REVUE FRANÇAISE D'AUTOMATIQUE INFORMATIQUE RECHERCHE OPÉRATIONNELLE. INFORMATIQUE THÉORIQUE

LOUIS NOLIN

Algorithmes universels

Revue française d'automatique informatique recherche opérationnelle. Informatique théorique, tome 8, nº R1 (1974), p. 5-18

<http://www.numdam.org/item?id=ITA_1974__8_1_5_0>

© AFCET, 1974, tous droits réservés.

L'accès aux archives de la revue « Revue française d'automatique informatique recherche opérationnelle. Informatique théorique » implique l'accord avec les conditions générales d'utilisation (http://www.numdam.org/conditions). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme Numérisation de documents anciens mathématiques

http://www.numdam.org/

R.A.I.R.O.
(8e année, R-1, 1974, p. 5-18)

ALGORITHMES UNIVERSELS

par Louis NOLIN (1)

communiqué par M. NIVAT

Résumé. — *On rappelle tout d'abord les grandes lignes d'une théorie des algorithmes conçue tout spécialement pour les besoins de l'Informatique. Puis sont étudiés quelques algorithmes universels qui permettent de définir les concepts fondamentaux des langages de programmation. On les emploie alors pour traduire une procédure dans la théorie, donnant ainsi sa signification, un moyen pour en obtenir des propriétés élémentaires et une indication pour la compiler ou l'interpréter. On conclut en affirmant que c'est là un simple aperçu du rôle que peut jouer cette théorie.*

Nous nous proposons de montrer comment une certaine théorie des algorithmes permet de préciser et parfois de résoudre nombre de problèmes relevant de la sémantique des procédures. Cette théorie nous occupe depuis bientôt trois ans et a fait l'objet de plusieurs publications miméographiées ainsi que d'un cours de DEA non publié. Au fil des mois, nous avons corrigé des erreurs parfois grossières et rectifié certaines définitions ou hypothèses, balançant entre l'insuffisance et la contradiction. L'amplitude de l'oscillation est maintenant faible; c'est que les chercheurs qui participent à l'ATP 7110, notamment O. Carrière, M. Duboué, P. Jammes, F. Le Berre, J. L'helgoualc'h, B. Robinet et G. Ruggiu nous ont beaucoup aidé. Et que dire des coups de pouce donnés par G. Huet, A. Lentin, M. Nivat et M. P. Schützenberger !

L'essentiel de ce mémoire est dans la *partie 2* où sont étudiés quelques algorithmes universels qui permettent de définir les concepts fondamentaux des langages de programmation, et dans la *partie 3* où ils sont mis en œuvre sur un exemple. Pour en faciliter la lecture et pour tenir compte des dernières corrections, nous résumons dans la *partie 1* l'essentiel de la théorie. En guise de conclusion, enfin, nous faisons allusion à quelques unes des applications possibles de la théorie. Si ce début plaît, peut-être reviendrons-nous là-dessus quelque jour.

(1) Groupe d'Informatique théorique, U.E.R. de Mathématiques, Université de Paris VII.

1. RAPPEL DES PRINCIPES

Soit T une collection non vide, $\mathcal{A}$ une collection de parties de T contenant $\emptyset$ et T comme éléments, close pour l'intersection infinie.

Dans tout ce qui suit, les lettres V, W, X, Y, Z, affectées au besoin d'indices, désignent des éléments de $\mathcal{A}$, sauf indication contraire explicite.

Définition 1

1. $\overline{\bigcup_i} X_i$ est le plus petit majorant des $X_i (i \in I \neq \emptyset)$.

2. $\{ X_i / i \in I \neq \emptyset \}$ est une *pré-partition* de X ssi $\bigcup_i X_i = X$ et si $i \neq j$ entraîne $X_i \nsubseteq X_j (i, j \in I)$.

3. X est *atomique* ssi $X \subseteq \bigcup_i Y_i (i \in I \neq \emptyset)$ entraîne $X \subseteq Y_j$ pour un j au moins dans I.

4. $\mathcal{F}$ est la collection des applications $f : \mathcal{A} \to \mathcal{A}$ qui sont *normales*, i.e. telles que, pour tout X, $f(X) = \overline{\bigcup} \{ f(Y) / Y \text{ atomique} \subseteq X \}$.

5. $\mathcal{F}_{X,Y}$ est la sous-collection de $\mathcal{F}$ telle que $f \in \mathcal{F}_{X,Y}$ ssi $f(X) \subseteq Y$; si $Y \neq T$, alors $FXY = \mathcal{F}_{X,Y}$, sinon $T \supseteq FXY \supseteq \mathcal{F}_{X,Y}$.

6. $\mathcal{A}_F$ est la plus petite collection qui contient tous les FXY où $Y \neq T$ et qui est close pour l'intersection infinie.

7. Si $X' \in \mathcal{A} \cup \mathcal{A}_F$, l'*image de* Y *par* X', en abrégé $X'[Y]$, est $\bigcap \{ Z / X' \subseteq FYZ \}$.

Théorème 1. Si $Z_j (j \in J \neq \emptyset)$ est atomique (et si $\bigcup_j Z_j \in \mathcal{A}$), alors

$$\left(\bigcap_{i \in I \neq \emptyset} FX_i Y_i \right) \left[\bigcup_j Z_j \right] = \overline{\bigcup_{j \in J}} \left(\bigcap_{i \in I} \{ Y_i / Z_j \subseteq X_i \} \right).$$

La démonstration de ce résultat s'appuie sur quelques lemmes que nous emploierons plus loin à d'autres fins :

(L 1). $FXY \subseteq FX_1 Y_1$ ssi ou bien $Y_1 = T$, ou bien $X_1 \subseteq X$ et $Y \subseteq Y_1$.

