Dactyl

UNIVERSITE DE NANCY I

U. E. R. DE MATHEMATIQUES

STRUCTURES D'INFORMATION

formalisation des notions d'accès et de modifications d'une donnée

thèse

présentée pour l'obtention du doctorat de spécialité de mathématiques appliquées

par

jean luc remy

soutenue le 25 juin 1974

jury : Président M. C. PAIR

Examinateurs M. C. NIVAT

M. J.L. OVAERT

UNIVERSITE DE NANCY I

U. E. R. DE MATHEMATIQUES

STRUCTURES D'INFORMATION

formalisation des notions d'accès et de modifications d'une donnée

thèse

présentée pour l'obtention du doctorat de spécialité de mathématiques appliquées

par

jean luc remy

soutenue le 25 juin 1974

jury : Président M. C. PAIR

Examinateurs M. C. NIVAT

M. J.L. OVAERT

Je tiens à remercier ici

C. Pair à qui je dois les idées premières de ce travail. Son souci de rigueur m'a incité à approfondir toujours plus ma recherche. Grâce à ses remarques, j'ai amélioré la présentation des démonstrations et de l'ensemble du texte.

M. Nivat dont les remarques m'ont permis de préciser ce travail.

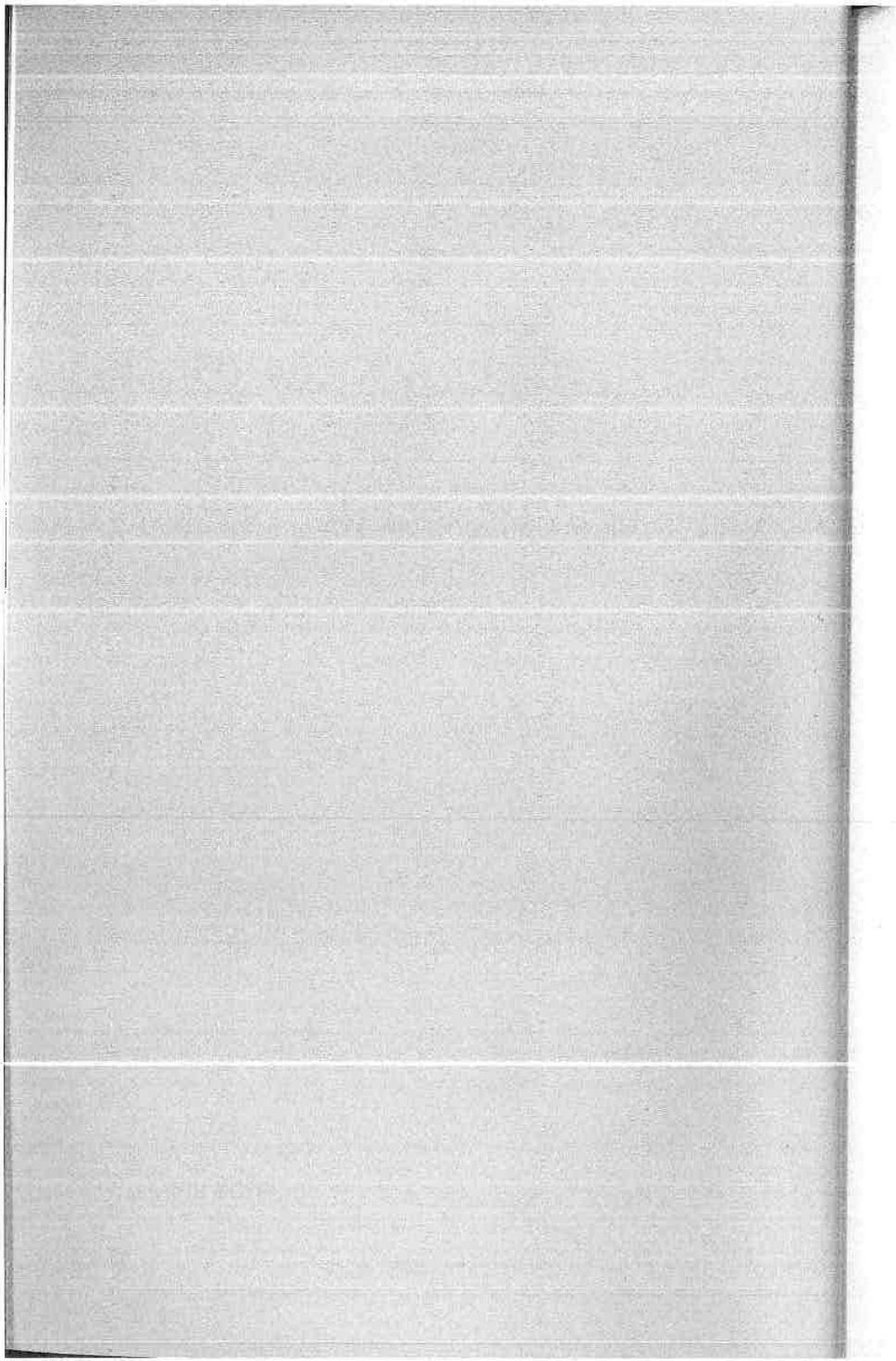
J.L. Ovaert qui m'a soutenu chaleureusement au cours de cette recherche.

J.P. Ferrier avec qui j'ai commencé mon activité de chercheur, en analyse. Malgrè le peu de temps que je consacre actuellement à ce domaine, il m'a toujours encouragé à poursuivre cette thèse.

Je tiens aussi à remercier J.P. Finance et P. Lescanne avec lesquels j'ai particulièrement travaillé ainsi que P. Marchand, R. Mohr et A. Quere avec lesquels j'ai discuté régulièrement.

Ma reconnaissance va enfin à Madame Driquert dont j'ai mis la patience à rude épreuve pour la frappe de cette thèse.

sommaire



introduction

INTRODUCTION

Le but principal de ce travail est d'apporter un cadre formel pour l'étude des problèmes informatiques. On entend ici par problème le passage d'une information donnée à une autre information. Plus généralement, un problème s'applique à des informations d'un certain type. Il s'agit donc de préciser les notions de données, d'informations et de type d'information (ou de structure d'information).

Une première idée, étudiée au paragraphe 1.1, est de définir une donnée par des ensembles $E_1,\ldots,E_m$ et un ensemble Λ d'applications(ou accès) définies sur des ensembles de la forme $E_{i_1} \times \ldots \times E_{i_q}$ et à valeurs dans l'un des E_j. On dira alors que deux données sont de même type si leurs accès vérifient les mêmes propriétés.

Une manière d'exprimer les propriétés des accès d'une donnée consiste à construire un système formel dans lequel chaque application de Λ est représentée par un symbole et chaque objet de $E_1,\ldots,E_m$ par un schéma fonctionnel. On a regroupé au chapitre 0 quelques rappels sur la théorie des langages, les schémas fonctionnels, les systèmes formels et le calcul des propositions. Les définitions de système fonctionnel, d'information et d'interprétation d'une information sont données au paragraphe 1.2.

Cette méthode axiomatique nous permet de définir les structures d'information

de la même manière que l'on définit en mathématiques les structures de groupe et de corps. Nous nous efforçons, au cours de ce travail, d'utiliser cette méthode à d'autres applications. On définira des systèmes d'axiomes pour les informations (1.), les modifications d'uns structure d'information (5.) et pour définir récursivement des accès nouveaux (3. et 4.).

Au chapitre 2, nous présentons quelques résultats, classiques en logique, sur les théorèmes d'un système fonctionnel. Nous comparons, en particulier, les systèmes fonctionnels avec deux autres types de systèmes formels : le calcul des propositions et le calcul des prédicats du premier ordre avec égalité. Enfin, nous dégageons deux types d'informations particulièrement intéressantes (informations consistantes et complètes). Nous étudions les relations entre informations et interprétations et nous obtenons plusieurs critères pour qu'une information soit consistante ou complète.

Au chapitre 3, nous étudions les extensions d'une structure d'information. Après avoir donné, dans un premier paragraphe, quelques propriétés générales, nous nous intéressons plus particulièrement aux extensions conservatrices. Enfin, au paragraphe 3.3, nous étudions, sur l'exemple des modes récursifs d'Algol 68, quelques techniques de constructions d'interprétations.

Au chapitre 4, nous portons notre attention sur les systèmes d'équations récursives. Nous obtenons, dans le cadre des systèmes formels, la plupart des résultats classiques. En particulier, on peut définir une règle d'induction point fixe semblable à celle de Scott. Notre travail permet de plus de préciser, grâce à un système d'axiomes, les domaines de base.

Le chapitre 5 est consacré à une étude des modifications. Une structure d'information est entièrement spécifiée par la donnée d'un système fonctionnel, précisant les propriétés des accès, et d'un ensemble de modifications élémentaires, applications définies sur l'ensemble des informations. Une modification peut être obtenue, soit par composition des modifications élémentaires, soit comme solution minimale d'un système d'équations. Nous donnons une méthode axiomatique de définition des modifications. Enfin, pour conclure le chapitre, nous étudions brièvement les notions de programme, de calcul et de problème.

Nous développons, dans la conclusion, un certain nombre de points qui devraient être approfondis, ou même simplement abordés. Il est très probable que certaines définitions du présent travail ne sont pas adéquates. Il s'agit là simplement d'un premier essai de formalisation et il est indispensable d'étudier, dans ce cadre, des exemples de données plus complexes. Nous signalons également, en conclusion, un certain nombre de travaux reliés à cette étude.

*
* *

Avertissement

L'utilisation de symboles fonctionnels à plusieurs variables et d'objets de types distincts introduit généralement des notations assez lourdes. Nous nous sommes efforcés d'utiliser des notations abrégées permettant de se ramener au cas d'une variable. On trouvera, en fin de ce travail, un index des notations et un index terminologique.

0

rappels

0.- RAPPELS

0.1.- Théorie des langages (HOPCROFT-ULLMAN)

Définition 1 : Soit $\mathscr{L}$ un ensemble. Une suite finie, éventuellement vide d'éléments de $\mathscr{L}$ s'appelle un mot sur $\mathscr{L}$. On note $\mathscr{L}^*$ l'ensemble des mots sur $\mathscr{L}$ et $\wedge$ le mot vide sur $\mathscr{L}$.

Si $\alpha = (a_1,\ldots,a_n)$ et $\beta = (b_1,\ldots,b_p)$ sont deux mots sur $\mathscr{L}$, on définit le mot $\alpha\beta = (c_1,\ldots,c_{n+p})$ par

$$1 \leqslant i \leqslant n \implies c_i = a_i$$
$$n+1 \leqslant i \leqslant n+p \implies c_i = b_{i-n}$$

On obtient ainsi une loi de composition interne dans $\mathscr{L}^*$, appelée concaténation. En particulier $\alpha\wedge = \wedge\alpha = \alpha$.

Un langage L sur $\mathscr{L}$ est un sous ensemble de $\mathscr{L}^*$. $\mathscr{L}$ est encore appelé alphabet de L.

Définition 2 : (Opérations dans $\mathcal{P}(\mathscr{L}^*)$)

Soit $\mathscr{L}$ un alphabet, L, L' deux langages sur $\mathscr{L}$. On appelle produit de L par L' et on note LL' le langage $LL' = \{\alpha\alpha',\ \alpha \in L \text{ et } \alpha' \in L'\}$. $\{a\}L$ sera noté aL. Si L est un langage sur $\mathscr{L}$, $LL = \{\alpha\beta \ ;\ \alpha \in L \text{ et } \beta \in L\}$ se note L^2. De même $L^n = \{\alpha_1 \ldots \alpha_n \ ;\ \alpha_i \in L\}$ ($L^0 = \{\wedge\}$; $L^{i+1} = L^iL$).

Exemple : Si F est un langage et si $\supset$, (,) sont des symboles, $(F \supset F)$ est un langage.

Définition 3 : (Système à point fixe sur $\mathcal{F}(\mathcal{L}^*)$).

Soit $\mathcal{L}$ un alphabet. Un système à point fixe sur $\mathcal{F}(\mathcal{L}^*)$ est un système du type

$$(S) \qquad A_i = \bigcup_{j \in \Lambda_i} B_{ij}^1 \dots B_{ij}^{q_{ij}} \qquad \text{pour } i = 1,\dots,n$$

où $A_1,\dots,A_n$ sont les inconnues et les B_{ij}^k désignent soit des inconnues soit une partie de $\mathcal{L}$ réduite à une lettre.

Nous supposerons que, pour tout i, Λ_i est non vide (Λ_i peut être, par contre, infini) et que pour tout i,j, q_{ij} est non nul.

On peut associer à (S) l'application Φ de $\mathcal{F}(\mathcal{L}^*)^n$ dans lui-même qui associe à $(L_1,\dots,L_n)$ $\Phi(L_1,\dots,L_n) = (L_1',\dots,L_n')$ où les L_i' sont définies par

$$L_i' = \bigcup_{j \in \Lambda_i} C_{ij}^1 \dots C_{ij}^{q_{ij}}$$

avec, pour $k = 1,\dots,q_{ij}$:

$$C_{ij}^k = \begin{cases} L_r & \text{si } B_{ij}^k \text{ est l'inconnue } A_r \\ B_{ij}^k & \text{si } B_{ij}^k \in \mathcal{L} \end{cases}$$

Le n-uple de langages $(L_1,\dots,L_n)$ est solution de (S) si et seulement si $\Phi(L_1,\dots,L_n) = (L_1,\dots,L_n)$.

Autrement dit, les solutions du système (S) sont les points fixes de l'application Φ .

Théorème 1.- (théorème du point fixe)

Tout système à point fixe (S) admet une solution minimale unique. De plus, si aucun terme de (S) ne se réduit à une inconnue, (S) admet une solution unique dans $\mathcal{F}(\mathcal{L}^* - \{\Lambda\})$.

Nous généralisons la première partie du théorème au paragraphe 4.2.

