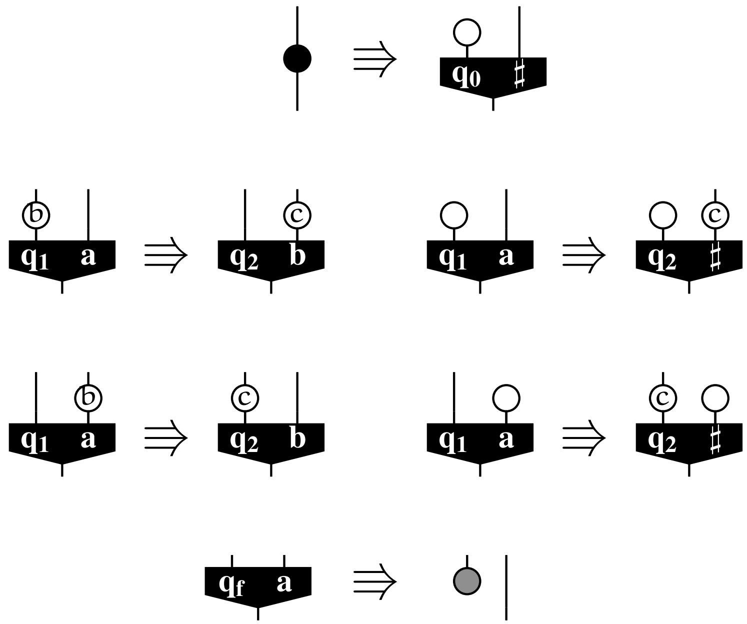
Calculabilité

Théorème. Les programmes polygraphiques forment un modèle de calcul Turing-complet.

Calculabilité

Théorème. Les programmes polygraphiques forment un modèle de calcul Turing-complet.

Preuve. On construit une machine de Turing polygraphique :



if $\delta(q_1, a) = (q_2, c, L)$

if $\delta(q_1, a) = (q_2, c, R)$

Constructeurs : $\Sigma_2^C = \left\{ \circ, ⓐ, \ldots \right\}$ pour les mots.

Fonctions : $\bullet$ (celle que l'on veut calculer) et $\boxed{q \; a}$ (représente l'état q, lisant a, les parties gauche et droite du ruban dans les entrées).

2. Dérivations et complexité des programmes polygraphiques

Taille des programmes polygraphiques

Problème informel : Borner la taille des calculs menant d'une 2-cellule à son résultat, avec une fonction et une valeur, en fonction de la taille de .

Taille des programmes polygraphiques

Problème informel : Borner la taille des calculs menant d'une 2-cellule à son résultat, avec une fonction et une valeur, en fonction de la taille de .

Taille des cellules. On note $\|\ \|$ et $|||\ |||$ la **taille** d'une 2-cellule et d'une 3-cellule , c'est-à-dire le nombre de 2-cellules ou 3-cellules génératrices qui les composent.

Problème : Soit une fonction de source u et d'arité n.

On cherche une application $P : \mathbb{N}^n \to \mathbb{N}$ telle que, pour toute valeur $= \bigtriangledown_1 \star_0 \cdots \star_0 \bigtriangledown_n$ de type u, pour toute 3-cellule de source , on ait :

$$|||\ ||| \leq P\left(\|\bigtriangledown_1\|, \ldots, \|\bigtriangledown_n\|\right).$$

Interprétations fonctorielles (courants) et taille des valeurs

Définitions. Un 2-foncteur $F : \Sigma_2^* \to \mathbf{Ord}$ est **compatible** si, pour toute 3-cellule génératrice $\alpha : f \Rrightarrow g$, on a $F(f) \geq F(g)$. Il est **additif** si :

- Pour toute 1-cellule génératrice u, on a $F(u) \subseteq \mathbb{N}$.
- Pour tout constructeur $\triangledown$ d'arité n, il existe un entier non nul $a_{\triangledown}$ tel que :

$$F\left(\triangledown\right)(x_1, \ldots, x_n) = x_1 + \cdots + x_n + a_{\triangledown}.$$