(L 2). $\bigcap_{i \in I \neq \emptyset} FXY_i = FX \left(\bigcap_i Y_i \right)$; $\left(\text{si } \bigcup_{i \in I \neq \emptyset} X_i \in \mathcal{A} \right)$,

alors $\bigcap_{i \in I} FX_i Y = F \left(\bigcup_i X_i \right) Y$.

(L 3). $X[Y] \subseteq Z$ ssi $X \subseteq FYZ$; d'où un cas particulier du théorème 1, à savoir : $FXY[Z] =$ (si $Z \subseteq X$ alors Y, sinon T).

(L 4). Si $X', X'_1 \in \mathcal{A} \cup \mathcal{A}_f$ et si $X' \subseteq X'_1$ et $Y \subseteq Y_1$, alors $X'[Y] \subseteq X'_1[Y_1]$; d'où la monotonicité des éléments de $\mathcal{A} \cup \mathcal{A}_F$ et aussi la conséquence : $\left(\overline{\bigcup_{i \in I \neq \emptyset}} X'\right)[Y] = \overline{\bigcup_{i \in I}} (X'[Y])$.

(L 5). Si $X' \in \mathcal{A} \cup \mathcal{A}_F \left(\text{et si } \bigcup_{j \in J \neq \emptyset} Y_j \in \mathcal{A}\right)$, alors $X'\left[\bigcup_j Y_j\right] = \overline{\bigcup_j} (X'[Y_j])$.

(L 6). Un cas particulier du théorème 1, compte tenu de (L 3) : $\bigcap_{i \in I \neq \emptyset} FX_iY_i \subseteq F\left(\bigcap_i X_i\right)\left(\bigcap_i Y_i\right)$.

(L 7). Un cas particulier du théorème 1, à savoir : si Z est atomique, alors $\left(\bigcap_{i \in I \neq \emptyset} FX_iY_i\right)[Z] = \bigcap_i \{ Y_i/Z \subseteq X_i \}$.

Remarquant alors que si $\mathcal{A} = \mathcal{P}(T)$ est infini, $\mathcal{A}$ et $\mathcal{A}_F$ ont même cardinalité, nous pensons qu'il n'est pas déraisonnable de plonger $\mathcal{A}_F$ dans $\mathcal{A}$; ajoutons à cela quelques hypothèses servant à éliminer des situations apparemment sans intérêt, et nous sommes conduits à donner la définition suivante :

Définition 2. Soit T une collection non vide; une collection $\mathcal{A}$ de parties de T est appelée une *collection d'algorithmes* ssi elle remplit les conditions suivantes :

1. il existe une collection $\mathcal{B}$ de parties de T qui contient $\emptyset$ et T comme éléments, telle que $\mathcal{A}$ est la plus petite sous-collection de $\mathcal{P}(T)$ qui contient $\mathcal{B}$ et qui est close pour l'intersection infinie et pour l'opération $X, Y \longmapsto FXY$, avec $FXT = T$ pour tout X ;

2. soit $\mathcal{A}_B$ la fermeture de $\mathcal{B}$ par l'intersection infinie, amputée de $\emptyset$ et de T; alors, si $X \in \mathcal{A}_B$ et si $Y \in \mathcal{A}_F$, X et Y sont incomparables (ni $X \subseteq Y$, ni $Y \subseteq X$) ;

3. $\bigcup \{ Z/Z \text{ atomique } \subseteq X \} = X$;

4. T est atomique.

Outre le théorème 1 et les lemmes 1 à 7, on tire de cette définition les conséquences suivantes :

(L 8). $\{ \{ \emptyset \}, \mathcal{A}_B \text{ (si } \neq \emptyset), \mathcal{A}_F, \{ T \} \}$ est une partition de $\mathcal{A}$; les éléments de $\mathcal{A}_B$ sont les *ensembles de base*, ceux de $\mathcal{A}_F$ les *algorithmes propres*.

(L 9). $\emptyset[Y] = \emptyset$ pour tout Y ; $X[Y] = T$ pour tout Y ssi $X \in \mathcal{A}_B \cup \{ T \}$; ($X \subseteq Y$ ssi $\forall\, Z/X[Z] \subseteq Y[Z]$) ssi $X, Y \in \mathcal{A}_F \cup \{ T \}$; (on peut même réduire la quantification aux Z qui sont atomiques).

(L 10). $\overline{\bigcup_{i \in I \neq \emptyset}} FX_iY_i = F\left(\bigcap_i X_i\right)\left(\overline{\bigcup_i} Y_i\right)$.

Théorème 2. Pour tout algorithme propre X, existe une *base* unique, $\mathcal{A} = \{ FY_iZ_i / i \in I \neq \emptyset \}$ telle que $X = \cap \mathcal{A}$ et, pour tous i, j dans I : Y_i est atomique; si Y atomique $\subseteq Y_i$ alors existe $k \in I$ tel que $Y = Y_k$; $i \neq j$ entraîne $Y_i \neq Y_j$; $Y_i \subseteq Y_j$ entraîne $Z_i \subseteq Z_j$; $Z_i \neq T$. Ainsi, lorsque $V_j (j \in J \neq \emptyset)$ est atomique $\left(\text{et que } \bigcup_j V_j \in \mathcal{A}\right)$, on a

$$\left(\bigcap_{i \in I} FY_iZ_i\right)\left[\bigcup_j V_j\right] = \bar{\bigcup_j} \{ Z_i / V_j = Y_i \},$$

ou encore : $\left(\bigcap_{i \in I} Y_iZ_i\right)[V] = \bar{\cup} \{ Z_i / Y_i \subseteq V \} = \bar{\cup} \{ X[Y] / Y \text{ atomique} \subseteq V \}$.

C'est ce résultat que nous emploierons le plus souvent, sous l'une ou l'autre de ses formes pour calculer l'image $X[V]$.