Remarque : Soit (S) un système à point fixe, tel qu'aucun terme ne se réduise à une inconnue, et $(L_1,\dots,L_n)$ sa solution unique dans $\mathcal{F}(\mathcal{L}^* - \{\Lambda\})^n$. Considérons le système (S') obtenu en ajoutant à (S) une équation du type

$$A_o = \bigcup_{j \in \Lambda} B_j^1 \dots B_j^{q_j}$$

Considérons d'autre part l'équation

$$(E) \qquad A_o = \bigcup_{j \in \Lambda} C_j^1 \dots C_j^{q_j}$$

où $$C_j^k = \begin{cases} L_r & \text{si } B_j^k \text{ est l'inconnue } A_r \\ B_1^k & \text{sinon} \end{cases}$$

Soit enfin $L_o \subset \mathcal{L}^* - \{\Lambda\}$. L_o est solution de (E) si et seulement si $(L_o,\dots,L_n)$ et solution de (S'). Dans la suite nous dirons encore que (E) est une équation à point fixe.

Exemple : Soit A un langage sur un alphabet $\mathcal{L}_o$, composante de la solution d'un système à point fixe et $\supset$, $\neg$, (,) des symboles n'appartenant pas à $\mathcal{L}_o$. L'ensemble F des formules du calcul des propositions défini sur A est l'unique solution de l'équation

$$F = A \cup \neg F \cup (F \supset F)$$

0.2.- Schémas fonctionnels (KREISEL-KRIVINE)

On se donne une famille dénombrable $L_n (n = 0, 1, \dots)$ d'ensembles disjoints.

Un élément de L_n sera appelé symbole fonctionnel à n variables ou encore symbole n-aire. On pose $L = \bigcup_n L_n$.

Définition 4 : Un schéma fonctionnel (construit à l'aide des symboles de L) est un élément de l'ensemble $\hat{L}$, solution de l'équation à point fixe

$$X = \bigcup_{n \geq 0} \left(\bigcup_{f \in L_n} f \underbrace{X \ldots X}_{n \text{ fois}} \right)$$

On vérifie aisément que $\hat{L}$ est non vide si et seulement si $L_0 \neq \emptyset$. Signalons également sans démonstration les résultats suivants :

1) Tout $u \in \hat{L}$ s'écrit, de manière unique, $u = f\, u_1 \ldots u_n$ avec $f \in L_n$ et $u_1, \ldots, u_n \in \hat{L}$.

2) Etant donnés un ensemble X et, pour chaque n, une application r_n de L_n dans l'ensemble des applications de X^n dans X, il existe une application $\hat{r}$ de $\hat{L}$ dans X, et une seule, telle que, pour tout $f \in L_n$ et pour tout $u_1, \ldots, u_n \in \hat{L}$, on ait

$$\hat{r}(f\, u_1 \ldots u_n) = r_n(f)(\hat{r}(u_1), \ldots, \hat{r}(u_n))$$

Exemple : $L_0 = \{a\}$ $L_1 = \{f\}$ $L_2 = \{g\}$ $L_n = \emptyset$ $(n > 2)$

a, fa, g f^2a f^5a , f^kg f a f^ma sont des schémas fonctionnels construits à partir des symboles de L.

0.3.- Généralités sur les systèmes formels (SHOENFIELD)

Définition 5 : Un système formel $\mathcal{F} = (\mathcal{A}, F, \mathcal{X}, R)$ est la donnée

- d'un alphabet $\mathcal{A}$ (ensemble des symboles),
- d'un langage F sur cet alphabet (ensemble des formules),
- d'un sous ensemble $\mathcal{X}$ de F (ensemble des axiomes),

- d'un ensemble R de règles ou relations n-aires sur F : si r est une règle et si $(p_1, \ldots p_n, p) \in r$, on écrit $p_1, \ldots, p_n \vdash p$.

On dit qu'un sous ensemble E de F est saturé si :

$$\forall p_1 \ldots p_n,\ p \in F \quad \left((p_1 \in E \text{ et } \ldots \text{ et } p_n \in E \text{ et } p_1, \ldots, p_n \vdash p) \Rightarrow p \in E \right)$$

L'ensemble des théorèmes de $\mathcal{F}$ est le plus petit sous ensemble saturé de F contenant $\mathcal{X}$. On écrira $\vdash_{\mathcal{F}} p$ (ou $\vdash p$ s'il n'y a pas d'ambiguïté) pour noter que p est un théorème de $\mathcal{F}$.

Une suite $D = p_1, \ldots p_n$ de formules est une démonstration (de p_n) dans $\mathcal{F}$ si chaque p_i est un axiome ou se déduit par une règle de R de certaines des formules précédant p_i. Il est clair qu'une formule est un théorème si et seulement si elle possède une démonstration dans $\mathcal{F}$.

Pour montrer qu'une propriété P est vérifiée pour tout théorème de $\mathcal{F}$, il suffira de montrer que l'ensemble $\{p \in F.; P(p)\}$

i) contient $\mathcal{X}$

ii) est saturé.

(Raisonnement par récurrence sur les démonstrations de $\mathcal{F}$).

On est amené, pour raccourcir les démonstrations de $\mathcal{F}$, à énoncer des règles supplémentaires ou règles dérivées.

Nous allons préciser ces notions sur un exemple en étudiant le système formel associé au calcul des propositions.

0.4.- Calcul des propositions (SHOENFIELD)

Définition 6 : Soit A un langage sur un alphabet $\mathcal{A}_0$. Un système propositionnel sur A est un système formel $\mathcal{F} = (\mathcal{A}, F, \mathcal{X}, R)$ tel que

i) $\mathcal{A} = \mathcal{A}_0 \cup \{\neg, \supset, (,)\}$

ii) $F = A \cup \neg F \cup (F \supset F)$

iii) $\mathcal{X} = \mathrm{Prop}(A)$ est formé des 3 ensembles (ou schémas) d'axiomes suivants :

$$X_1 = \{(p \supset (q \supset p)) \ ; \ p,q \in F\}$$

$$X_2 = \{((p \supset (q \supset r)) \supset ((p \supset q) \supset (p \supset r))) \ ; \ p,q,r \in F\}$$

$$X_3 = \{((\neg p \supset \neg q) \supset (q \supset p)) \ ; \ p,q \in F\}$$

iv) l'unique règle de R est le "modus ponens" : pour tout p,q de F $p, p \supset q \vdash q$.

Les formules de A sont appelées formules atomiques.

Abréviations et notations

On adopte les notations suivantes :

$(p \vee q)$ tient lieu de $(\neg p \supset q)$

$(p \wedge q)$ tient lieu de $\neg(\neg p \vee \neg q)$

$(p \leftrightarrow q)$ tient lieu de $((p \supset q) \wedge (q \supset p))$

Il faut noter que $\vee$, $\wedge$, $\leftrightarrow$ ne sont pas des symboles de $\mathcal{L}$. On peut, toujours à partir d'une formule contenant $\vee$, $\wedge$, $\leftrightarrow$ reconstruire la formule de F qui lui correspond.

On supprime également certaines parenthèses en convenant d'évaluer les expressions de la droite vers la gauche. Ainsi

$p \supset q \supset r$ tient lieu de $(p \supset (q \supset r))$

Les systèmes propositionnels permettent de formaliser les propriétés de l'implication ($\supset$) et de la négation ($\neg$) indépendamment de la nature des formules atomiques. L'intérêt de tels systèmes réside dans le fait que les théorèmes de $\mathcal{F}$ sont exactement les formules tautologiques, comme par exemple

$p \supset p$ (reflexivité de l'implication)

$(p \supset q) \supset (q \supset r) \supset (p \supset r)$ (transitivité de l'implication)

(Dans les exemples précédents et dans la suite, p,q,r désigneront des formules quelconques). Plus précisément, introduisons les définitions suivantes.

Définition 7 : Une fonction de vérité sur un ensemble A est une application de A dans l'ensemble $B = \{\underline{\text{vrai}}, \underline{\text{faux}}\}$.

Considérons les deux applications $H_\neg$ de B dans lui-même et $H_\supset$ de B^2 dans B définies ci-dessous :

$$H_\neg(\underline{\text{vrai}}) = \underline{\text{faux}} \qquad H_\neg(\underline{\text{faux}}) = \underline{\text{vrai}}$$

$$H_\supset(a,b) = \begin{cases} \underline{\text{faux}} & \text{si } a = \underline{\text{vrai}} \text{ et } b = \underline{\text{faux}} \\ \underline{\text{vrai}} & \text{sinon} \end{cases}$$

$H_\neg$ et $H_\supset$ définissent les "tables de vérité" de l'implication et de la négation.

Proposition 1.- Soit V une fonction de vérité sur A. Il existe une application unique $\tilde{V}$, de F dans B, prolongeant V et telle que

$$\tilde{V}(\neg p) = H_\neg(\tilde{V}(p)) \ , \ \tilde{V}((p \supset q)) = H_\supset(\tilde{V}(p), \tilde{V}(q))$$

La démonstration est immédiate en raisonnant par récurrence sur la longueur d'une formule de F.

Définition 8 : Une formule p est une tautologie si $\tilde{V}(p) = \underline{\text{vrai}}$ pour toute fonction de vérité V sur A. Plus généralement, on dit que q est conséquence tautologique de $p_1, \ldots p_n$ si, pour toute fonction de vérité V sur A,

$$\tilde{V}(p_1) = \ldots = \tilde{V}(p_n) = \underline{\text{vrai}} \implies \tilde{V}(q) = \underline{\text{vrai}}$$

Théorème 2.- (théorème de tautologie). Soit $\mathcal{F} = (\mathcal{L}, F, \mathcal{X}, R)$ un système propositionnel. Tout théorème de $\mathcal{F}$ est une tautologie. Réciproquement toute

tautologie est un théorème de $\mathcal{F}$. Plus généralement doit $\mathcal{F} = (\mathcal{L}, F, \mathcal{X}', R')$ un système formel tel que $\mathcal{X}'$ contienne $\mathcal{X}$ et R' contienne R. Si $\vdash_{\mathcal{F}} p_1, \ldots, \vdash_{\mathcal{F}} p_n$ et si q est conséquence tautologique de $p_1, \ldots, p_n$, alors $\vdash_{\mathcal{F}} q$.

Démonstration : On démontre immédiatement que tout axiome de $\mathcal{F}$ est une tautologie. De plus, si p et $p \supset q$ sont des tautologie, q est une tautologie. L'ensemble des tautologies est saturé, donc contient l'ensemble des théorèmes de $\mathcal{F}$.

La réciproque est plus longue à démontrer (voir par exemple (SCHOENFIELD)). Le dernier point est un corollaire immédiat : si q est conséquence tautologique de $p_1, \ldots, p_n$, la formule $p_1 \supset \ldots \supset p_n \supset q$ est une tautologie. Donc $\vdash_{\mathcal{F}} p_1 \supset \ldots \supset p_n \supset q$. Et, puisque $\vdash_{\mathcal{F}} p_1, \ldots \vdash_{\mathcal{F}} p_n$, il vient, en appliquant la règle de modus ponens, $\vdash_{\mathcal{F}} q$.

Le théorème 1 exprime la complétude du calcul des propositions, dans le sens suivant : le calcul des propositions étant indépendant de la signification des formules atomiques, une interprétation de ce calcul consiste en une application arbitraire V de A dans $\{\underline{\text{vrai}}, \underline{\text{faux}}\}$. On dit qu'une formule p de F est vraie pour V si $\tilde{V}(p) = \underline{\text{vrai}}$ et que p est universellement vraie si $\tilde{V}(p) = \underline{\text{vrai}}$ pour tout V. On peut définir la notion d'interprétation et de formule universellement vraie (ou valide) pour d'autres systèmes formels. On dira alors que $\mathcal{F}$ est complet si les théorèmes de $\mathcal{F}$ sont exactement les formules valides de $\mathcal{F}$.

Applications du théorème 2

Le théorème 2 permet d'appliquer les règles suivantes dans les démonstrations de $\mathcal{F}$.

$p \supset q,\ q \supset r \vdash p \supset r$ (transitivité de $\supset$)

$p_1 \supset p_2 \supset \ldots \supset p_n \supset q \vdash p_{i_1} \supset \ldots \supset p_{i_n} \supset q$ (si $(i_1, \ldots, i_n)$ est une permutation de $[1,n]$)

$p \supset q,\ \neg p \supset q \vdash q$

$p \supset q,\ p \supset r \vdash p \supset (q \wedge r)$

$p, q \vdash p \wedge q$

$p, \neg q \vdash \neg(p \supset q)$

$\neg p \vdash p \supset q$

$q \vdash p \supset q$

Théorème 3.- (Règle de l'équivalence)

Soient $p_1, \ldots, p_n$, $p'_1, \ldots, p'_n$, q des formules de F. Soit q' la formule obtenue en remplaçant dans q des occurrences de $p_1, \ldots, p_n$ respectivement par $p'_1, \ldots, p'_n$. Alors

$$p_1 \leftrightarrow p'_1, \ldots, p_n \leftrightarrow p'_n \vdash q \leftrightarrow q'$$

La démonstration est immédiate en raisonnant par récurrence sur la longueur de q et en utilisant le théorème de tautologie.

1

structures d'information

1.- STRUCTURES D'INFORMATION

1.1.- Point de vue fonctionnel (PAIR - FINANCE)

1.1.1.- Exemples de données - Définitions

Essayons tout d'abord de comprendre sur des exemples ce qu'est une information composée.

a) Liste

Une liste sur un ensemble E est une suite finie d'éléments de E. Pour préciser cette notion de suite, et en particulier pour exprimer les accès aux éléments d'une telle suite, on introduit un ensemble fini F totalement ordonné (ensemble des "places" dans la suite étudiée) dont le premier élément τ s'appelle tête de liste et le dernier δ est la queue de liste. L'ordre total se traduit par la donnée d'une fonction de succession dans F, notée σ, qui est une bijection de $F - \{\delta\}$ dans $F - \{\tau\}$. Alors tout élément de F est l'image de τ par σ^i pour un certain i. Enfin, une application ν assigne à chaque élément de F une valeur dans E.

Ainsi une liste est un triplet (F, E, Λ) où Λ est l'ensemble de fonctions $\{\sigma, \nu, \tau\}$. (τ est ici considéré comme une fonction à 0 variable.. L'accès à un élément quelconque de la liste passe par l'accès à l'élément de tête).