Dans ce cas, on note :

$$a = \max\left\{ a_{\triangledown}, \triangledown \in \Sigma_{2C} \right\}.$$

Interprétations fonctorielles (courants) et taille des valeurs

Définitions. Un 2-foncteur $F : \Sigma_2^* \to \mathbf{Ord}$ est **compatible** si, pour toute 3-cellule génératrice $\alpha : f \Rrightarrow g$, on a $F(f) \geq F(g)$. Il est **additif** si :

- Pour toute 1-cellule génératrice u, on a $F(u) \subseteq \mathbb{N}$.
- Pour tout constructeur $\overset{\cdots}{\bigtriangledown}$ d'arité n, il existe un entier non nul $a_{\overset{\cdots}{\bigtriangledown}}$ tel que :

$$F\left(\overset{\cdots}{\bigtriangledown}\right)(x_1, \ldots, x_n) = x_1 + \cdots + x_n + a_{\overset{\cdots}{\bigtriangledown}}.$$

Dans ce cas, on note :

$$a = \max\left\{a_{\overset{\cdots}{\bigtriangledown}}, \overset{\cdots}{\bigtriangledown} \in \Sigma_{2C}\right\}.$$

Proposition. Soit F est un 2-foncteur additif. Pour toute valeur $\bigtriangledown$ de coarité 1, on a :

$$\left|\left|\bigtriangledown\right|\right| \leq F\left(\bigtriangledown\right) \leq a\left|\left|\bigtriangledown\right|\right|.$$

Dérivations (chaleurs)

On fixe un 2-foncteur $F : \Sigma_2^* \to \mathbf{Ord}$.

Une **dérivation de Σ_2^* à valeurs dans F** est une application d qui envoie chaque 2-cellule f d'arité n de Σ_2^* sur une application monotone

$$d(f) : \mathbb{N}^n \to \mathbb{N},$$

de telle sorte qu'on ait :

$$d\left(\boxed{f}\,\boxed{g}\right) = d\left(\boxed{f}\right) \circ \pi_1 + d\left(\boxed{g}\right) \circ \pi_2 \qquad \text{et} \qquad d\left(\begin{array}{c}\boxed{f}\\ \boxed{g}\end{array}\right) = d\left(\boxed{f}\right) + d\left(\boxed{g}\right) \circ F\left(\boxed{f}\right).$$

Cela entraîne que $d(1_u) = 0$ pour toute 1-cellule u :

$$d(1_u) = d(1_u \star_1 1_u) = d(1_u) + d(1_u) \circ F(1_u) = 2d(1_u).$$

Une dérivation d est **compatible** lorsque, pour toute 3-cellule génératrice $\alpha : f \Rrightarrow g$, on a $d(f) > d(g)$.

Taille des calculs

Théorème. Soient $F : \Sigma_2^* \to \mathbf{Ord}$ un 2-foncteur additif et compatible et d une dérivation de Σ_2^* dans F compatible et nulle sur les constructeurs.

On considère une fonction , de source u et d'arité n, et une valeur $= \bigtriangledown_1 \star_0 \cdots \star_0 \bigtriangledown_n$ de type u.

Alors, pour toute 3-cellule de source , on a :

$$\left|\left|\left| \quad \right|\right|\right| \leq P \left(|| \bigtriangledown_1 ||, \ldots, || \bigtriangledown_n || \right),$$

avec $P(x_1, \ldots, x_n) = d \left(\quad \right) (a x_1, \ldots, a x_n)$.

Taille des calculs

Théorème. Soient $F : \Sigma_2^* \to \mathbf{Ord}$ un 2-foncteur additif et compatible et d une dérivation de Σ_2^* dans F compatible et nulle sur les constructeurs.

On considère une fonction , de source u et d'arité n, et une valeur $= \bigtriangledown_1 \star_0 \cdots \star_0 \bigtriangledown_n$ de type u.

Alors, pour toute 3-cellule de source , on a :

$$\left|\left|\left| \quad \right|\right|\right| \leq P\left(||\bigtriangledown_1||, \ldots, ||\bigtriangledown_n||\right),$$

avec $P(x_1, \ldots, x_n) = d\left(\quad\right)(ax_1, \ldots, ax_n)$.

Preuve . On décompose en 3-cellules de taille 1 : $= {}_1 \star_2 \cdots \star_2 {}_{|||\ |||}$.