Théorème 3. $\emptyset$, T, les singletons ($\in \mathcal{A}_B$) s'il y en a, les algorithmes propres sont tous atomiques.

Ainsi tout algorithme non atomique ne peut être qu'un ensemble de base; et un algorithme atomique n'est pas forcément un singleton, à preuve FXY en général.

Nous nous excusons de passer sous silence les démonstrations de ces résultats.

Pour alléger l'écriture, nous adoptons la convention suivante :

Définition 3. $F_1X_1Y = FX_1Y$; pour tout $n \in \mathbf{N}^+$:

$$F_{n+1}X_1 \dots X_{n+1}Y = F_nX_1 \dots X_n(FX_{n+1}Y)$$

et

$$X[Y_1, \dots, Y_{n+1}] = (X[Y_1, \dots, Y_n])[Y_{n+1}].$$

Ainsi, par exemple,

$$F_3X_1X_2X_3Y = F_2X_1X_2(FX_3Y) = FX_1(FX_2(FX_3Y)) = FX_1(F_2X_2X_3Y),$$

$$X[Y_1, Y_2, Y_3] = (X[Y_1, Y_2])[Y_3] = ((X[Y_1])[Y_2])[Y_3] = (X[Y_1])[Y_2, Y_3].$$

2. QUELQUES ALGORITHMES UNIVERSELS

Nous nous contentons ici de définir et d'étudier les algorithmes universels qui nous serviront dans la *partie 3* à représenter une procédure dans notre théorie; il s'agit des algorithmes qui permettent de traduire ce qu'expriment, dans les langages de programmation, le point-virgule, l'instruction vide, les tests, les boucles et les déclarations.

Le point-virgule

Définition 4. $Puis = \bigcap \{ F_3 V_1 V_2 V_3 (V_2[V_1[V_3]]) / V_1, V_2, V_3 \in \mathcal{A} \}$.

Proposition 4. Si $Z_i (i \in I \neq \emptyset)$ est atomique $\left(\text{et si } \bigcup_i Z_i \in \mathcal{A}\right)$ alors $Puis \left[X, Y, \bigcup_i Z_i\right] = \overline{\bigcup_i} (Y[X[Z_i]])$.

En effet, nous pouvons supposer que les V_1, V_2, V_3 qui figurent dans la définition de *Puis* sont atomiques (Th. 2). Alors

$$Puis\,[X] = \bigcap_{V_2, V_3} F_2 V_2 V_3 (V_2[X[V_3]]):$$

c'est vrai si X est atomique (Th. 2) et si $X = \bigcup_j X_j$ — X_j atomique — alors $Puis\,[X] = \overline{\bigcup_j} \left(\bigcap_{V_2, V_3} F_2 V_2 V_3 (V_2[X_j[V_3]]) \right)$ (Th. 2); mais les X_j et X sont alors dans $\mathcal{A}_B$ (Th. 3) et $V_2[X_j[V_3]] = V_2[T] = V_2[X[V_3]]$ (L 9), d'où le même résultat que si X est atomique. On démontre de la même façon que

$$Puis\,[X, Y] = (Puis\,[X])[Y] = \bigcap_{V_3} F V_3 (Y[X[V_3]]).$$

Mais la simplification tirée de (L 9) ne vaut plus dans la troisième étape et on a simplement (Th. 2) le résultat énoncé plus haut.

L'algorithme *Puis* n'est associatif que dans les bons cas.

Définition 5. L'algorithme X est simple ssi, pour tout Y atomique, $X[Y]$ est atomique.

Proposition 5. $Puis\,[V_1, Puis\,[V_2, V_3]] \subseteq Puis\,[Puis\,[V_1, V_2], V_3]$; si V_1 est simple, ces deux algorithmes sont égaux.

Puisque ces algorithmes $\in \mathcal{A}_F \cup \{ T \}$, il suffit de montrer (L 9) que l'image donnée par le premier d'un X atomique est incluse dans celle qu'en donne le second. Soit donc X atomique; $(Puis\,[V_1, Puis\,[V_2, V_3]])[X] = Puis\,[V_1, Puis\,[V_2, V_3], X] = (Puis\,[V_2, V_3])[V_1[X]]$ (Pp. 4) $= Puis\,[V_2, V_3, V_1[X]]$; soit $V_1[X] = \bigcup_{i \in I \neq \emptyset} Y_i$ (Y_i atomique); on a alors

$$Puis \left[V_2, V_3, \bigcup_i Y_i\right] = \overline{\bigcup_i} V_3[V_2[Y_i]]$$

(Th. 2); en vertu de (L 4), cet algorithme est $\subseteq^*$ $V_3 \left[V_2 \left[\bigcup_i Yi \right] \right] = V_3[V_2\,[V_1[X]] = V_3[Puis\,[V_1, V_2, X]]$ (Pp. 4) $= V_3[(Puis\,[V_1, V_2])[X]] = Puis\,[Puis\,[V_1, V_2], V_3, X]$ (Pp. 4) $= (Puis\,[Puis\,[V_1, V_2], V_3])[X]$. Et si V_1 est simple, $V_1[X]$ est atomique de sorte que l'inclusion $\subseteq^*$ devient l'identité.

L'instruction vide

Définition 6. $\underline{I} = \bigcap_X FXX.$

Proposition 6. $\underline{I}[X] = X$; *Puis* $[\underline{I}, Z] =$ *Puis* $[Z, \underline{I}] = Z$ ssi $Z \in \mathcal{A}_F \cup \{T\}$.