Adjoindre un élément en tête d'une liste $(F, E, \wedge)$ revient à construire une nouvelle liste $(F', E, \wedge')$ en posant

$$F' = F \cup \{\tau'\}$$
$$\wedge' = \{\sigma', \nu', \tau'\}$$
$$\begin{cases} \sigma'(\tau') = \tau \\ \sigma'(x) = \sigma(x) & x \in F - \{\delta\} \end{cases}$$
$$\nu'(x) = \nu(x) \qquad x \in F$$

On obtient en fait seulement une quasi-liste puisque la valeur assignée à τ' n'est pas définie. On constate également sur cet exemple simple qu'il n'est pas facile de définir des modifications dans ce point de vue.

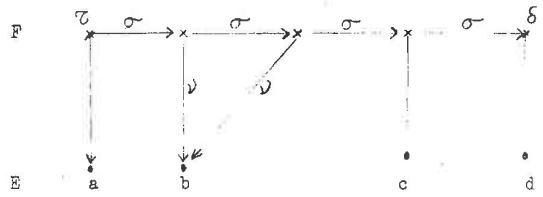


Figure 1 : schéma de la liste abbcd

b) Gestions de produits dans des stocks

Une usine dispose d'un ensemble P de produits rangés dans un ensemble D de dépôts. Pour chaque produit p, on constitue une liste $d_1(p), \ldots, d_{k_p}(p)$ des dépôts dans lesquels ce produit est stocké et pour chaque dépôt d, on constitue de même une liste $p_1(d), \ldots, p_{\ell_d}(d)$ des produits stockés dans ce dépôt. Enfin pour chaque produit et pour chaque dépôt on connait la quantité du produit stockée dans ce dépôt.

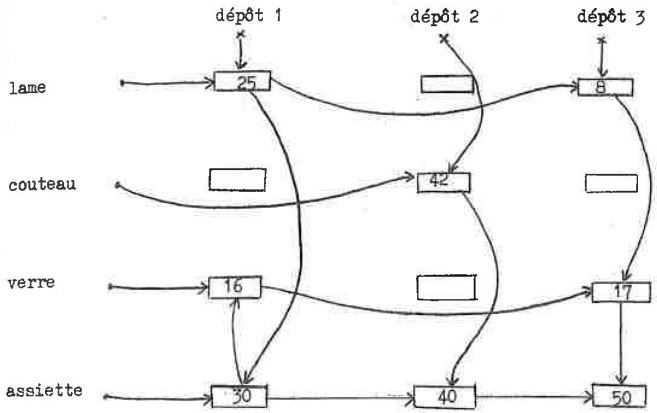


Figure 2 : gestion d'un stock

On peut représenter cette donnée en considérant les ensembles suivants :

D = ensemble des dépôts

P = ensemble des produits

F = D x P

$\mathbb{Q}$ = ensemble des quantités

et les fonctions suivantes :

τ_1 associé à chaque dépôt d le couple $(d, p_1(d))$

τ_2 associé à chaque produit p le couple $(d_1(p), p)$

σ_1 associé à chaque couple (d,p) le couple (d,p'), s'il existe, tel que p' suive p dans la liste $p_1(d), p_2(d) \ldots$

σ_2 associé à chaque couple (d,p) le couple (d',p) s'il existe, tel que d' suive d dans la liste $d_1(p), d_2(p) \ldots$

π_1 associé à chaque couple (d,p) sa première composante d

π_2 associé à chaque couple (d,p) sa deuxième composante p

χ associé à chaque couple (d,p) la quantité du produit p stockée dans d.

Enfin l'on a accès directement à chaque produit et à chaque dépôt ; aussi faut-il considérer D et P comme des ensembles de fonctions 0-aires.

La donnée ci-dessus est donc un quintuplet $(D, P, F, \mathcal{Q}, \wedge)$ avec

$$\wedge = D \cup P \cup \{\tau_1, \tau_2, \sigma_1, \sigma_2, \pi_1, \pi_2, \chi\}$$

Notons que, dans la figure 2, des couples tels que (dépôt 2, lame) ou (dépôt 1, couteau) ne sont pas "accessibles".

Nous pouvons plus généralement définir une donnée de la manière suivante :

Définition 1 : Une donnée est un $m+1$ - uple $(E_1,\ldots,E_m, \wedge)$ où les E_i $(i = 1,\ldots,m)$ sont des ensembles d'objets et $\wedge$ un ensemble de fonctions (partielles) définies dans des ensembles de la forme $E_{i_1} \times \ldots \times E_{i_q}$ $(q \geqslant 0$ et $1 \leqslant ij \leqslant m,.)$ à valeurs dans l'un des E_j. Une telle fonction s'appellera accès élémentaire à q variables, une fonction à 0 variable étant confondue avec sa valeur.

Remarque : Il est toujours possible de considérer des fonctions totales à la place de fonctions partielles en introduisant un élément $\perp$ (élément "non défini"). Si E est un ensemble ne contenant pas $\perp$, on notera $E_+ = E \cup \{\perp\}$. On prolonge alors une fonction partielle f, définie dans E, en posant

$$\begin{cases} \bar{f}(x) = \text{si } f(x) \text{ est définie alors } f(x) \text{ sinon } \perp \quad (x \in E) \\ \bar{f}(\perp) = \perp \end{cases}$$

1.1.2.- Structures de donnée

Nous avons donc formalisé la notion de donnée. Nous avons également présenté des exemples de structures : listes, gestion de stocks. Chaque fois, nous avons défini ces structures en exprimant les propriétés vérifiées par les ensembles de base et les accès constituant une donnée. Il est difficile d'énoncer des résultats

généraux sur les structures sans donner un sens mathématique à la notion de propriété. On peut apporter une réponse axiomatique à ce problème en associant à une structure de donnée un système formel. (C'est ainsi que l'on définit, en mathématiques, les structures de groupes, de corps ...). Les propriétés que doivent vérifier les données d'une structure sont des formules, ou axiomes, construites sur un alphabet constitué de symboles fonctionnels ou accès, de variables, du symbole $\equiv$ et des symboles $\supset$ (implication), $\neg$ (négation), (,) (parenthèses) du calcul des propositions.

Une structure n'est pas entièrement spécifiée par son système formel. Il est important de préciser les transformations ou modifications élémentaires applicables aux données de la structure. Pour les listes, par exemple, on veut pouvoir :

- modifier la valeur d'un élément
- adjoindre un élément en tête de liste
- supprimer un élément, etc ..

Cet aspect sera traité au chapitre 5.

1.2.- Systèmes fonctionnels

Dans l'approche précédente, une donnée $(E_1,\ldots,E_m, \wedge)$ est un objet "concret". Par opposition, nous considérons une information comme un objet "idéal" qui permet d'exprimer des affirmations. Brièvement, une information sera l'ensemble des théorèmes d'un système formel (cf. 0.3). Une information pourra alors être interprétée (au sens de la logique) par une donnée.

Une structure d'information $\mathcal{S}$ sera constituée d'un système formel $\mathcal{F}$ décrit dans ce paragraphe et d'un ensemble $\mathcal{M}$ de modifications élémentaires étudié au chapitre 5. Dans la suite, nous emploierons quelquefois des expressions telles que alphabet de $\mathcal{S}$, axiomes de $\mathcal{S}$, théorèmes de $\mathcal{S}$, information de $\mathcal{S}$ au lieu de alphabet de $\mathcal{F}$, axiomes de $\mathcal{F}$, théorèmes de $\mathcal{F}$, information de $\mathcal{F}$.

Définition 2 : Un système fonctionnel est un système formel $\mathcal{F} = (\mathcal{L}, F, \mathcal{X}, R)$ vérifiant les conditions énoncées dans les paragraphes suivants ; les formules de F sont des formules propositionnelles sur un ensemble A c'est-à-dire que F vérifie l'équation

$$F = A \cup \neg F \cup (F \supset F)$$

1.2.1.- Alphabet

On se donne un entier $m \geqslant 1$ qui est le nombre de types d'objets du système : on note $[m]$ l'intervalle des entiers compris entre 1 et m, c'est-à-dire $[m] = \{ n \in \mathbb{N} \ ; \ 1 \leqslant n \leqslant m \}$.

$\mathcal{A} = L \cup V \cup \{\equiv\} \cup \{(\ ,\ ), \neg, \supset\}$ est l'alphabet de F

$\equiv$ est le symbole d'égalité : son emploi est régi par les axiomes Eg (cf 1.2.5)

L est l'ensemble des symboles fonctionnels

$V = \bigcup_{j=1}^{m} V_j$ est l'ensemble des variables.

On définit une application source de L dans $[m]^*$ (cf. 0.1.) ; source (f) est un mot sur $[m]$, c'est-à-dire une suite d'entiers compris entre 1 et m. La longueur de source (f) s'appelle l'arité de f. Si elle est égale à 0, 1, 2, q respectivement f est appelé symbole 0-aire, unaire, binaire, q-aire. Les symboles 0-aires sont aussi appelés symboles de constante. On définit une application but de L dans $[m]$. A partir de source et de but on construit une application profil de L dans $[m]^* \times [m]$ telle que :

$$\text{profil}\ (f) = (\text{source}(f), \text{but}(f))$$

Notations : Dans la suite nous utiliserons les lettres a, b, c pour les symboles de constante et les lettres f, g pour les symboles fonctionnels. $x = (x_1,\ldots,x_n)$ désigne toujours un n-uple de variables distinctes. On note alors $L(x) = L \cup \{x_1,\ldots,x_n\}$. On pose enfin $L' = L \cup V$.

Les symboles fonctionnels sont encore les accès (élémentaires) du système. Ceci est justifié par le fait que tous les objets considérés dans le système formel sont obtenus par composition de ces accès (cf ci-dessous, en particulier la définition 5).

Exemples : Dans la structure de liste on considère deux types d'objets (les places et les valeurs). $L = \{t, s, v, \underline{nil}, \underline{nil}'\}$

$$\text{profil}(t) = (1) \qquad \text{profil}(\underline{nil}) = (1) \qquad \text{profil}(\underline{nil}') = (2)$$

$$\text{profil}(s) = (1,1) \qquad \text{profil}(v) = (1,2)$$

Remarque : nil représente l'élément fictif qu'il est commode de placer en fin de liste. On pose $\sigma(\delta) = \sigma(\underline{nil}) = \underline{nil}$ et $\nu(\underline{nil}) = \underline{nil}'$

1.2.2.- Schémas fonctionnels du système

Nous considérons seulement certains schémas fonctionnels (cf. 0.2), ceux qui sont obtenus en tenant compte des profils dans la composition. Introduisons auparavant quelques notations.

Notations

1) Si $A_1,\ldots,A_m$ sont des ensembles et si $i = (i_1,\ldots,i_n) \in [m]^*$, on pose $A_i = A_{i_1} \times \ldots \times A_{i_n}$. La lettre i désigne toujours par la suite un élément de $[m]^*$.

2) Si $u = (u_1,\ldots,u_n) \in (L'^*)^n$ et $f \in L$, on note fu au lieu de $fu_1 \ldots u_n$.

3) Si $A \subset (L'^*)^n$, $fA = \{fu \ ; \ u \in A\}$

Définition 3 : Un schéma fonctionnel compatible avec l'application profil est un élément du langage S' sur L' défini par $S' = \bigcup_{j=1}^{m} S'_j$ où $(S'_1,\ldots,S'_m)$ est la solution unique du système

$$S'_j = \bigcup \left\{ f\, S'_i \ ; \ f \in L \text{ et } \underline{\text{profil}}\,(f) = (i,j) \right\} \ \cup \ V_j$$

On appellera aussi <u>termes</u> les schémas fonctionnels compatibles avec <u>profil</u>. Un schéma fonctionnel u est de <u>type</u> j si $u \in S'_j$.

Pour $x = (x_1,\ldots,x_n)$, on pose $S_j(x) = S'_j \cap L(x)^*$. En particulier $S_j = S_j(\wedge)$ est l'ensemble des termes de type j sans variable. On pose enfin $S = \bigcup_{j=1}^{m} S_j$ et $S(x) = \bigcup_{j=1}^{m} S_j(x)$.

<u>On suppose que $S_j \neq \emptyset$ pour $j = 1,\ldots,m$</u>

<u>Exemple</u> : Dans la structure de liste, les termes sans variable sont les éléments de l'ensemble S défini par

$$S = S_1 \cup S_2$$

$$S_1 = sS_1 \cup \{t, \underline{\text{nil}}\}$$

$$S_2 = vS_1 \cup \{\underline{\text{nil}}'\}$$

<u>Notations</u> : Nous utiliserons les lettres u, v, w pour désigner des schémas fonctionnels ou des n-uples de schémas fonctionnels. Dans le cas où ces schémas fonctionnels contiennent des variables, on utilisera également les lettres h, k.

Soit h un schéma fonctionnel, $i = (i_1,\ldots,i_n) \in [m]^*$, $x = (x_1,\ldots,x_n) \in V_i$ et $u = (u_1,\ldots,u_n) \in S'_i$. On note $h_x[u]$ (ou $h[u]$ s'il n'y a pas d'ambiguïté) le terme obtenu en remplaçant dans h toutes les occurrences de $x_1,\ldots,x_n$ par $u_1,\ldots,u_n$ respectivement. On pose encore $\underline{\text{type}}(u) = \underline{\text{type}}(x) = i$.

<u>Définition 4</u> : Une <u>interprétation de S</u> est un $(m+1)$-uple $R = (E_1,\ldots,E_m,r)$ où les E_i sont des ensembles non vides et r une application associant à tout $f \in L$ tel que $\underline{\text{profil}}(f) = (i,j)$ une application de E_i dans E_j. Si $E = (E_1,\ldots,E_m)$ on note encore $R = (E,r)$.

Nous utiliserons par la suite la λ-notation. Soit $E = (E_1,\ldots,E_m)$, $i \in [m]^*$ et $x = (x_1,\ldots,x_n) \in V_i$. Si $u(x)$ est une expression de fonction, la notation $\lambda x.u(x)$ représente la fonction qui a tout x de E_i associe $u(x)$.