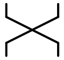
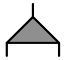
Alors : $d\left(\quad\right) > d\left(t(\ _1)\right) > \ldots > d\left(t(\ _{|||\ |||})\right) \geq 0.$ D'où : $\left|\left|\left| \quad \right|\right|\right| \leq d\left(\quad\right)$.

De plus :

$$\begin{aligned} d\left(\quad\right) &= d\left(\quad\right) + d\left(\quad\right) \circ F\left(\quad\right) \\ &\leq d\left(\quad\right)\left(a||\bigtriangledown_1||, \ldots, a||\bigtriangledown_n||\right). \end{aligned}$$

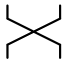
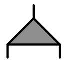
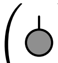
Exemple

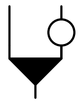
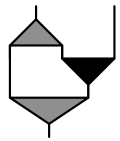
Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

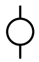
$F\Big($  $\Big)(x, y) = (y, x),\quad F\Big($  $\Big)(x) = (x, x),\quad F\Big($  $\Big)(x) = *$

Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

$F\Big($  $\Big)(x,y) = (y,x)$, $\quad F\Big($  $\Big)(x) = (x,x)$, $\quad F\Big($  $\Big)(x) = *$

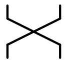
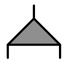
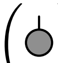
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

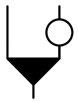
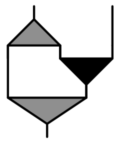
$F\Big($  $\Big) = 1$, $\quad F\Big($  $\Big)(x) = X+1$, $\quad F\Big($  $\Big)(x,y) = x+y$, $\quad F\Big($  $\Big)(x,y) = xy$,

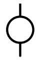
$d\Big($  $\Big)(x,y) = x$, $\quad d\Big($  $\Big)(x,y) = (x+1)y$.

Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

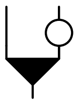
$F\Big($  $\Big)(x,y) = (y,x),\quad F\Big($  $\Big)(x) = (x,x),\quad F\Big($  $\Big)(x) = *$

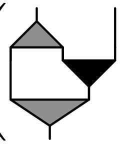
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

$F\Big($  $\Big) = 1,\quad F\Big($  $\Big)(x) = X+1,\quad F\Big($  $\Big)(x,y) = x+y,\quad F\Big($  $\Big)(x,y) = xy,$

$d\Big($  $\Big)(x,y) = x,\quad d\Big($  $\Big)(x,y) = (x+1)y.$

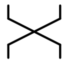
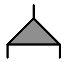
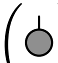
On calcule : $\Bigg\{$

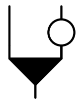
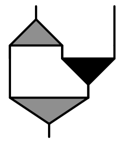
$d\Big($  $\Big)(x,y)$

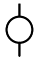
$d\Big($  $\Big)(x,y)$

Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

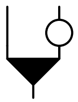
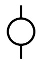
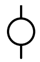
$F\Big($  $\Big)(x,y) = (y,x),\quad F\Big($  $\Big)(x) = (x,x),\quad F\Big($  $\Big)(x) = *$

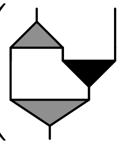
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

$F\Big($  $\Big) = 1,\quad F\Big($  $\Big)(x) = X+1,\quad F\Big($  $\Big)(x,y) = x+y,\quad F\Big($  $\Big)(x,y) = xy,$

$d\Big($  $\Big)(x,y) = x,\quad d\Big($  $\Big)(x,y) = (x+1)y.$

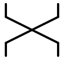
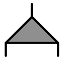
On calcule :

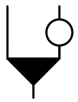
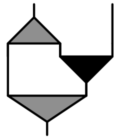
$d\Big($  $\Big)(x,y) = d\Big($  $\Big)\Big(x, F\Big($  $\Big)(y)\Big) + d\Big($  $\Big)(y)$

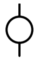
$d\Big($  $\Big)(x,y)$

Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

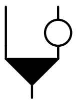
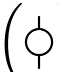
$F\Big($  $\Big)(x,y) = (y,x)$, $F\Big($  $\Big)(x) = (x,x)$, $F\Big($  $\Big)(x) = *$

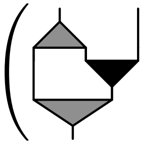
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

$F\Big($  $\Big) = 1$, $F\Big($  $\Big)(x) = X+1$, $F\Big($  $\Big)(x,y) = x+y$, $F\Big($  $\Big)(x,y) = xy$,

$d\Big($  $\Big)(x,y) = x$, $d\Big($  $\Big)(x,y) = (x+1)y$.