En effet, $\underline{I} = \cap \{FXX/X$ atomique $\}$ (Th. 2), d'où, si X_i est atomique $\left(\text{et si } \bigcup_i X_i \in \mathcal{A}\right)$, $\underline{I}\left[\bigcup_i X_i\right] = \overline{\bigcup_i} X_i$ (Th. 2) $= \bigcup_i X_i$. En examinant la démonstration de la proposition 4 on voit que

$$Puis\ [\underline{I}, Z] = \bigcap_{V_3} FV_3(Z[\underline{I}[V_3]]) = \bigcap_{V_3} FV_3(Z[V_3]),$$

en vertu du résultat précédent; et, de même, que

$$Puis\ [Z, \underline{I}] = \bigcap_{V_3} FV_3(\underline{I}[Z[V_3]]) = \bigcap_{V_3} FV_3(Z[V_3]).$$

Soit W cet algorithme; les seuls cas possibles sont les suivants (L 8) : ou bien $Z = \varnothing$ et $W = \bigcap_{V_3} FV_3\,\varnothing$ (L 9) $= FT\,\varnothing$ (L 1); comme $FT\varnothing = \{f/f(V_3) = \varnothing$ pour tout $V_3 \in \mathcal{A}\}$ (Df. 1), on a alors $W \neq Z$; ou bien $Z \in \mathcal{A}_B$ et $W = \bigcap_{V_3} FV_3T$ (L 9); ici encore $W \neq Z$ (Df. 2) ; ou bien

$$Z = T \text{ et } W = \bigcap_{V_3} FV_3T \text{ (L 9)} = T \text{ (Df 2)};$$

ou enfin $Z \in \mathcal{A}_F$ et, pour tout X atomique, $W[X] = Z[X]$ (Th. 2), d'où $W = Z$ (L 9). La proposition est démontrée.

Ainsi, dans les bons cas encore, $\underline{I}$ est élément neutre de *Puis* à gauche et à droite.

Les tests

Définition 7.

$$P_2^1 = \bigcap_{Y,Z} F_2YZY;\ P_2^2 = \bigcap_{Y,Z} F_2YZZ;\ \underline{\cap} = \bigcap_{Y,Z} F_2YZ(Y \cap Z);$$

$$\underline{\cup} = \bigcap_{Y,Z} F_2YZ(Y\,\bar{\cup}\,Z).$$

Proposition 7.

$$P_2^1[V_1] = FTV_1,\ P_2^2[V_1] = \underline{I},\ P_2^1[V_1, V_2] = V_1,\ P_2^2[V_1, V_2] = V_2;$$

$$\underline{\cap} = P_2^1 \cap P_2^2,\ \underline{\cap}\,[V_1] = \bigcap_Z FZ(V_1 \cap Z),\ \underline{\cap}\,[V_1, V_2] = V_1 \cap V_2;$$

$$\underline{\cup} = P_2^1\,\bar{\cup}\,P_2^2,\ \bar{\cup}\,[V_1, V_2] = V_1\,\bar{\cup}\,V_2.$$

En effet :

1. Si V_i est atomique $\left(\text{et si } \bigcup_i V_i \in \mathcal{A}\right)$, alors

$$P_2^1\left[\bigcup_i V_i\right] = \bar{\bigcup_i}\left(\bigcap_Z FZV_i\right)(\text{Th. 2}) = \bar{\bigcup_i} FTV_i\ (\text{L 1}) = FT\left(\bigcup_i V_i\right)\ (\text{L 10})$$

donc $P_2^1[V_1, V_2] = (FTV_1)[V_2] = V_1$ (L 3).

2. Si V_i est atomique, alors

$$P_2^2\left[\bigcup_i V_i\right] = \bar{\bigcup_i}\left(\bigcap_Z FZZ\right)(\text{Th. 2}) = \bigcap_Z FZZ = \underline{I}(\text{Df. 6});$$

d'où $P_2^2[V_1, V_2] = \underline{I}[V_2] = V_2$ (Pp. 6).

3. $$P_2^1 \cap P_2^2 = \left(\bigcap_{Y,Z} F_2YZY\right) \cap \left(\bigcap_{Y,Z} F_2YZZ\right)$$
$$= \bigcap_{Y,Z} (F_2YZY \cap F_2YZZ) = \bigcap_{Y,Z} (F_2YZ(Y \cap Z))\ (\text{L 2}) = \underline{\cap};$$

soit alors Y_i atomique; $\underline{\cap}\left[\bigcup_i Y_i\right] = \bar{\bigcup_i}\left(\bigcap_Z FZ(Y_i \cap Z)\right)$ (Th. 2); si V_2 est atomique, alors

$$\underline{\cap}\left[\bigcup_i Y_i, V_2\right] = \left(\bar{\bigcup_i}\left(\bigcap_Z FZ(Y_i \cap Z)\right)\right)[V_2]$$
$$= \bar{\bigcup_i}\left(\left(\bigcap_Z FZ(Y_i \cap Z)\right)[V_2]\right)\ (\text{L 4}) = \bar{\bigcup_i}(Y_i \cap V_2)\ (\text{Th. 2})$$
$$= \left(\bigcup_i Y_i\right) \cap V_2 = \left(\bigcap_Z FZ\left(\left(\bigcup_i Y_i\right) \cap Z\right)\right)[V_2]\ (\text{Th. 2});$$

donc $\underline{\cap}[V_1] = \bigcap_Z FZ(V_1 \cap Z)$ (L 9). Et du même coup, on a montré que $\underline{\cap}[V_1, V_2] = V_1 \cap V_2$ si V_2 est atomique ; ainsi, si Z_j est atomique,

$$\underline{\cap}\left[V_1, \bigcup_j Z_j\right] = \bar{\bigcup_j}(V_1 \cap Z_j)\ (\text{Th. 2}) = V_1 \cap \left(\bigcup_j Z_j\right);$$

donc, pour tous V_1 et V_2, $\underline{\cap}[V_1, V_2] = V_1 \cap V_2$.