<u>Exemple</u> : $\lambda x.x_k$ représente l'application π_k définie par

$$\pi_k(x_1,\ldots,x_n) = x_k \text{ pour tout } (x_1,\ldots,x_n) \text{ de } E_i.$$

<u>Proposition 1</u> : Soit (E,r) une interprétation de S, $x = (x_1,\ldots,x_n) \in V_i$ r détermine une application unique $\hat{r}$ de $S(x)$ dans l'ensemble des applications de E_i dans $\bigcup_j E_j$ telle que

$$\hat{r}(x_k) = \lambda x.x_k \qquad \text{pour } k = 1,\ldots,n$$

$$\hat{r}(a) = \lambda x.r(a) \qquad \text{si } a \text{ est une constante}$$

$$\hat{r}(g\, u_1 \ldots u_n) = r(g)(\hat{r}(u_1),\ldots,\hat{r}(u_n)) \quad \text{si } g \in L$$

La démonstration est immédiate et analogue à la démonstration pour les schémas fonctionnels en général (0.2).

<u>Remarque</u> : En toute rigueur, il faudrait écrire $\hat{r}_x$ au lieu de $\hat{r}$. Notons cependant que si $\{x_1,\ldots,x_n\} \subset \{y_1,\ldots y_p\}$ et si $u \in S(x)$, on peut identifier $\hat{r}_x(u)$ et $\hat{r}_y(u)$. En particulier, si u est un terme sans variable de type j ($u \in S_j$), $\hat{r}(u)$ est une application constante que l'on identifie avec sa valeur.

<u>Définition 5</u> : Une interprétation (E,r) de S est <u>stricte</u> si $\hat{r}(S_j) = E_j$ pour $j = 1,\ldots,m$.

1.2.3.- Formules atomiques

L'ensemble A des formules atomiques est défini par le système

$$A = \bigcup_{j=1}^{m} S'_j \equiv S'_j$$

Prédicats : Supposons que L contient un symbole vrai de type j. On appelle symbole propositionnel (ou prédicat) tout symbole p tel que but(p) = j. On conviendra d'écrire alors $p\ u_1 \dots u_n$ au lieu de $p\ u_1 \dots u_n \equiv$ vrai. Ceci est une simple réécriture (de la même manière que $p \vee q$ est une réécriture de $\neg p \supset q$).

Exemple : Le système fonctionnel $\mathcal{N}$, formalisant l'arithmétique comporte 2 types d'objets : le type entier et le type booléen. Les symboles de $\mathcal{N}$ sont : (on indique entre parenthèses le profil)

0	(1)	S	(1,1)	+	(1,1,1)	*	(1,1,1)
vrai	(2)	<	(1,1,2)				

La formule $S0 < 0$ est une réécriture de $< (S0,0) \equiv$ vrai.

Dans la suite on omettra d'indiquer le type booléen et on spécifiera un prédicat p en indiquant uniquement source (p). (Ici source (<) = (1,1)).

Nous écrirons également $x + y$ et $x * y$ au lieu de $+ xy$ et $* xy$.

1.2.4.- Formules

L'ensemble F est la solution de l'équation à point fixe

$$F = A \cup \neg F \cup (F \supset F)$$

Il revient au même de dire

1) toute formule atomique est une formule

2) si p est une formule, $\neg p$ est une formule

3) si p et q sont des formules, $(p \supset q)$ est une formule.

4) toute formule est obtenue par application des 3 règles précédentes.

Pour démontrer qu'une propriété P est vraie pour toute formule il suffit de montrer (raisonnement par récurrence sur la longueur d'une formule) :

1) P est vérifiée pour toute formule atomique.

2) Si P(p) est vérifiée, $P(\neg p)$ est vérifiée.

3) Si P(p) et P(q) sont vérifiées, $P((p \supset q))$ est vérifiée.

Rappelons les règles de réécriture et les conventions adoptées en 2.1. On peut écrire

$(p \vee q)$ pour $(\neg p \supset q)$

$(p \wedge q)$ pour $\neg(\neg p \vee \neg q)$

$(p \leftrightarrow q)$ pour $((p \supset q) \wedge (q \supset p))$

De plus, on écrit $u \not\equiv v$ au lieu de $\neg u \equiv v$

1.2.5.- Axiomes

$\mathcal{X} =$ Prop(A) $\cup$ Eg $\cup$ X est l'ensemble des axiomes de $\mathcal{F}$.

. les formules de Prop(A) sont les axiomes du système propositionnel construit sur A (cf 0.4).

. les formules de Eg définissent le symbole d'égalité $\equiv$. Pour chaque j de $[m]$ soit x_j, y_j, z_z trois variables déterminées de type j.

Plus généralement, pour chaque i de $[m]^*$ tel que $q = |i| > 0$, soit $(x_{i1},\dots,x_{iq})$ et $(y_{i1},\dots,y_{iq})$ deux suites de variables distinctes,

de type i, telles que $x_{ik} \neq y_{i\ell}$ pour $k,\ell \in [1,q]$. Posons

$p_j = x_j \equiv x_j$ pour $j \in [m]$

$q_j = x_j \equiv y_j \supset x_j \equiv z_j \supset y_j \equiv z_j$ pour $j \in [m]$ et, si f est un symbole q-aire $(q \neq 0)$ tel que $\underline{\text{source}}(f) = i$

$r_f = x_{i1} \equiv y_{i1} \supset \ldots \supset x_{iq} \equiv y_{iq} \supset fx_{i1} \ldots x_{iq} \equiv fy_{i1} \ldots y_{iq}$

$Eg = \{ p_j \; ; \; j \in [m] \} \cup \{ q_j \; ; \; j \in [m] \} \cup \{ r_f \; ; \; f \in L \text{ et arité}(f) > 0 \}$

. les formules de X sont les axiomes propres du système fonctionnel.

Exemple :

1) les axiomes propres de la structure de liste seront

. $sx \equiv sy \supset x \equiv y \vee sx \equiv \underline{nil}$

. $s_x \equiv t \supset t \equiv \underline{nil}$

. $s\,\underline{nil} \equiv \underline{nil}$, $vx \equiv \underline{nil}' \leftrightarrow x \equiv \underline{nil}$

Les axiomes ci-dessus expriment que

. ou bien la liste est infinie : dans ce cas $s^k t \neq s^\ell t$ $(k \neq \ell)$

. ou bien la liste est de longueur finie n : dans ce cas $s^n t = \underline{nil}$ et $s^k t \neq s^\ell t$ $(0 \leqslant k, \ell < n$ et $k \neq \ell)$

2) les axiomes propres de la structure $\mathcal{N}$ (entiers positifs) sont les axiomes de Peano

$Sx \not\equiv 0$ $\qquad$ $x.Sy = (x.y) + x$

$Sx \equiv Sy \supset x \equiv y$ $\qquad$ $\neg(x < 0)$

$x + 0 \equiv x$ $\qquad$ $x < Sy \leftrightarrow (x < y \vee x \equiv y)$

$x + Sy \equiv S(x+y)$ $\qquad$ $x < y \vee x \equiv y \vee y < x$

$x.0 \equiv 0$

3) Dans toute structure, il est utile d'introduire, pour tout $(i,j) \in [m] \times [m]$, un symbole $\text{cond}_{i,j}$ de profil (i,i,j,j,j) formalisant la fonction "conditionnelle". Les axiomes de $\text{cond}_{i,j}$ seront

$\text{cond}_{i,j}(x,x,z,t) \equiv z$

$x \not\equiv y \supset \text{cond}_{i,j}(x,y,z,t) \equiv t$

Par la suite, on omettra les indices i,j , le contexte indiquant toujours leurs valeurs.

1.2.6.- Règles d'inférence

On dispose de deux règles d'inférence

modus ponens : $p,\ p \supset q \vdash q$

substitution : $p \vdash p_x[u]$ $\quad (p \in F,\ x \in V,\ u \in S'$ $\underline{\text{type}}(u) = \underline{\text{type}}(x))$

Remarquons dès maintenant que si $p \in T$, alors le schéma de formules $\{ p_x[u] \; ; \; u \in S_j \}$ est contenu dans T.

Nous étudierons au chapitre suivant la portée exacte de la règle de substitution : il est possible de se passer de cette règle à condition de prendre pour axiomes de $\mathcal{F}$ l'ensemble des "exemplaires" de formule de $\mathcal{X}$.

L'usage des variables présente cependant plusieurs avantages.

1) Il est plus agréable de manipuler des axiomes que des schémas d'axiomes. De même, étant donnée une formule p sur L(x), une démonstration de p sera

plus lisible qu'une démonstration de $p\,[u]$ $(u \in S_i)$.

2) Lorsqu'on définira de nouveaux accès, c'est-à-dire de nouveaux symboles fonctionnels (chapitre 3 et 4), la règle de substitution s'appliquera aux termes contenant des occurrences de ces nouveaux symboles. Ainsi, dans la structure $\mathcal{N}$ des entiers positifs, définissons le quotient de deux nombres par la formule

$$y \neq 0 \supset (x < y*(\underline{quotient}(x,y)+1) \wedge (y*\underline{quotient}(x,y) \equiv x \vee y*\underline{quotient}(x,y) < x))$$

Les axiomes de l'addition, de la multiplication et du prédicat $<$ s'appliquent à $\underline{quotient}\,(m,n)$ (ce qui permet, pour chaque couple m,n $(n \neq 0)$ de calculer $\underline{quotient}\,(m,n)$).

1.2.7.- Informations

Définition 6 : Une information de $\mathcal{F}$ est un sous ensemble I de F contenant les théorèmes du système formel et saturé.

I est donc une information de la structure si

i) $\mathcal{X} \subset I$

ii) Pour tous p,q de F, $p \in I$ et $p \supset q \in I$ implique $q \in I$

iii) Pour toute formule p de F, toute variable x et tout terme u de même type que x, $p \in I$ implique $p_x[u] \in I$.

Définition 7 : Soit I une information de $\mathcal{F}$. Un ensemble Y des formules est un système d'axiomes de I si I est l'ensemble des théorèmes du système fonctionnel $\mathcal{F}'$ d'axiomes propres Y.

Exemples :

1) Listes : On peut déduire des axiomes de la structure de liste que pour tout k, $\ell \in \mathbb{N}$, $k \neq \ell$

$$\vdash s^k t \equiv s^\ell t \supset s^k t \equiv \underline{nil}$$

Pour spécifier une information il faut donc préciser sa longueur et les relations entre les valeurs de la liste. Par exemple :

$$\begin{cases} s^3 t \equiv \underline{nil} \\ s^2 t \neq \underline{nil} \\ vt \equiv vs^2 t \\ vt \neq vst \end{cases}$$

On obtient également une information en considérant seulement les 3 premiers axiomes ou plus généralement un sous ensemble strict de ces axiomes. En fait la première information est "complète" alors que les autres ne le sont pas. Nous préciserons cela au chapitre 2.

2) Listes à valeurs dans un ensemble déterminé

Dans la plupart des cas la structure de liste n'est pas étudiée dans l'absolu. On se donne un ensemble de valeurs, par exemple les entiers, c'est-à-dire que S_2 contient d'autres termes que vt, vst, ... Ainsi la structure de liste à valeurs entières est obtenue en juxtaposant la structure $\mathcal{L}$ de liste et la structure $\mathcal{N}$. Une liste à valeurs entières sera alors déterminée par un système d'axiomes du type

$$\begin{cases} s^n t = \underline{nil} \\ vs^k t \equiv n_k \end{cases} \qquad 0 \leqslant k < n-1$$

où les n_k sont des termes de $\mathcal{N}$.

Notons qu'on peut trouver des informations pathologiques. Ainsi l'information d'axiomes

$$vt \equiv s^3 0$$
$$vst \equiv s^2 0$$
$$vs^2 t \not\equiv s^n 0 \qquad n \geqslant 0$$
$$s^2 t \not\equiv \underline{nil}$$
$$s^3 t \equiv \underline{nil}$$

est complète, bien que vs^2t ne soit pas "à valeurs entières". Ce fait est inhérent à la formalisation choisie : les entiers formant une suite infinie, il n'existe pas de formule affirmant que vs^2t est à valeurs entières.

1.2.8.- Interprétations d'une information

Soit I une information, R = (E,r) une interprétation stricte de S et $x = (x_1,\ldots,x_n) \in V_i$. Il existe une application $\tilde{r}$ unique de l'ensemble des formules construites sur L(x) dans l'ensemble des applications de E_i dans $B = \{\underline{vrai}, \underline{faux}\}$ telle que

$$\tilde{r}(u \equiv v) = \underline{si}\ \hat{r}(u) = \hat{r}(v)\ \underline{alors}\ \underline{vrai}\ \underline{sinon}\ \underline{faux}$$
$$\tilde{r}(\neg p) = H_\neg(\tilde{r}(p))$$
$$\tilde{r}(p \supset q) = H_\neg(\tilde{r}(p), \tilde{r}(q))$$

Définition 8 : R est une interprétation de I si, pour tout théorème p de I (construit sur L(x))

$$\tilde{r}(p) = \lambda x.\ \underline{vrai} \qquad (1)$$

Proposition 2 : R est une interprétation de I si et seulement si, pour tout théorème de I, sans variable

$$\tilde{r}(p) = \underline{vrai}$$

Si Y est un système d'axiomes de I, il suffit que (1) soit vérifiée pour toute formule de Y.

Démonstration :

1) La condition est clairement nécessaire. Inversement soit p une formule de I sur L(x). Pour tout $u \in S_i$, p [u] est un théorème de I. Donc

$$\tilde{r}(p\,[u]) = \underline{vrai}$$

Mais $\tilde{r}(p\,[u]) = \tilde{r}(p)\,(\hat{r}(u))$

Pour tout $j \in [m]$, $E_j = \hat{r}(S_j)$. Donc $E_i = \hat{r}(S_i)$

d'où $\tilde{r}(p) = \lambda x.\underline{vrai}$

2) Si $p \in Prop(A)$, p est une tautologie. Donc $\tilde{r}(p) = \lambda x.\underline{vrai}$. De même si $p \in Eg$.