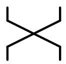
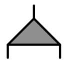
On calcule :

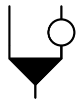
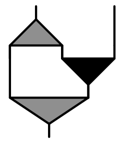
$d\Big($  $\Big)(x,y) = d\Big($  $\Big)(x,y+1) + d\Big($  $\Big)(y)$

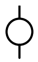
$d\Big($  $\Big)(x,y)$

Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

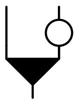
$F\Big($  $\Big)(x,y) = (y,x),\quad F\Big($  $\Big)(x) = (x,x),\quad F\Big($  $\Big)(x) = *$

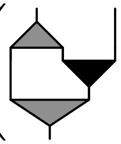
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

$F\Big($  $\Big) = 1,\quad F\Big($  $\Big)(x) = X+1,\quad F\Big($  $\Big)(x,y) = x+y,\quad F\Big($  $\Big)(x,y) = xy,$

$d\Big($  $\Big)(x,y) = x,\quad d\Big($  $\Big)(x,y) = (x+1)y.$

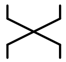
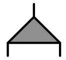
On calcule :

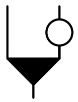
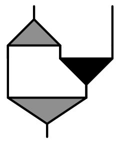
$d\Big($  $\Big)(x,y) = (x+1)(y+1)+0$

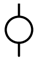
$d\Big($  $\Big)(x,y)$

Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

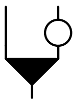
$F\Big($  $\Big)(x,y) = (y,x)$, $F\Big($  $\Big)(x) = (x,x)$, $F\Big($  $\Big)(x) = *$

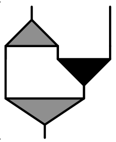
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

$F\Big($  $\Big) = 1$, $F\Big($  $\Big)(x) = X + 1$, $F\Big($  $\Big)(x,y) = x + y$, $F\Big($  $\Big)(x,y) = xy$,

$d\Big($  $\Big)(x,y) = x$, $d\Big($  $\Big)(x,y) = (x+1)y$.

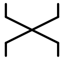
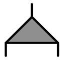
On calcule :

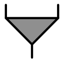
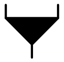
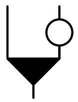
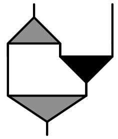
$d\Big($  $\Big)(x,y) = xy + x + y + 1$,

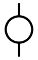
$d\Big($  $\Big)(x,y)$

Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

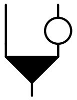
$F\Big($$\Big)(x,y) = (y,x)$, $F\Big($$\Big)(x) = (x,x)$, $F\Big($$\Big)(x) = *$

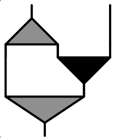
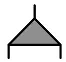
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

$F\Big($$\Big) = 1$, $F\Big($$\Big)(x) = X+1$, $F\Big($$\Big)(x,y) = x+y$, $F\Big($$\Big)(x,y) = xy$,

$d\Big($$\Big)(x,y) = x$, $d\Big($$\Big)(x,y) = (x+1)y$.

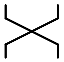
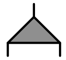
On calcule : $\left\{\right.$

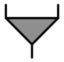
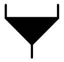
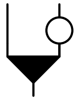
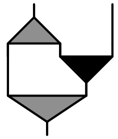
$d\Big($$\Big)(x,y) = xy+x+y+1$,

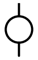
$d\Big($$\Big)(x,y) = d\Big($$\Big)\Big(x, F\Big($$\Big)(x,y)\Big) + d\Big($$\Big)(x,y) + d\Big($$\Big)(x)$

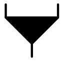
Exemple

Interprétations standard. On peut toujours prendre le 2-foncteur F suivant, avec d nulle :

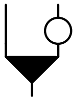
$F\Big($  $\Big)(x,y) = (y,x)$, $\quad F\Big($  $\Big)(x) = (x,x)$, $\quad F\Big($  $\Big)(x) = *$

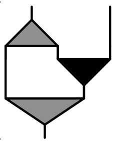
Addition  **et multiplication** **.** Pour  $\Rrightarrow$ , on considère :

$F\Big($  $\Big) = 1$, $\quad F\Big($  $\Big)(x) = X+1$, $\quad F\Big($  $\Big)(x,y) = x+y$, $\quad F\Big($  $\Big)(x,y) = xy$,

$d\Big($  $\Big)(x,y) = x$, $\quad d\Big($  $\Big)(x,y) = (x+1)y$.