4. Pour tous V_1, V_2,

$$(P_2^1 \bar{\cup} P_2^2)[V_1, V_2] = P_2^1[V_1, V_2] \bar{\cup} P_2^2[V_1, V_2]\ (\text{L 4}) = V_1 \bar{\cup} V_2$$

(résultats 1 et 2 ci-dessus) ; d'autre part, si V_1 et V_2 sont atomiques,

$$\underline{\cup}[V_1, V_2] = V_1 \bar{\cup} V_2\ (\text{Th. 2});\ \text{ainsi (L 9)},\ \underline{\cup} = P_2^1 \bar{\cup} P_2^2.$$

Définition 8. Pour tous $X_1, X_2, \Delta_{X_1,X_2} = FX_1P_2^1 \cap FX_2P_2^2$.

Proposition 8. $\Delta_{X_1,X_2}[V] =$

(si $V \subseteq X_1 \cap X_2$ alors $\underline{\cap}$, sinon si $V \subseteq X_1$ alors P_2^1,, sinon si $V \subseteq X_2$ alors P_2^2, sinon si $\{ V \cap X_1, V \cap X_2 \}$ est une pré-partition de V alors $\underline{\cup}$, sinon T).

Il suffit en effet d'appliquer le théorème 2 dans chacun des cinq cas suivants : les Z atomiques $\subseteq V$ sont dans X_1 et dans X_2, ou dans X_1 seul, ou dans X_2 seul, ou partie dans X_1 et partie dans X_2, ou enfin l'un d'eux, Z_i n'est ni dans X_1, ni dans X_2 ; dans ce cas, comme $\Delta_{X_1,X_2} = \Delta_{X_1,X_2} \cap FZ_iT$, l'image que cet algorithme fournit de V est T.

Corollaire 8

$$\Delta_{X_1,X_2}[V, Y, Z] = (FX_1Y \cap FX_2Z)[V] ;$$

si $X_1 \cap X_2 = \emptyset$ et si $V \neq \emptyset$ est atomique, alors $\Delta_{X_1,X_2}[V, Y, Z] =$ (si $V \subseteq X_1$ alors Y, sinon si $V \subseteq X_2$ alors Z, sinon T).

Ces résultats découlent des propositions 7 et 8.

Corollaire 8 *bis*

Si V est atomique, alors :

$$\Delta_{X_1,X_2}\left[V, \bigcap_{i\in I} Y_i, \bigcap_{i\in I} Z_i\right] = \bigcap_{i\in I} \Delta_{X_1,X_2}[V, Y_i, Z_i];$$

si $V \nsubseteq X_1 \cap X_2$, alors :

$$\overline{\bigcup_{i\in I}} \Delta_{X_1,X_2}[V, Y_i, Z_i] = \Delta_{X_1,X_2}\left[V, \overline{\bigcup_{i\in I}} Y_i, \overline{\bigcup_{i\in I}} Z_i\right];$$

si $V \nsubseteq X_1 \cap X_2$ et si (V est atomique ou $Y \cup Z \in \mathcal{A}$) alors :

$$X[\Delta_{X_1,X_2}[V, Y, Z]] = \Delta_{X_1,X_2}[V, X[Y], X[Z]] ;$$

si $V \nsubseteq X_1 \cap X_2$ ou si (W est atomique et Y, Z sont des algorithmes propres) alors :

$$(\Delta_{X_1,X_2}[V, Y, Z])[W] = \Delta_{X_1,X_2}[V, Y[W], Z[W]] ;$$

chacune de ces égalités est valide si $X_1 \cap X_2 = \emptyset$ et si $V \neq \emptyset$ est atomique ; elle doit être remplacée par une inclusion ($\subseteq$) dans la plupart des cas où la condition qui la concerne n'est pas remplie.

Corollaire 8 *ter*

Si $V \nsubseteq X_1 \cap X_2$ ou si (Y et Z sont des algorithmes propres et W un algorithme simple) alors :

$$Puis\,[W, \Delta_{X_1,X_2}[V, Y, Z]] = \Delta_{X_1,X_2}[V, Puis\,[W, Y], Puis\,[W, Z]] ;$$

si, pour tout X atomique $V[X] \nsubseteq X_1 \cap X_2$ et ($V[X]$ est atomique ou $Y[X] \cup Z[X] \in \mathcal{A}$), alors :

$$Puis\ [\Delta_{X_1,X_2}[V, Y, Z], W] = \Delta_{X_1,X_2}[V, Puis\ [Y, W], Puis\ [Z, W]].$$

Les démonstrations de ces corollaires ne présentent pas de difficulté.

Les boucles

Définition 9. Y est un point fixe de X ssi $X[Y] = Y$.

Proposition 9. Tout algorithme X a un point fixe minoré par $\overline{\bigcup_{n \in \mathbf{N}}} X^n[\varnothing]$ et majoré par $\bigcap_{n \in \mathbf{N}} X^n[T]$.

Les propriétés des algorithmes sont suffisamment proches de celles des fonctions pour qu'on puisse démontrer l'existence d'un point fixe en adaptant simplement aux circonstances la démonstration de A. Tarski. La construction du minorant et du majorant s'inspirent de la méthode de L. Kalmár.

Corollaire 9. Si, pour un entier m, $X^m[T] \subseteq X^{m+1}[T]$, alors $\bigcap_{n \in \mathbf{N}} X^n[T] = X^m[T]$ est le plus grand point fixe de X.

En effet, pour tout $n \in \mathbf{N}$, $X^n[T] \supseteq X^{n+1}[T]$: la propriété est vraie pour $n = 0$ et se transmet de m à $m + 1$ (L 4) ; si donc $X^m[T] \subseteq X^{m+1}[T]$, alors $X^m[T] = X^{m+1}[T] = X[X^m[T]]$.