Si p est déduit de q et $q \supset p$ par la règle de modus ponens

$$\tilde{r}(q) = \lambda x.\underline{vrai}\ ,\ \tilde{r}(q \supset p) = \lambda x.\underline{vrai}$$

Donc $\tilde{r}(p) = \lambda x.\underline{vrai}$ par définition de $H_\supset$.

Si $q \in I$ et $p = q_y[u]$ $\tilde{r}(p) = \tilde{r}(q)(\hat{r}(u)) = \lambda x.\underline{vrai}$

Donc R est une interprétation de I dès que $\tilde{r}(p) = \lambda x.\underline{vrai}$ pour $p \in Y$.

1.2.9.- Structures d'information

Si $\mathcal{F}$ est un système fonctionnel, on note $\mathcal{J}(\mathcal{F})$ l'ensemble des informations de $\mathcal{F}$ et pour tout $n \geqslant 0, \ldots$ $\mathcal{A}_n(\mathcal{F})$ l'ensemble des applications de $[\mathcal{J}(\mathcal{F})]^n$ dans $\mathcal{J}(\mathcal{F})$. Soit $\mathcal{A}(\mathcal{F}) = \bigcup_{n \geqslant 0} \mathcal{A}_n(\mathcal{F})$.

Définition 9 : Une structure d'information $\mathcal{S}$ est la donnée d'un système fonctionnel $\mathcal{F}$ et d'un sous ensemble $\mathcal{M}$ de $\mathcal{A}(\mathcal{F})$ (ensemble des modifications élémentaires de $\mathcal{S}$).

Nous reprendrons au chapitre 5 l'étude des modifications d'uns structure d'information.

2

étude mathématique des systèmes fonctionnels

2.- ETUDE MATHEMATIQUE DES SYSTEMES FONCTIONNELS

Nous regroupons au paragraphe 2.1 quelques résultats classiques sur les théorèmes d'un système fonctionnel. Le théorème 1, en particulier, précise la portée de la règle de substitution. Le paragraphe 2.2 étudie la structure de l'ensemble des informations d'un système fonctionnel en tant qu'ensemble ordonné. Il développe également la notion d'interprétation d'une information et, donc, de donnée. Au paragraphe 2.3, nous nous interrogeons sur l'intérêt d'introduire le quantificateur $\exists$ dans la formalisation. Il est montré que, dans une certaine mesure, ceci n'est pas nécessaire : lorsqu'on veut résoudre un problème, il s'agit moins de montrer l'existence d'une solution que de construire cette solution en utilisant les accès précédemment définis.

Le paragraphe 2.1 peut être omis en première lecture par un logicien. Le paragraphe 2.2 est, par contre, essentiel pour la suite. Le paragraphe 2.3 enfin, est relativement indépendant des chapitres suivants.

2.1.- Résultats concernant les théorèmes d'un système fonctionnel (*)

2.1.1.- Règle de substitution

Définition 1 : Soit $x = (x_1,\ldots,x_n) \in V_i$, u un terme et p une formule

(*) Dans l'ensemble, ces résultats ne sont pas nouveaux. Nous nous sommes largement inspirés de la présentation de (SHOENFIELD).

sur $L(x)$. Un exemplaire de u (resp de p) est un terme (resp une formule) de la forme $u[v]$ (resp $p[v]$) où $v \in S_i'$.

Définition 2 : Soit $\mathcal{F}$ un système fonctionnel, $x = (x_1,\dots,x_n) \in V_i$. La restriction de $\mathcal{F}$ à $L(x)$ est le système formel $\mathcal{F}_o(x) = (\mathcal{L}_o(x), F_o(x), \mathcal{X}_o(x), R_o)$ défini par :

$$\mathcal{L}_o(x) = L(x) \cup \{\equiv\} \cup \{(\ ,\ ),\ \neg,\ \supset\}$$

$$A_o(x) = \bigcup_{j=1}^{m} S_j(x) \equiv S_j(x)$$

$$F_o(x) = A_o(x) \cup \neg F_o(x) \cup (F_o(x) \supset F_o(x))$$

$$\mathcal{X}_o(x) = \mathrm{Prop}(A_o(x)) \cup Eg_o(x) \cup X_o(x)$$

où $Eg_o(x)$ (resp $X_o(x)$) est formé des exemplaires de formules de Eg (resp X) sur $\mathcal{L}_o(x)$.

$$R_o = \{\text{modus ponens}\}$$

On notera en particulier $\mathcal{F}_o$ la restriction de $\mathcal{F}$ à L.

Définition 3 : Une information de $\mathcal{F}_o$ est un sous ensemble de F_o contenant les théorèmes de $\mathcal{F}_o$ et stable par application de la règle de modus ponens.

Notations :

. Soit I une information de $\mathcal{F}$, Y un système d'axiomes de I. On note Y_o l'ensemble des exemplaires sans variable de formules de Y et I_o l'ensemble des formules sans variable de I.

. Soit $\mathcal{F}$ un système fonctionnel, $x \in V_i$ et $u \in S_i'$.

Soit $y = (y_1,\dots,y_m)$ les variables apparaissant dans u. On note $\mathcal{F}_o(u)$ le système formel $(\mathcal{L}_o(y), F_o(y), \mathcal{X}_o(u), R_o)$

où $\mathcal{X}_o(u) = \mathrm{Prop}(A_o(y)) \cup Eg_o(y) \cup \{p[u]\ ;\ p \in X_o(x)\}$

Enfin, si x,y sont des suites de variables sans point commun et u,v des suites de termes de type correspondant, on définit de manière similaire les systèmes $\mathcal{F}_o(x,y)$, $\mathcal{F}_o(x,v)$, $\mathcal{F}_o(u,v)$.

Théorème 1 : Soit $\mathcal{F}$ un système fonctionnel, p une formule sur $L(x)$. Alors, si $\vdash_{\mathcal{F}} p$,

1) $\vdash_{\mathcal{F}_o(x)} p$

2) $\vdash_{\mathcal{F}_o(u)} p[u]$ pour tout $u \in S_i'$.

Démonstration : par récurrence sur la longueur d'une démonstration de p

a) si $p \in \mathcal{X}$, $p \in \mathcal{X}_o(x)$ et $p[u] \in \mathcal{X}_o(u)$.

b) soit q une formule sur $L(x,y) = L \cup \{x_1,\dots,x_n\} \cup \{y_1,\dots,y_m\}$ telle que $\vdash_{\mathcal{F}} q$ et $\vdash_{\mathcal{F}} q \supset p$. Soit $v = (v_1,\dots,v_m)$ une suite de termes sans variable de même type que y. Alors, par récurrence,

$$\vdash_{\mathcal{F}_o(x,v)} q_y[v] \quad \text{et} \quad \vdash_{\mathcal{F}_o(x,v)} q_y[v] \supset p$$

Donc $\vdash_{\mathcal{F}_o(x,v)} p$ soit $\vdash_{\mathcal{F}_o(x)} p$

De plus si $u \in S_i'$, par récurrence,

$$\vdash_{\mathcal{F}_o(u,v)} q_{x,y}[u,v] \quad \text{et} \quad \vdash_{\mathcal{F}_o(u,v)} q_{x,y}[u,v] \supset p[u]$$

soit $\vdash_{\mathcal{F}_o(u)} p[u]$

Soit y une variable de type j, $v \in S_j(x)$ et q une formule sur $L(x,y)$ telle que $\vdash_{\mathcal{F}} q$ et $p = q_y[v]$

Par récurrence $\vdash_{\mathcal{F}_0(x,v)} q_y[v]$ donc $\vdash_{\mathcal{F}_0(x)} p$

De même, si $u \in S_i'$,

$$\vdash_{\mathcal{F}_0(u,v[u])} q_{x,y}[u,v[u]]$$

soit $$\vdash_{\mathcal{F}_0(u)} p\,[u] .$$

Remarque : En fait p peut être déduit d'un nombre fini d'axiomes de $\mathcal{F}_0(x)$ en utilisant la règle de modus ponens.

Corollaire 1 : Soit T (resp T_0) l'ensemble des théorèmes de $\mathcal{F}$ (resp $\mathcal{F}_0$). Alors $T_0 = T \cap F_0$.

Corollaire 2 : Soit I une information de $\mathcal{F}$, Y un système d'axiomes pour I. Alors I_0 est une information de $\mathcal{F}_0$ admettant Y_0 comme système d'axiomes.

2.1.2.- Théorème de la déduction

Définition 4 : Soit $\mathcal{F}$ un système formel (resp. une information), Y un ensemble de formules ; $\mathcal{F}[Y]$ est le système formel (resp. l'information) obtenu en ajoutant comme axiomes les formules de Y. Si $Y = \{p_1,\dots,p_n\}$, on notera $\mathcal{F}[Y] = \mathcal{F}[p_1,\dots,p_n]$.

Théorème 2.- Soit $\mathcal{F}$ un système fonctionnel, $p_1,\dots,p_n$ des formules sans variable, p une formule. Alors p est un théorème de $\mathcal{F}[p_1,\dots,p_n]$ si et seulement si $\vdash_{\mathcal{F}} p_1 \supset \dots \supset p_n \supset p$.

Démonstration :

Si $p_1 \supset \dots \supset p_n \supset p$ est un théorème de $\mathcal{F}$, il suit, en appliquant la règle de modus ponens, que p est un théorème de $\mathcal{F}[p_1,\dots,p_n]$. Réciproquement, il suffit de montrer que si p est un théorème de $\mathcal{F}[q]$, $q \supset p$ est un théorème de $\mathcal{F}$.

Raisonnons par récurrence sur la démonstration de p dans $\mathcal{F}[q]$

1) Si p est un axiome de $\mathcal{F}$, $q \supset p$ est un théorème de $\mathcal{F}$.

2) Si $p = q$, $q \supset p$ est un théorème de $\mathcal{F}$.

3) Si p est déduit de r et de $r \supset p$, par récurrence, $q \supset r$ et $q \supset (r \supset p)$ sont des théorèmes de $\mathcal{F}$. Or

$$(q \supset (r \supset p)) \supset (q \supset r) \supset (q \supset p)$$

est un axiome de $\mathcal{F}$ et, en appliquant deux fois la règle de modus ponens, $q \supset p$ est un théorème de $\mathcal{F}$.

4) Si r est un théorème de $\mathcal{F}[q]$ et si $p = r\,[u]$, par récurrence, $q \supset r$ est un théorème de $\mathcal{F}$. Donc $q\,[u] \supset r\,[u]$ est un théorème de $\mathcal{F}$ et, puisque q est sans variable, $q \supset p$ est un théorème de $\mathcal{F}$.

Corollaire : Soit I une information, Y un ensemble de formules, p une formule sans variable. $p \in I[Y]$ si et seulement si il existe des exemplaires $p_1,\dots,p_n$ de formules de Y, sans variable, tels que $p_1 \supset \dots \supset p_n \supset p \in I$.

Remarque : Si $p_1,\dots,p_n$ sont des formules comportant des variables, p peut être un théorème de $\mathcal{F}[p_1,\dots,p_n]$ sans que $\vdash_{\mathcal{F}} p_1 \supset \dots \supset p_n \supset p$.

Exemple : Soit $\mathcal{F}$ un système fonctionnel sans axiome propre et contenant deux symboles de constantes 0 et 1 et soit $p = x \equiv 0$ et $q = 1 \equiv 0$.

q est un théorème de $\mathcal{F}[p]$ mais $p \supset q$ n'est pas un théorème de $\mathcal{F}$.

2.1.3.- Règles du calcul des propositions

Le théorème de tautologie et la règle d'équivalence, démontrés en 0.4 s'appliquent aux systèmes fonctionnels.

2.1.4.- Propriétés de l'égalité

Voici tout d'abord un résultat simple dont la démonstration utilise les règles précédentes.

Proposition 1 : L'égalité est symétrique et transitive.

Démonstration : Pour $i = 1,\ldots,m$ soit x_i, y_i, z_i des variables de type i.

Symétrie : Soit $i \in [m]$. Il faut montrer que $x_i \equiv y_i \leftrightarrow y_i \equiv x_i$ est un théorème de $\mathcal{F}$. Or

$$x_i \equiv y_i \supset x_i \equiv z_i \supset y_i \equiv z_i \qquad \text{(Axiome } q_i\text{)}$$

En particulier :

$$x_i \equiv y_i \supset x_i \equiv x_i \supset y_i \equiv x_i \qquad \text{(règle de substitution)}$$

Comme $x_i \equiv x_i$

$$x_i \equiv y_i \supset y_i \equiv x_i \qquad \text{(Théorème de tautologie)}$$

Donc, également

$$y_i \equiv x_i \supset x_i \equiv y_i \qquad \text{(règle de substitution)}$$

et $$x_i \equiv y_i \leftrightarrow y_i \equiv x_i \qquad \text{(Théorème de tautologie)}$$

Transitivité :

$$x_i \equiv y_i \supset x_i \equiv z_i \supset y_i \equiv z_i \qquad \text{(Axiome } q_i\text{)}$$

donc

$$x_i \equiv y_i \supset z_i \equiv x_i \supset z_i \equiv y_i \qquad \text{(règle d'équivalence et symétrie de l'égalité)}$$

ou $$z_i \equiv x_i \supset x_i \equiv y_i \supset z_i \equiv y_i \qquad \text{(théorème de tautologie)}$$

soit

$$x_i \equiv y_i \supset y_i \equiv z_i \supset x_i \equiv z_i \qquad \text{(règle de substitution en permutant } x_i, y_i, z_i\text{)}$$

Théorème 3.- Soient $u_1,\ldots,u_n$, $u'_1,\ldots,u'_n$, v, v' des termes et p, p' des formules tels que v' (resp p') soit obtenu en remplaçant dans v (resp p) certaines occurrences de $u_1,\ldots,u_n$, respectivement par $u'_1,\ldots,u'_n$. Alors

$$u_1 \equiv u'_1,\ldots,u_n \equiv u'_n \vdash v = v'$$

et

$$u_1 \equiv u'_1,\ldots,u_n \equiv u'_n \vdash p \leftrightarrow p'$$

La démonstration est immédiate, par récurrence sur la longueur de p (resp v).