On calcule :

$d\Big($  $\Big)(x,y) = xy+x+y+1$,

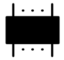
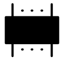
$d\Big($  $\Big)(x,y) = xy+x+y$.

Complexité des programmes polygraphiques

3. Caractérisations polygraphiques de classes de complexité

La classe PTime

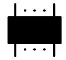
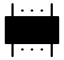
Classe **P** : programmes polygraphiques Σ convergents, complets, qui admettent :

- Un 2-foncteur $F : \Sigma_2^* \to \mathbf{Ord}$ additif et compatible.
- Une dérivation d de Σ_2^* dans F, compatible, nulle sur les constructeurs, telle que, pour toute fonction , l'application d () est bornée par un polynôme.

Alors : toute application f calculée par Σ est dans **PTime**.

La classe PTime

Classe **P** : programmes polygraphiques Σ convergents, complets, qui admettent :

- Un 2-foncteur $\mathrm{F} : \Sigma_2^* \to \mathbf{Ord}$ additif et compatible.
- Une dérivation d de Σ_2^* dans F, compatible, nulle sur les constructeurs, telle que, pour toute fonction , l'application d () est bornée par un polynôme.

Alors : toute application f calculée par Σ est dans **PTime**.

Réciproquement : toute fonction de **PTime** est calculable par un programme polygraphique de **P** (on construit une machine de Turing polygraphique à horloge).

Théorème. Les programmes polygraphiques de **P** calculent exactement les fonctions de **PTime**.

Autres classes de complexité (1)

Temps (doublement) exponentiel.

Classe **E** : comme **P**, en remplaçant la majoration polynomiale par une borne exponentielle.

Ou : comme **P**, en avec un 2-foncteur **affine** (au lieu d'additif), *i.e.* pour tout constructeur $\bigtriangledown$ d'arité n :

$$F\left(\bigtriangledown\right)(x_1,\ldots,x_n) = a_1x_1 + \cdots + a_nx_n + b.$$

Alors $||\bigtriangledown|| \leq F(\bigtriangledown) \leq a||\bigtriangledown||$ devient :

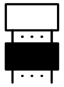
$$||\bigtriangledown|| \leq F(\bigtriangledown) \leq a^{||\bigtriangledown||}.$$

La classe **E** caractérise la classe des fonctions calculables en temps exponentiel.

Pour le temps doublement exponentiel, les $F\left(\bigtriangledown\right)$ sont des polynômes quelconques.

Autres classes de complexité (2)

Non déterminisme.

Classe **NP** : comme **P**, avec des polygraphes non confluents et un ordre total sur les formes normales des 2-cellules  (décidable en temps linéaire).

La classe **NP** caractérise la classe de complexité **NPTime**.

Autres classes de complexité (2)

Non déterminisme.

Classe **NP** : comme **P**, avec des polygraphes non confluents et un ordre total sur les formes normales des 2-cellules (décidable en temps linéaire).

La classe **NP** caractérise la classe de complexité **NPTime**.

Espace polynomial.

On remplace, dans la définition de dérivation, les relations de compatibilité avec les compositions par :

$$\mathrm{d}\left(\boxed{f}\;\boxed{g}\right) = \max\left\{ \mathrm{d}\left(\boxed{f}\right)\circ\pi_1 ,\; \mathrm{d}\left(\boxed{g}\right)\circ\pi_2 \right\}$$

et :

$$\mathrm{d}\left(\begin{array}{c}\boxed{f}\\ \boxed{g}\end{array}\right) = \max\left\{ \mathrm{d}\left(\boxed{f}\right) ,\; \mathrm{d}\left(\boxed{g}\right)\circ F\left(\boxed{f}\right) \right\} .$$

Conjecture : la classe obtenue caractérise **PSpace**.

Rencontre INVAL
Complexité des programmes polygraphiques

Guillaume Bonfante – Yves Guiraud

Lyon – 26 septembre 2008