Définition 10. Pour tous X_1, X_2 :

$$R_{X_1,X_2} = \bigcap_{V_0,V_1,V_2,X} F_2 V_0 V_1 V_2 X\,(\Delta_{X_1,X_2}[V_1[X], V_0[V_1, V_2, V_2[X]], X]);$$

$$Tq_{X_1,X_2} = \bigcap_{n \in \mathbf{N}} ((R_{X_1,X_2})^n[T]).$$

Proposition 10. Si X est atomique et si Y et Z sont des algorithmes propres simples tels que $Y[X] \nsubseteq X_1 \cap X_2$, alors $Tq_{X_1,X_2}[Y, Z, X] =$

$$= \Delta_{X_1,X_2}[Y[X], Tq_{X_1,X_2}[Y, Z, Z[X]], X].$$

En effet, puisque T, Y, Z, X, $Z[X]$, $Y[X]$ sont atomiques, on a : $R_{X_1,X_2}[T, Y, Z, X] = \Delta_{X_1,X_2}[Y[X], T[T, Y, Z, Z[X]], X] = \Delta_{X_1,X_2}[Y[X], T, T]$ (L 9) qui est atomique, étant soit T, soit X (Cor. 8) ; pour la même raison, $R_{X_1,X_2}[T, Y, Z, Z[X]]$ est atomique ; s'il en est de même pour $(R_{X_1,X_2})^n[T, Y, Z, X]$, alors

$$(R_{X_1,X_2})^{n+1}[T, Y, Z, X] = \Delta_{X_1,X_2}[Y[X], (R_{X_1,X_2})^n[T, Y, Z, Z[X]], X],$$

lui aussi atomique. Ainsi

$$Tq_{X_1,X_2}[Y, Z, X] = \bigcap_{n\in\mathbf{N}} (R_{X_1,X_2})^n[T, Y, Z, X]$$

$$= \bigcap_{n\in\mathbf{N}} \Delta_{X_1,X_2}[Y[X], (R_{X_1,X_2})^n[Y, Z, Z[X]], X]$$

$$= \Delta_{X_1,X_2}[Y[X], \bigcap_{n\in\mathbf{N}} ((R_{X_1,X_2})^n[Y, Z, Z[X]]), X] \text{ (Cor. 8 } bis\text{)}$$

$$= \Delta_{X_1,X_2}[Y[X], Tq_{X_1,X_2}[Y, Z, Z[X]], X].$$

Corollaire 10. Dans les mêmes conditions, si $Y[Z^n[X]] \subseteq X_1$ pour tout $n < p$ et si $Y[Z^p[X]] \subseteq X_2$, alors $Tq_{X_1,X_2}[Y, Z, X] = Z^p[X]$.

Évident.

Les déclarations

Définition 11. $|_Y = \underline{\cap}\,[Y]$.

Proposition 11.

$$|_Y = \bigcap_Z FZ(Y \cap Z),\ |_Y[X] = Y \cap X;\ X \subseteq Y \text{ ssi } X = |_Y[X], \text{ ssi } X \subseteq |_Y[X].$$

Évident (Pp. 7).

Définition 12. $|_{X,Y} = \bigcap_{V,Z} F_2VZ((FX\underline{I})[Z, Y \cap V[Z]])$.

Proposition 12.

$$|_{X,Y}[V] = \bigcap_{Z\subseteq X} FZ(Y \cap V[Z]) = \bigcap_{Z\subseteq X} FZ(|_Y[V[Z]]) = FXY \cap \left(\bigcap_{Z\subseteq X} FZ(V[Z])\right);$$

$$V \subseteq FXY \text{ ssi } |_{X,Y}[V] = \bigcap_{Z\subseteq X} FZ(V[Z]), \text{ ssi } V \subseteq |_{X,Y}[V].$$

On démontre la première égalité par la méthode employée pour la proposition 4 ; la seconde est évidente (Pp. 11) et la troisième résulte de (L 2) et de (L 1). Démontrons la première équivalence. Soit $V \subseteq FXY$; alors, si $Z \subseteq X$, on a $V[Z] \subseteq V[X] \subseteq Y$ (L 4 et L 3), donc $|_{X,Y}[V] = \bigcap_{Z\subseteq X} FZ(V[Z])$ (1re égalité) ; réciproquement, soit $|_{X,Y}[V] = \bigcap_{Z\subseteq X} FZ(V[Z])$; alors $\bigcap_{Z\subseteq X} FZ(V[Z]) \subseteq FXY$ (3e égalité), donc $\left(\bigcap_{Z\subseteq X} FZ(V[Z])\right)[X] \subseteq Y$ (L 3); on peut supposer que les Z en question sont atomiques (Th. 2) ; alors (Th. 2) $(\cap\ \{ FZ(V[Z])/Z \text{ atomique} \subseteq X \})[X] = \bar{\cup}\ \{ V[Z]/Z \text{ atomique} \subseteq X \} = V[X]$; ainsi $V[X] \subseteq Y$ donc $V \subseteq FXY$ (L 3).

D'autre part, on a $V \subseteq \bigcap_{Z \subseteq X} FZ(V[Z])$, quel que soit $Z \in \{ \varnothing \}$, $\mathcal{A}_B$, $\mathcal{A}_F$ ou $\{ T \}$ (L 9 et Pp. 6) ; donc

$$V \subseteq FXY \text{ ssi } V \subseteq FXY \cap \left(\bigcap_{Z \subseteq X} FZ(V[Z]) \right), \text{ ssi } V \subseteq |_{X,Y}[Z]$$

(3e égalité). Ceci démontre la seconde équivalence.