2.2.- Informations consistantes, complètes

Nous allons étudier maintenant deux types d'informations particulièrement importants. On dira qu'une information est consistante si toute formule n'est pas un théorème de cette information. Il est clair que l'information inconsistante ne présente pas d'intérêt. La notion d'information consistante se révèle également intéressante dans le cadre des équations sur une structure. Appelons équation sur $\mathcal{S}$ la donnée d'un symbole f, n'appartenant pas à l'alphabet de $\mathcal{S}$, et d'une formule égalitaire sur cet alphabet étendu. Par exemple, considérons la structure $\mathbb{Z}$ des entiers relatifs et un symbole a, 0-aire, avec l'équation

$$a^2 + a + 1 \equiv 0$$

L'information $\mathbb{Z}'$ obtenue en ajoutant ce symbole et cet axiome est inconsistante. En effet, on peut montrer dans $\mathbb{Z}$ le théorème

$$x^2 + x + 1 \not\equiv 0$$

qui, par substitution, donne

$$a^2 + a + 1 \not\equiv 0$$

L'inconsistance de $\mathbb{Z}'$ exprime le fait que l'équation

$$x^2 + x + 1 \equiv 0$$

ne possède pas de solution dans $\mathbb{Z}$.

Considérons maintenant l'équation

$$a^2 - 3a + 2 \equiv 0$$

Cette fois-ci l'équation admet deux solutions (1 et 2). Ceci se traduit formellement par le fait que l'on peut démontrer dans $\mathbb{Z}'$ le théorème

$$a \equiv 1 \vee a \equiv 2$$

Appelons complète une information consistante I telle que, pour toute formule p sans variable, $p \in I$ ou $\neg p \in I$. $\mathbb{Z}'$ n'est pas complète ; en effet, on ne peut démontrer dans $\mathbb{Z}'$ ni $a \equiv 1$ ni $a \not\equiv 1$. On obtient par contre une information complète en ajoutant soit la formule $a \equiv 1$ soit la formule $a \equiv 2$. Ainsi une information est incomplète si elle possède plusieurs extensions consistantes distinctes.

2.2.1.- Définitions

Proposition 2 : Soit I une information. Les propriétés suivantes sont équivalentes.

i) $I \neq F$

ii) pour toute formule atomique a de A, $a \in I \Longrightarrow \neg a \notin I$

iii) pour toute formule p de F, $p \in I \Longrightarrow \neg p \notin I$

Démonstration :

i ⇒ ii Soit a une formule atomique telle que a et $\neg a$ appartiennent à I ; la formule $a \wedge \neg a$ appartient à I. Toute formule p est conséquence de $a \wedge \neg a$ (puisque $\tilde{V}(a \wedge \neg a) =$ faux pour toute fonction de vérité). Donc toute formule est un théorème de I (théorème de tautologie).

ii ⇒ iii Raisonnons par récurrence sur la longueur de p.

Si $p = \neg q$, $\neg p = \neg\neg q$. Si $\neg p \in I$, $q \in I$; donc, par récurrence, $\neg q \notin I$. Donc $p \in I \Longrightarrow \neg p \notin I$.

Si $p = q \supset r$, $\neg p = \neg(q \supset r)$. Si $\neg p \in I$, $q \in I$ et $\neg r \in I$.

Donc, par récurrence, $r \notin I$ et donc $q \supset r \notin I$. D'où $p \in I \Rightarrow \neg p \in I$.

iii ⇒ i évident.

Définition 5 : Une information I est consistante si elle vérifie les conditions équivalentes de la proposition 2.

Soit a_0 une formule atomique sans variable et $q_0 = a_0 \wedge \neg a_0$. Alors

1) pour toute formule p , $p \supset q_0$ est équivalent à $\neg p$.

2) une information I est consistante si et seulement si $q_0 \notin I$.

Nous utiliserons dans la suite la formule q_0.

Proposition 3 : Soit I une information et p une formule sans variable, p appartient à I si et seulement si $I\,[\neg p]$ est inconsistante.

Démonstration : Si $p \in I$, p et $\neg p$ sont des théorèmes de $I\,[\neg p]$. Réciproquement si $I(\neg p)$ est inconsistante, $q_0 \in I\,[\neg p]$. Par le théorème de la déduction, $\neg p \supset q_0 \in I$; donc $\neg\neg p \in I$ soit encore $p \in I$.

Si p possède des variables, le résultat précédent est en défaut. Considérons la formule $p = (2x + 3 \equiv x + 4)$ dans la structure $\mathcal{N}$ des entiers. p n'est pas un théorème de $\mathcal{N}$, bien que $\mathcal{N}(\neg p)$ soit inconsistante. (En effet, $2 + 3 \equiv 1 + 4$ est un théorème de $\mathcal{N}$).

Remarque : Soit I une information et I_0 l'ensemble des formules sans variables de I. Si Y est un système d'axiomes de I, soit Y_0 l'ensemble des exemplaires sans variables de formules de Y. Nous avons vu (théorème 1) que I_0 était l'ensemble des théorèmes déduits des axiomes de Y_0 en utilisant uniquement le modus ponens. On dira que I_0 est inconsistante si $I_0 = F_0$

Corollaire : Soit I une information et p une formule sans variable. p appartient à I_0 si et seulement si $I_0\,[\neg p]$ est inconsistante.

Proposition 4 : Soit I une information et p une formule sur L(x). $I[p]$ est consistante si et seulement si, pour tout ensemble fini Γ d'exemplaires de p sans variable, $I\,[\Gamma]$ est consistante.

Démonstration : La condition est évidemment nécessaire par application de la règle de substitution. Réciproquement, supposons $I\,[p]$ inconsistante. $q_0 \in I[p]$ et, puisque q_0 est une formule sans variable, il existe d'après le théorème 1, un ensemble fini Γ d'exemplaires de p sans variable tel que $q_0 \in I\,[\Gamma]$, donc tel que $I\,[\Gamma]$ soit inconsistante.

Proposition 5 : Soit I une information consistante. Les propriétés suivantes sont équivalentes :

i) I_0 est maximale dans l'ensemble des informations consistantes de $\mathcal{F}_0$

ii) pour toute formule atomique a sans variable, $a \notin I \Rightarrow \neg a \in I$

iii) pour toute formule p sans variable, $p \notin I \Rightarrow \neg p \in I$.

Démonstration :

i ⇒ ii Si $a \notin I$, $I\,[a]$ est inconsistante. Donc $\neg a \in I$.

ii ⇒ iii Raisonnons par récurrence sur la longueur de p

Si $p = \neg q$, $\neg p = \neg\neg q$. Si $\neg p \notin I$, $q \notin I$. Donc, par récurrence, $\neg q \in I$. D'où $p \notin I \Rightarrow \neg p \in I$.

Si $p = q \supset r$, $\neg p = \neg(q \supset r)$. Si $\neg p \notin I$, $q \notin I$ ou $\neg r \notin I$. Donc, par récurrence, $\neg q \in I$ ou $r \in I$. Donc $p \in I$.

iii ⇒ i Soit I'_0 une information contenant strictement I_0, p une formule appartenant à $I'_0 - I_0$. Alors, $\neg p \in I_0 \subset I'_0$. Donc I'_0 est inconsistante.

Définition 6 : Une information consistante I est complète si elle vérifie les propriétés équivalentes de la proposition 5.

Définition 7 : Soit I, I' des informations consistantes de $\mathcal{F}$. I' est une extension maximale de I si I' contient I et si I' est maximale parmi les informations consistantes de $\mathcal{F}$.

Remarque : Si I est une information maximale parmi les informations consistantes de $\mathcal{F}$, I_0 est à fortiori maximale parmi les informations consistantes de $\mathcal{F}_0$. Donc, I est complète.

Proposition 6 : Soit I une information consistante. I admet une extension maximale I'.

Démonstration : Il suffit de montrer que l'ensemble des informations consistantes contenant I admet un élément maximal. Nous utilisons pour cela le lemme suivant de la théorie des ensembles.

Soit E un ensemble et $\mathcal{J}$ une classe de sous ensembles de E. On dit que $\mathcal{J}$ est de caractère fini si, pour tout sous-ensemble A de E, A est dans $\mathcal{J}$ si et seulement si tout sous ensemble fini de A est dans $\mathcal{J}$.

Lemme 1.- (Terchmüller - Tukey) Si $\mathcal{J}$ est une classe non vide de sous ensemble de E qui est de caractère fini, $\mathcal{J}$ admet un élément maximal.

Revenons à la démonstration de la proposition. Soit $\mathcal{J}$ la classe de tous les sous ensembles Γ de F tels que $I[\Gamma]$ soit consistante. $\mathcal{J}$ contient l'ensemble vide. Si Γ est dans $\mathcal{J}$, tout sous ensemble fini de Γ est dans $\mathcal{J}$. Réciproquement si Γ n'est pas dans $\mathcal{J}$, $q_0 \in I[\Gamma]$. Il existe un sous ensemble fini Γ' de Γ tel que $q_0 \in I[\Gamma']$ donc tel que $I[\Gamma']$ soit inconsistante. $\mathcal{J}$ est donc de caractère fini. Soit Γ_0 un élément maximal de $\mathcal{J}$. $I' = I(\Gamma_0)$ est maximale parmi les informations

consistantes contenant I.

Une information complète n'est pas nécessairement maximale parmi les informations consistantes de $\mathcal{F}$. On a cependant le résultat suivant.

Proposition 7 : Soit I une information consistante. I est complète si et seulement si elle admet une extension maximale unique.

Démonstration : Si I n'est pas complète, soit p une formule sans variable telle que p et $\neg p$ n'appartiennent pas à I. Alors $I[p]$ et $I[\neg p]$ admettent des extensions maximales distinctes.

Réciproquement supposons I complète. Soient I_1, I_2 deux informations consistantes maximales contenant I et soit $p \in I_2$. Pour tout exemplaire p' de p, sans variable, $p' \in I_2$. Donc $\neg p' \notin I_2$ et, puisque I est complète, $p' \in I \subset I_1$. D'après la proposition 4, $I_1[p]$ est donc consistante et $p \in I_1$ puisque I_1 est maximale. D'où $I_2 \subset I_1$ et, puisque I_2 est maximale, $I_2 = I_1$.

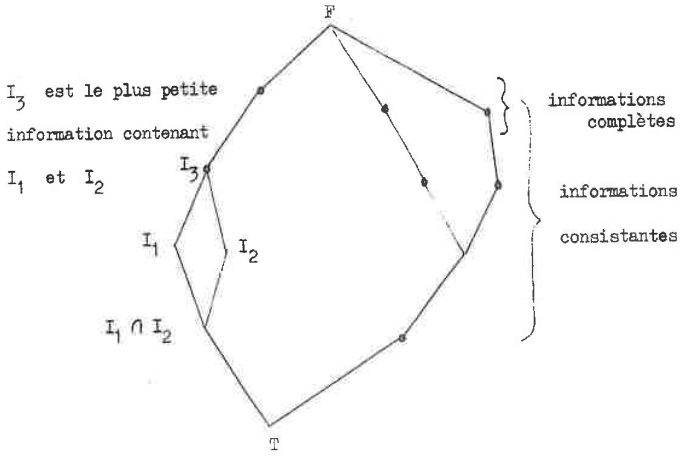
Remarque : Il ressort de la démonstration précédente que I est maximale parmi les informations consistantes de $\mathcal{F}$ si et seulement si I est complète et si, pour toute formule p de F,

$p \in I \iff p' \in I$, pour tout exemplaire p' de p sans variable.

Structure de l'ensemble des informations d'un système formel

L'ensemble $\mathcal{J}(\mathcal{F})$ est ordonné par la relation d'inclusion. Pour cet ordre, $\mathcal{J}(\mathcal{F})$ admet un plus petit élément, l'ensemble T des théorèmes, et un plus grand élément, l'ensemble F des formules. Si (I_λ) est une famille d'informations, $\bigcap_{\lambda \in \Lambda} I_\lambda$ est encore une information qui est la borne

inférieure de la famille (I_λ). De même, il existe une plus petite information contenant tous les I_λ. $\mathcal{J}(\mathcal{F})$ est donc un treillis complet (SCOTT). Enfin, si (I_λ) est une famille totalement ordonnée, $\bigcup_\lambda I_\lambda$ est encore une information qui est la borne supérieure de la famille (I_λ). $\mathcal{J}(\mathcal{F})$ est donc un ensemble complètement inductif (cf 4.2). Nous pouvons résumer cette étude dans le diagramme suivant.



2.2.2.- Interprétations d'une information consistante

Nous démontrons dans ce paragraphe le résultat fondamental suivant

Une information est consistante si et seulement si elle admet une interprétation.

et son corollaire

Soit I une information et p une formule sans variable. p est un théorème de I si et seulement si p est valide dans I,

(c'est-à-dire vérifie

$\tilde{r}(p) =$ vrai pour toute interprétation r de I).

Ces résultats nous serviront souvent par la suite. Il est en effet fort difficile de montrer la consistance d'une information par des méthodes directes.

Définition 8 : Soit $\sim$ une relation sur $S = \bigcup_{j=1}^{m} S_j$ telle que

$$u \sim v \implies \exists j \in [m] \ (u \in S_j \text{ et } v \in S_j) \qquad (1)$$

On dit que $\sim$ est compatible avec la composition des symboles fonctionnels si, pour tout $i \in [m]^*$, tout $f \in L$ tel que source$(f) = i$:

$$\forall u, v \in S_i \ , \ (u_1 \sim v_1 \text{ et } \ldots \text{ et } u_n \sim v_n \implies fu \sim fv)$$

Proposition 8 : Soit $\sim$ une relation d'équivalence sur S, compatible avec la composition des symboles fonctionnels. Pour chaque $j \in [m]$, soit $E_j = S_j/\sim$ le quotient de S_j par $\sim$. Il existe une interprétation stricte $R = (E,r)$ de S, unique, telle que $E = (E_1, \ldots, E_m)$ et que, pour tout $u \in S$, $\hat{r}(u)$ soit la classe d'équivalence $\dot{u}$ de u par $\sim$.