Corollaire 12. Si $\{ FV_i W_i / i \in I \neq \varnothing \}$ est la base de Z, alors $\{ FV_i(Y \cap Wi)/i \in I$ et $V_i \subseteq X \}$ est la base de $|_{X,Y}[Z]$; $|_{X,T}[Z]$ est la restriction de Z à X ; $|_{X,Y}[Z] = FXY \cap |_{X,T}[Z]$; $|_Y = |_{T,Y}[I]$.

Facile.

3. UNE APPLICATION

Supposons que les ensembles de base sont N et ses parties récursivement énumérables (à l'exception de la partie vide). Seuls parmi eux sont atomiques les singletons. Posons

$$suc = \bigcap_{i \in \mathrm{N}} F\{ i \}\{ i+1 \}, \qquad carré = \bigcap_{i \in \mathrm{N}} F\{ i \}\{ i^2 \},$$

$$+ = \bigcap_{i,j \in \mathrm{N}} F_2\{ i \}\{ j \}\{ i+j \},$$

$$\leqslant = \left(\bigcap_{\substack{i,j \in \mathrm{N} \\ i \leqslant j}} F_2\{ i \}\{ j \}\{ \text{VRAI} \} \right) \cap \left(\bigcap_{\substack{i,j \in \mathrm{N} \\ i > j}} F_2\{ i \}\{ j \}\{ \text{FAUX} \} \right).$$

Pour tout $n \in \mathrm{N}$ et tous $X_0, ..., X_n \in \mathcal{A}$, posons aussi

$$\langle X_0, ..., X_n \rangle = \bigcap_{i=0}^{n} F\{ i \} X_i;$$

c'est là une bonne définition d'un $n+1$-uple d'éléments de $\mathcal{A}$ car $\langle X_0, ..., X_n \rangle [\{ i \}] =$ (si $i \leqslant n$ alors X_i, sinon T) ; or $\langle X_0, ..., X_n \rangle$ et $\langle Y_0, ..., Y_m \rangle$ sont égaux ssi les images qu'ils donnent de tout Z atomiques sont égales (L 9), donc ssi $m = n$ et, pour tout $i \leqslant n$, $X_i = Y_i$.

Nous allons maintenant traduire dans notre théorie la procédure suivante, destinée à calculer $\sum_{Y=0}^{X} W(Y)$:

0. ENTIER NATUREL PROCEDURE $S(W, X)$;
   PROCEDURE W telle que $W(\mathrm{N}) \subseteq \mathrm{N}$;
   ENTIER NATUREL X ;
   DEBUT
1. ENTIER NATUREL Y, Z ;

2. $Y := 0$; $Z := 0$;
3. TANT QUE $Y \leqslant X$ FAIRE
4. DEBUT $Z := Z + W(Y)$; $Y := Y + 1$ FIN ;
5. RESULTAT : Z

FIN.

0. L'algorithme ainsi défini est

$$S = \bigcap_{W,X} F_2 WX(S'[\langle W, X \rangle]),$$

avec

$$S' = |_{\langle FNN,N\rangle,N}[V],$$

où V est l'algorithme défini par le corps de la procédure.

1. La déclaration crée l'algorithme

$$V_1 = \bigcap_V FV(< V[\{0\}], V[\{1\}], \mathbf{N}, \mathbf{N} >),$$

de sorte que

$$V_1[\langle W, X \rangle] = \langle W, X, \mathbf{N}, \mathbf{N} \rangle;$$

elle enjoint en outre d'appliquer $|_{\mathbf{N}}$ à toute nouvelle précision sur Y ou Z apportée par une instruction d'affectation.

2. $$V_2 = \bigcap_V FV(\langle V[\{0\}], V[\{1\}], \{0\} \cap \mathbf{N}, \{0\} \cap \mathbf{N} \rangle);$$

ainsi :

$$V_2[\langle W, X, Y, Z \rangle] = \langle W, X, \{0\}, \{0\} \rangle.$$

3. $$V_3 = Tq_{\{\mathrm{VRAI}\},\{\mathrm{FAUX}\}}[V_{3bis}, V_4],$$

avec

$$V_{3bis} = \bigcap_V FV(\leqslant [V[\{2\}], V[\{1\}]]) :$$

$$V_{3bis}[\langle W, X, Y, Z \rangle] = \leqslant [Y, X]$$

4. $$V_4 = \bigcap_V FV(\langle V[\{0\}], V[\{1\}], suc\,[V[\{2\}]] \cap \mathbf{N}, + [V[\{3\}], (V[\{0\}])[V[\{2\}]]] \cap \mathbf{N} \rangle);$$

ainsi

$$V_4[\langle W, X, Y, Z \rangle] = \langle W, X, suc\,[Y] \cap \mathbf{N}, + [Z, W[Y]] \cap \mathbf{N} \rangle.$$

5. $$V_5 = \bigcap_V FV(V[\{3\}]),$$

de sorte que

$$V_5[\langle W, X, Y, Z \rangle] = Z.$$

L'algorithme défini par le corps de la procédure est donc :

$$V = Puis\,[Puis\,[Puis\,[V_1,\ V_2,]\ V_3],\ V_5],$$

et ainsi :

$$S' = |_{\langle FNN,N\rangle,N}[V],$$

$$S = \bigcap_{W,X} F_2WX(S'[\langle W, X\rangle]).$$

L'appel de procédure S(carré, 5) est donc traduit par l'algorithme :

$S\,[carré,\ \{5\}] = S'[\langle carré,\ \{5\}\rangle]$
$= |_{\langle FNN,N\rangle,N}[V, \langle carré,\ \{5\}\rangle]$
$=$ si $\langle carré,\ \{5\}\rangle \subseteq \langle FNN, N\rangle$ alors $N \cap V[\langle carré,\ \{5\}\rangle]$, sinon T,
$=$ si $\{5\} \subseteq N$ et si $carré \subseteq FNN$ alors $N \cap V[\langle carré,\ \{5\}\rangle]$, sinon T,
$= N \cap V[\langle carré,\ \{5\}\rangle]$