Démonstration :

Existence : Soit $f \in L$ tel que source$(f) = i$. Soit $x = (x_1, \ldots, x_n) \in E_i$. Posons

$$r(f)(x) = \dot{fu}$$

où $u = (u_1, \ldots, u_n)$ est tel que $x_i = \dot{u}_i$ pour $i \in [1,n]$. Puisque $\sim$ est compatible avec la composition des schémas, la définition de $r(f)(x)$ ne dépend pas du choix de u. On montre alors par récurrence sur u que $\hat{r}(u) = \dot{u}$.

Unicité : Soit $f \in L$, $x \in E_i$ et u comme ci-dessous. Par définition de $\hat{r}$

$$r(f)(x) = \hat{r}(fu) = \widehat{\dot{fu}}$$

Définition 9 : L'interprétation définie dans la proposition précédente est l'interprétation canonique associée à $\sim$.

Remarque : Soit $u,v \in S$ $\tilde{r}(u \equiv v) = \underline{vrai} \iff u \sim v$.

Proposition 9 : Soit I une information complète, $R = (E,r)$ une interprétation stricte de S. R est une interprétation de I si et seulement si, pour tout couple (u,v) de termes de même type sans variable,

$$u \equiv v \in I \iff \tilde{r}(u \equiv v) = \underline{vrai}$$

Démonstration : La condition est nécessaire. Réciproquement, montrons par récurrence sur p que $p \in I \iff \tilde{r}(p) = \underline{vrai}$.

Si p est atomique c'est l'hypothèse.

Si $p = \neg q$, $\tilde{r}(p) = \underline{vrai} \iff \tilde{r}(q) = \underline{faux} \iff q \notin I \iff p \in I$

Si $p = q \supset r$, $\tilde{r}(p) = \underline{vrai} \iff \tilde{r}(q) = \underline{faux}$ ou $\tilde{r}(r) = \underline{vrai}$

$\iff q \notin I$ ou $r \in I \iff q \supset r \in I$

Théorème 4 : Une information I est consistante si et seulement si elle admet une interprétation.

Démonstration : Soit $R = (E,r)$ une interprétation de I . $\tilde{r}(q_o) = \underline{faux}$. Donc $q_o \notin I$ et I est consistante.

Réciproquement, soit I' une extension maximale de I (proposition 6). La relation

$$u \sim v \iff u \equiv v \in I'$$

est une relation d'équivalence compatible avec la composition des symboles fonctionnels (propriétés de l'égalité $\equiv$). Soit $R = (E,r)$ l'interprétation canonique associée à $\sim$. Il résulte clairement de la remarque et de la proposition précédente que R est une interprétation de I' , donc de I.

Définition 10 : Soit I une information et p une formule. On dit que p est valide dans I si $\tilde{r}(p) = \underline{vrai}$ pour toute interprétation de I.

Corollaire : Soit p une formule sans variable. p est un théorème de I si et seulement si p est valide dans I.

Démonstration : Si $p \in I$ et $R = (E,r)$ est une interprétation de I, $\tilde{r}(p) = \underline{vrai}$. Inversement, si $p \notin I$, $I\,[\neg p]$ est consistante. Soit $R = (E,r)$ une interprétation de $I\,[\neg p]$ $\tilde{r}(\neg p) = \underline{vrai}$; donc $\tilde{r}(p) = \underline{faux}$ et p n'est pas valide.

Remarque : Pour montrer que $p \notin I$, il suffit donc de construire une interprétation $R = (E,r)$ de I telle que $\tilde{r}(p) = \underline{faux}$ (cf 3.3).

2.2.3.- Interprétation d'une information complète

Définition 11 : Soit I une information. Deux interprétations $R = (E,r)$ et $R' = (E',r')$ de I sont équivalentes si $\tilde{r}(p) = \tilde{r}'(p)$ pour toute formule p de F.

Lemme 2 : R et R' sont équivalentes si et seulement si, pour tout couple (u,v) d'éléments de S, de même type,

$$\hat{r}(u) = \hat{r}(v) \iff \hat{r}'(u) = \hat{r}'(v)$$

Il existe donc une bijection φ de $\bigcup_{i=1}^{m} E_i$ sur $\bigcup_{i=1}^{m} E'_i$, unique telle

que, pour tout $u \in S$,

$$\varphi(\hat{\tilde{r}}(u)) = \hat{\tilde{r}}'(u)$$

Théorème 5 : Une information I est complète si et seulement si elle est consistante et si toutes les interprétations de I sont équivalentes.

Démonstration : Si I est complète et si R est une interprétation de I

$$u \equiv v \in I \iff \hat{\tilde{r}}(u) \equiv \hat{\tilde{r}}(v)$$

Réciproquement, si I n'est pas complète, il existe $p \in F$ tel que $I_1 = I[p]$ et $I_2 = I[\neg p]$ soient des informations consistantes. Soit R_1 une interprétation de I_1, R_2 une interprétation de I_2. R_1, R_2 sont deux interprétations non équivalentes de I. En effet :

$$\tilde{\tilde{r}}_1(p) = \underline{\text{vrai}} \qquad \tilde{\tilde{r}}_2(p) = \underline{\text{faux}}$$

Corollaire : Soit I une information complète ; une formule sans variable p est un théorème de I si et seulement si $\tilde{\tilde{r}}(p) =$ vrai pour une interprétation de I.

2.3.- Relation entre l'étude des systèmes fonctionnels et le calcul des prédicats

2.3.1.- Introduction

Nous nous posons ici la question de savoir si l'on obtient une théorie plus forte en utilisant des formules quantifiées dans notre système formel. Sans apporter une réponse complète, nous donnons ici plusieurs résultats qui nous ont incités à ne pas utiliser les quantificateurs.

Nous utilisons en fait implicitement des formules quantifiées. Ainsi la formule $x \equiv y \supset y \equiv x$ est-elle équivalente à la formule $\forall x \forall y \quad x \equiv y \supset y \equiv x$.

Plus généralement, toute formule $p[x_1,\ldots,x_n]$ est équivalente à $\forall x_1 \ldots \forall x_n \; p[x_1,\ldots,x_n]$, formule que nous noterons $\forall x \; p[x]$.

Bien entendu, on n'obtient ainsi que des formules universellement quantifiées. Si l'on prend la négation d'une telle formule ; soit

$$\neg \forall x \; p[x] \qquad (1)$$

on obtient une formule qui n'est pas équivalente à $\forall x \; \neg p[x]$ mais à $\exists x \neg p[x]$. Cette dernière formule n'est d'ailleurs qu'une réécriture de (1).

Examinons plus en détail l'usage qui est fait des quantificateurs.

1) Dans les axiomes d'une théorie il est fréquent d'affirmer l'existence d'objets vérifiant une propriété donnée. Dans l'axiomatique des corps on a, par exemple, la formule

$$\exists y [x \neq 0 \supset x * y \equiv 1] \qquad (2)$$

définissant l'inverse de tout nombre non nul (ou plutôt l'existence d'un inverse). Dans le cadre des structures d'information, il est intéressant de pouvoir accéder à cet inverse. Aussi introduit-on un symbole unaire inverse défini par la formule

$$x \neq 0 \supset x * \text{inverse}(x) \equiv 1 \qquad (3)$$

Notons que le terme inverse (0) n'a pas de valeur précise.

On appelle théories ouvertes les théories dont les axiomes propres sont sans quantificateurs. Etant donné un système fonctionnel, on obtient une théorie ouverte en ajoutant les symboles $\exists$ et $\forall$ ainsi que les axiomes et règles d'inférence du calcul des prédicats.

2) Dans le cadre d'une telle théorie, on est alors amené à s'intéresser aux théorèmes de la forme

$$\exists y \; p[y] \qquad (4)$$

où p est une formule quelconque.

Exemple : Soit la structure $\mathcal{Q}$ des rationnels. L'alphabet de $\mathcal{Q}$ est formé des symboles 0,1 (0-aires), inverse et - (unaires), + et * (binaires). Les axiomes de $\mathcal{Q}$ sont par ailleurs classiques. On se pose alors le problème suivant : étant donnés deux rationnels x,y existe-t-il des rationnels z,t tels que

$$\begin{cases} x * z + y * t = x \\ x * t + (-y) * z = y \quad ? \end{cases}$$

Autrement dit, la formule

$$\exists z \exists t \left[x*z + y*t \equiv x \wedge x*t + (-y)*z \equiv y \right]$$

est-elle un théorème ? Si oui, peut-on calculer z et t en fonction de x,y [ou au moins un z et un t, s'il n'y a pas unicité] ? La réponse à ces questions est positive : les expressions de z et t sont données, pour (x,y) différent de $(0,0)$, par les formules de Cramer. Pour $x = 0$ et $y = 0$, tout couple (z,t) est une solution.

De manière générale, nous verrons que, dans les théories ouvertes, les problèmes d'existence et de calcul d'une solution sont étroitement liés. En particulier, si p est une formule contenant des variables $x_1,\ldots,x_p$, $y_1,\ldots,y_n$ sans quantificateur, la formule $\exists y \; p[x,y]$ est un théorème si et seulement si l'on peut trouver des suites de termes $u_1[x],\ldots,u_m[x]$ tel que $p[x,u_1[x]] \vee \ldots \vee p[x,u_m[x]]$ soit un théorème (autrement dit si et seulement si l'un des u_k est une solution).

Notons, pour finir, que nous pouvons reformuler les problèmes du type $\exists y \; p[x,y]$ différemment. Introduisons des symboles $f_1,\ldots,f_n$ p-aires tels que $\underline{\text{source}}\,(f_k) = \underline{\text{type}}(x_1,\ldots,x_p)$ et $\underline{\text{but}}(f_k) = \underline{\text{type}}(y_k)$ et considérons la formule p' obtenue à partir de p en remplaçant chaque y_k par $f_k\, x_1 \ldots x_p$.

Exemple : Dans le système d'équations précédent remplaçons z par $f(x,y)$ et t par $g(x,y)$. On obtient la formule suivante, "définissant" f,g :

$$x*f(x,y) + y*g(x,y) \equiv x \;\wedge\; x*g(x,y) + (-y)*f(x,y) \equiv y$$

On obtient alors un problème dont les inconnues sont $f_1,\ldots,f_n$ et dont l'énoncé est la formule p'. Nous étudions ces problèmes dans les chapitres suivants.

2.3.2.- Théories du premier ordre

Définition 12 : Un langage du premier ordre est un ensemble F de formules défini par les éléments suivants

1) Les symboles de F sont

- les symboles d'un système fonctionnel
- le symbole $\exists$

2) Les termes et les formules atomiques de F sont définis comme pour les systèmes fonctionnels.

3) L'ensemble F des formules est l'unique solution de l'équation

$$F = A \cup \neg F \cup (F \supset F) \cup \bigcup_{x \in V} \exists x F$$

où A représente l'ensemble des formules atomiques.

Outre les abréviations introduites au chapitre 1, on note

$\forall x\ p$ au lieu de $\neg \exists x \neg p$

On suppose que l'ensemble V des variables est muni d'un ordre total arbitraire ou ordre alphabétique.

Définition 13 : Une occurrence de x dans p est liée dans p si elle apparaît dans une sous formule de p de la forme $\exists x\ q$. Sinon, elle est libre dans p. x est liée (resp libre) dans p si une occurrence de x est liée (resp libre) dans p.

Soit x une variable de type i, v un terme de même type. Si u est un terme on note $u_x[v]$ le terme obtenu en remplaçant chaque occurrence de x dans u par v.

Si p est une formule, on note $p_x[v]$ la formule obtenue en remplaçant chaque occurrence libre de x dans p par v. On dira que x peut être remplacé par v dans p si, pour toute variable libre y de v, aucune sous formule de p de la forme $\exists y\ q$ ne contient d'occurrence de x libre dans p. Par exemple, on ne peut remplacer x par y+1 dans la formule

$$\exists y\ (x \equiv 2\ y)$$

Nous convenons d'écrire $p_x[v]$ seulement si x peut être remplacé par v dans p.

On obtient des définitions similaires pour $x \in V_i$ et $u \in S_i^j$ $(i \in [m]^*)$.

Définition 14 : On appelle formules élémentaires les éléments de l'ensemble E défini par

$$E = A \cup \bigcup_{x \in V} \exists x F$$

Remarque : $F = E \cup \neg F \cup (F \supset F)$

A chaque fonction de vérité V sur E, on peut donc associer une application $\tilde{V}$ de F dans $\{\text{vrai}, \text{faux}\}$.

Introduisons les ensembles d'axiomes suivants :

. Prop(E) et Eg qui ont été définis respectivement en 0.4 et 1.2

. Subs $= \{p_x[u] \supset \exists x\ p \ ;\ p \in F$ et x peut être remplacé par u dans $p\}$

et la règle suivante, définissant l'utilisation de $\exists$.

$\exists$ - introduction : $p \supset q \vdash \exists x\ p \supset q$ si x n'est pas libre dans q.

Définition 15 : Une théorie du premier ordre est un système formel

$\mathcal{F} = (\mathcal{L}, F, \mathcal{X}, R)$ tel que

. F soit un langage du premier ordre, d'alphabet $\mathcal{L}$

. $\mathcal{X}$ = Prop(E) $\cup$ Eg $\cup$ Subs $\cup$ X

. R = {modus ponens, $\exists$-introduction}.

Remarque : Nous obtiendrons la règle de substitution définie dans les systèmes fonctionnels comme une règle dérivée.

Définition 16 : Soit p une formule ; p est ouverte si elle est sans quantificateur ; p est fermée si aucune variable n'est libre dans p.

Enonçons sans démonstration les principales règles concernant les théories du premier ordre.

$\forall$-introduction : Si $\vdash p \supset q$ et si x n'est pas libre dans p, $\vdash p \supset \forall x q$

Règle de généralisation : Si $\vdash p$, alors $\vdash \forall x p$

Règle de substitution : Si $\vdash p$, alors $\vdash p_{x_1,\ldots,x_n}[u_1,\ldots,u_n]$.