$= N \cap (Puis\,[Puis\,[Puis\,[V_1,\ V_2],\ V_3],\ V_5])[\langle carré,\ \{5\}\rangle]$
$= N \cap Puis\,[Puis\,[Puis\,[V_1,\ V_2],\ V_3],\ V_5,\ \langle carré,\ \{5\}\rangle]$
$= N \cap V_5[(Puis\,[Puis\,[V_1,\ V_2],\ V_3])[\langle carré,\ \{5\}\rangle]]$
$= N \cap V_5[Puis\,[Puis\,[V_1,\ V_2],\ V_3,\ \langle carré,\ \{5\}\rangle]]$
$=$ — — — — —
$= N \cap V_5[V_3[V_2[V_1[\langle carré,\ \{5\}\rangle]]]]$
$= N \cap V_5[V_3[V_2[\langle carré,\ \{5\},\ N, N\rangle]]]$
$= N \cap V_5[V_3[\langle carré,\ \{5\},\ \{0\},\ \{0\}\rangle]]$
$= N \cap V_5[Tq_{\{VRAI\},\{FAUX\}}[V_{3bis},\ V_4,\ \langle carré,\ \{5\},\ \{0\},\ \{0\}\rangle]]$
$= N \cap V_5[Tq_{\{VRAI\},\{FAUX\}}[V_{3bis},\ V_4,\ \langle carré,\ \{5\},\ \{1\},\ \{0\}\rangle]]$
$= N \cap V_5[Tq_{\{VRAI\},\{FAUX\}}[V_{3bis},\ V_4,\ \langle carré,\ \{5\},\ \{2\},\ \{1\}\rangle]]$
— — — — — —
$= N \cap V_5[Tq_{\{VRAI\},\{FAUX\}}[V_{3bis},\ V_4,\ \langle carré,\ \{5\},\ \{5\},\ \{30\}\rangle]]$
$= N \cap V_5[Tq_{\{VRAI\},\{FAUX\}}[V_{3bis},\ V_4,\ \langle carré,\ \{5\},\ \{6\},\ \{55\}\rangle]]$
$= N \cap V_5[\langle carré,\ \{5\},\ \{6\},\ \{55\}\rangle]$
$= N \cap \{55\}$
$= \{55\}$.

La succession de ces formes diverses de l'algorithme $S[carré,\ \{5\}]$ reflète fidèlement le déroulement du programme exprimant l'appel de procédure $S(carré, 5)$, et en particulier, les états successifs de la mémoire qu'il utilise.

C'est pour ces raisons que nous avons choisi un exemple particulièrement simple quant au nombre des algorithmes élémentaires qu'il requiert (V_1 à V_5), au nombre de leurs applications (six pour V_3 et V_{3bis}, cinq pour V_4, une seule pour chacun des autres), à la nature des données enfin ($\{5\}$ et *carré*

sont atomiques et *carré* est simple). Mais le lecteur attentif ne se laissera pas abuser par cette simplicité voulue, voyant bien que ce modeste exemple pose nombre de questions d'actualité. Pour le lecteur pressé citons-en quelques-unes.

Point de GOTO dans tout cela, point d'étiquettes, pas même d'affectations à proprement parler ; n'est-ce pas là de la *programmation structurée* ?

L'application des algorithmes élémentaires suppose qu'on exécute simultanément des opérations plus élémentaires encore :

$$V_2[\langle W, X, Y, Z \rangle] = \langle W, X, \{0\}, \{0\} \rangle, W_4[\langle W, X, Y, Z \rangle]$$
$$= \langle W, X, suc\,[Y] \cap \mathbf{N}, + [Z, W[Y]] \cap \mathbf{N} \rangle ;$$

calcul parallèle, semble-t-il ?

Les algorithmes sont traités *indépendamment de toute représentation* : l'algorithme

$$|_{\{0,\ldots,n\},\mathbf{N}}[S[carré]] = \bigcap_{i \in \{0,\ldots,n\}} F\{i\} \left\{ \sum_{j=0}^{n} i^2 \right\}$$

est-il une procédure ? un tableau ?

Un appel de procédure équivaut au résultat fourni par le programme qui le définit, et aussi à toutes ses transformations intermédiaires :

$$S[carré, \{5\}] = \ldots$$
$$= \mathbf{N} \cap V_5[Tq_{\{\text{VRAI}\},\{\text{FAUX}\}}[V_{3bis}, V_4, \langle carré, \{5\}, \{0\}, \{0\} \rangle]]$$
$$= \ldots = \{55\}.$$

N'est-ce pas une invite à *démontrer* au moins *quelques propriétés élémentaires des programmes* ?

Les déclarations sont des exemples typiques d'*algorithmes non déterministes* ; quiconque, pourtant, entreprend d'écrire un interpréteur conversationnel n'en est-il pas moins obligé de les traiter comme des instructions ?

Le calcul numérique n'est qu'un champ parmi d'autres où fleurit l'algorithme. Que n'avons-nous choisi de traduire un *programme commandant* une machine-outil, un ordinateur, ou *Dieu sait quel mécanisme* !

Et si même la théorie que nous venons de présenter n'était qu'*une méthode pour définir la sémantique des langages de programmation*, nous nous estimerions payés de notre peine.

Nous nous sommes inspirés essentiellement des deux ouvrages suivants, qui contiennent d'ailleurs une bibliographie très complète :

H. B. CURRY, R. FEYS, W. CRAIG, *Combinatory Logic*, Vol. 1, North-Holland, Amsterdam, 1958.

H. B. CURRY, J. R. HINDLEY, J. P. SELDIN, *Combinatory logic*, vol. 2, North-Holland, Amsterdam-London, 1972.