Définition 17 : Soit p une formule, $x_1,\ldots,x_n$ les variables libres dans p (rangées en ordre alphabétique). La formule $\forall x_1 \ldots \forall x_n$ est appelée la fermeture de p.

Théorème de fermeture : Si p' est la fermeture de p, $\vdash p \iff \vdash p'$.

Définition 18 : On dit que p' est une variante de p si p' peut être obtenue à partir de p en effectuant une suite de remplacements du type : remplacer $\exists x q$ par $\exists y\, q_x[y]$ où y n'est pas libre dans q.

Théorème de la variante : Si p' est une variante de p, $\vdash p \leftrightarrow p'$.

Définition 19 : p est une tautologie si, pour toute fonction de vérité sur l'ensemble E des formules élémentaires, $\tilde{V}(p) = \text{vrai}$. p est conséquence tautologique de $p_1,\ldots,p_n$ si, pour toute fonction de vérité V sur E

$$\tilde{V}(p_1) = \ldots = \tilde{V}(p_n) = \text{vrai} \implies \tilde{V}(p) = \text{vrai}$$

Théorème de tautologie : Si q est conséquence tautologique de $p_1,\ldots,p_n$ et si $\vdash p_1,\ldots, \vdash p_n$, alors $\vdash q$. En particulier, toute tautologie est un théorème.

Nous montrons maintenant que toute formule est équivalente à une formule dans laquelle tous les quantificateurs sont regroupés à l'extérieur. Cette forme canonique est utilisée, dans les systèmes de preuve de théorèmes.

Forme prénexe d'une formule

Définition 20 : Une formule p est en forme prénexe si elle a la forme $Q x_1 \ldots Q x_n q$ où chaque $Q x_i$ est, soit $\forall x_i$, soit $\exists x_i$ et où q est une formule ouverte.

Proposition 10.- Soit p une formule, il existe une formule p', en forme prénexe telle que $\vdash p \leftrightarrow p'$.

Donnons simplement un exemple illustrant les opérations nécessaires pour convertir une formule en forme prénexe

$$\exists x(x \equiv y) \supset \exists x(x \equiv 0 \vee \neg \exists y(y < 0))$$

$$\exists x(x \equiv y) \supset \exists z(z \equiv 0 \vee \neg \exists w(w < 0))$$

$$\exists x(x \equiv y) \supset \exists z(z \equiv 0 \vee \forall w \neg(w < 0))$$

$$\exists x(x \equiv y) \supset \exists z\, \forall w(z \equiv 0 \vee \neg(w < 0))$$

$$\forall x\, \exists z\, \forall w(x \equiv y \supset z \equiv 0 \vee \neg(w < 0))$$

Enfin nous généralisons les résultats sur la consistance obtenus au paragraphe 2.2.

Une théorie est consistante si toute formule n'est pas un théorème. On a alors la proposition suivante.

Proposition 11.- Soit p' la fermeture de p. p est un théorème de $\mathcal{F}$ si et seulement si $\mathcal{F}[\neg p']$ est inconsistante.

En particulier si q est une formule fermée existentielle

$$q = \exists x_1 \ldots \exists x_n\, p$$

q est un théorème de $\mathcal{F}$ si et seulement si $\mathcal{F}[\neg p]$ est inconsistante.

En effet $\neg q' = \forall x_1 \ldots \forall x_n \neg p$ et $\mathcal{F}[\neg q']$ est équivalent à $\mathcal{F}[\neg p]$ (théorème de fermeture).

2.3.3.- Théories ouvertes - Théorème de consistance

Une théorie est ouverte si tous ses axiomes propres sont des formules ouvertes.

On dira qu'une formule est une quasi-tautologie si c'est une conséquence tautologique d'exemplaires des axiomes d'égalité.

Remarque : Soit $p_1,\ldots,p_n$, q des formules. Si p peut être déduit de $p_1,\ldots,p_n$ par une démonstration utilisant uniquement la règle de tautologie (2.1.3) et la règle d'égalité (2.1.4), alors $p_1 \supset \ldots \supset p_n \supset q$ est une quasi-tautologie.

Nous pouvons alors énoncer le résultat suivant du à Hilbert et Ackermann.

Théorème de Consistance : Une théorie $\mathcal{F}$ ouverte est inconsistante si et seulement si il existe une suite $p_1,\ldots,p_n$ d'exemplaires des axiomes propres de $\mathcal{F}$ telle que

$$\neg p_1 \vee \ldots \vee \neg p_n$$

soit une quasi-tautologie.

La démonstration de ce théorème est un peu longue. On pourra la trouver, par exemple, dans (SHOENFIELD). On peut dériver de cette démonstration un algorithme construisant une suite $p_1,\ldots,p_n$ à partir de la preuve d'un théorème de la forme $u \neq u$.

Conséquence du théorème de consistance

Soit $y = (y_1,\ldots,y_n) \in V_i$ une suite de variables, $p[y]$ une formule ouverte et $u_1,\ldots,u_m$ des n-uplets de termes appartenant à S_i. Si

$$p[u_1] \vee \ldots \vee p[w_m]$$

est un théorème de $\mathcal{F}$, alors

$$\exists y\, p[y]$$

est également un théorème de $\mathcal{F}$.

En effet, d'après les axiomes de substitution

$$\vdash p[u_k] \supset \exists y\, p[y] \quad \text{pour } k = 1,\ldots,m.$$

Donc $(p[u_1] \vee \ldots \vee p[u_m]) \supset \exists y\, p[y]$ est également un théorème et $\exists y\, p[y]$ est un théorème.

Réciproquement

Théorème 6.- Soit $\mathcal{F}$ une théorie ouverte et $q = \exists y\, p[y]$ une formule existentielle fermée. Si q est un théorème de $\mathcal{F}$, il existe des exemplaires $p_1,\ldots p_k$ des axiomes non logiques de $\mathcal{F}$, et des n-uplets $u_1,\ldots,u_m$ de S_i tels que

$$p_1 \supset \ldots \supset p_k \supset (p[u_1] \vee \ldots \vee p[u_m])$$

soit une quasi-tautologie.

Démonstration : Si q est un théorème de $\mathcal{F}$, $\mathcal{F}[\neg p]$ est inconsistante (cf proposition 11) ; puisque $\mathcal{F}[\neg p]$ est une théorie ouverte, il existe des exemplaires $p_1,\ldots p_\ell$ des axiomes non logiques tels que

$$\neg p_1 \vee \ldots \vee \neg p_\ell$$

soit une quasi-tautologie. On peut toujours supposer que $p_1,\ldots p_k$ sont des exemplaires d'axiomes de $\mathcal{F}$ et que $p_{k+1},\ldots p_\ell$ sont des exemplaires de $\neg p$. Il existe donc $u_1,\ldots u_m$ tels que

$$\neg p_1 \vee \ldots \vee \neg p_k \vee p[u_1] \ldots \vee p[u_m]$$

soit une quasi-tautologie, donc tels que

$$p_1 \supset \ldots \supset p_k \supset (p[u_1] \vee \ldots \vee p[u_m])$$

soit une quasi-tautologie.

Remarques :

1) Puisque $p_1,\ldots,p_k$ sont des exemplaires des axiomes propres de $\mathcal{F}$, il vient immédiatement que $p[u_1] \vee \ldots \vee p[u_m]$ est un théorème de $\mathcal{F}$.

2) On peut préciser ce résultat lorsque la théorie est complète, dans le sens suivant. On dit qu'une théorie $\mathcal{F}$ est complète si $\mathcal{F}$ est consistante et si, pour toute formule p sans variable, p ou $\neg p$ est un théorème de $\mathcal{F}$. Alors, si $\mathcal{F}$ est complète, il existe u tel que $p[u]$ soit un théorème de $\mathcal{F}$. En effet pour chaque j de $[m]$, $p[u_j]$ ou $\neg p[u_j]$ est un théorème. Si $\neg p[u_j]$ est un théorème pour chaque j de $[m]$, $\neg[p[u_1] \vee \ldots \vee \neg p[u_m]]$ est aussi un théorème, ce qui contredit la remarque précédente.

3) Il suit de la remarque suivant le théorème de consistance qu'il existe un algorithme permettant de déduire de la démonstration du théorème $\exists y\, p[y]$ une suite de termes $u_1,\ldots,u_m$ tels que $p[u_1] \vee \ldots \vee p[u_m]$ soit un théorème.

Ces remarques illustrent le caractère constructif des théories ouvertes. Dans ce type de théories, démontrer un théorème d'existence c'est calculer une solution du problème. Ce point est à la base des systèmes de preuves de théorèmes. Nous donnons maintenant une généralisation immédiate du théorème précédent.

Corollaire.- Soit $\mathcal{F}$ une théorie ouverte, $q[x] = \exists y\, p[x,y]$ une formule

existentielle. Si q est un théorème de $\mathcal{F}$, il existe des exemplaires $p_1[x], \ldots, p_k[x]$ des axiomes non logiques de $\mathcal{F}$, et des n-uplets de termes $u_1[x],\ldots,u_m[x]$ tels que

$$p_1[x] \supset \ldots \supset p_k[x] \supset \left(p[x,u_1[x]] \vee \ldots \vee p[x,u_m[x]]\right)$$

soit une quasi-tautologie.

Démonstration : Supposons que $x = (x_1,\ldots x_p) \in V_i$. Soit $a = (a_1,\ldots a_p)$ un p-uple de constantes n'appartenant pas à L, de même type que $x_1,\ldots,x_p$ et soit $\mathcal{F}'$ la théorie obtenue en ajoutant ces constantes à l'alphabet. Nous avons besoin du lemme suivant.

Lemme 3.- (lemme des constantes) p est un théorème de $\mathcal{F}$ si et seulement si $p[a]$ est un théorème de $\mathcal{F}'$.

Démonstration : Si $\vdash_{\mathcal{F}} p$ alors $\vdash_{\mathcal{F}} \forall y\, p$; donc $\vdash_{\mathcal{F}} p[a]$ par la règle de substitution. Maintenant, si D est une démonstration de $p[a]$ dans $\mathcal{F}'$ soit $y_1,\ldots y_p$ p variables n'apparaissant pas dans D alors remplaçons $a_1,\ldots,a_p$ par $y_1,\ldots,y_p$. Il est facile de voir qu'on obtient ainsi une démonstration dans $\mathcal{F}$ de $p[y]$. Donc, $\vdash_{\mathcal{F}} p[y]$ soit $\vdash_{\mathcal{F}} p$ par la règle de substitution.

Revenons à la démonstration du corollaire.

Si q est un théorème de $\mathcal{F}$, $q[a]$ est un théorème de $\mathcal{F}'$. Il existe donc des exemplaires $p_1[a],\ldots,p_k[a]$ des axiomes non logiques de $\mathcal{F}'$ (donc de $\mathcal{F}$) et des n-uplets $u_1[a],\ldots, u_m[a]$ tels que

$$p_1[a] \supset \ldots \supset p_k[a] \supset (p[a,u_1[a]] \vee \ldots \vee p[a,u_m[a]])$$

soit une quasi-tautologie.

Le corollaire résulte alors du lemme précédent.

3

extensions d'une structure d'information

3.- EXTENSIONS D'UNE STRUCTURE D'INFORMATION

Dans ce chapitre, nous nous intéressons à la notion d'extension d'une structure d'information. Cette notion est utilisée plusieurs fois dans la suite de ce travail et le présent chapitre est une étude des propriétés générales. On obtient une extension en ajoutant des symboles et en "définissant" ces accès par un ensemble d'axiomes. Il peut s'agir de représenter une notion complexe par un nouveau symbole. Le symbole $\leqslant$ peut être défini, par exemple, par

$$x \leqslant y \longleftrightarrow x < y \vee x \equiv y$$

Il s'agira plus souvent de définir un nouvel accès de manière récursive (modes recursifs d'Algol 68 (cf 3.3) ou définitions récursives de fonctions (4)).

Sur un autre plan, pour définir le passage d'une information I_1 sur un alphabet L_1 à une autre information I_2 sur un alphabet L_2, on peut considérer une extension I' de I_1 et I_2 et représenter les relations entre I_1 et I_2 par un système d'axiomes Y. Cette technique est utilisée au chapitre 5 pour définir les modifications d'une structure d'informations (5.1) et pour définir la notion de problème comme passage d'une information donnée à une information résultat (5.3).

3.1.- Définitions

Soient $\mathcal{S}$ et $\mathcal{S}'$ deux structures d'information. On désigne par F ,

$T, \mathcal{T}$, et $F', T', \mathcal{T}'$ respectivement l'ensemble des formules, des théorèmes et des informations de $\mathcal{S}$ et $\mathcal{S}'$.

Définition 1 : On dit que $\mathcal{S}'$ est une extension de $\mathcal{S}$ si $F \subset F'$ et $T \subset T'$.

Il suffit pour cela que F soit inclu dans F' et que tout axiome propre de $\mathcal{S}$ soit un théorème de $\mathcal{S}'$.

Définition 2 : Soit I une information de $\mathcal{S}$; l'extension de I à $\mathcal{S}'$ est la plus petite information $\mathcal{E}(I)$ de $\mathcal{S}'$ contenant I. Soit I' une information de $\mathcal{S}'$; on appelle restriction de I' à $\mathcal{S}$ l'information $\mathcal{R}(I') = I' \cap F$ de $\mathcal{S}$.

Rappelons que, si I est une information, I_0 désigne la restriction de I à l'ensemble des formules sans variable. On pose alors

$$\mathcal{E}(I_0) = \left[\mathcal{E}(I)\right]_0$$

et $$\mathcal{R}(I'_0) = I'_0 \cap F .$$

Nous donnons d'abord un résultat sur les extensions conservant la complétude et la consistance.

Proposition 1.- Si $\mathcal{E}$ conserve la consistance,

$\mathcal{R} \circ \mathcal{E}(I_0) = I_0$ pour toute information I de $\mathcal{S}$.

Si $\mathcal{E}$ conserve la complétude,

$\mathcal{E} \circ \mathcal{R}(I'_0) = I'_0$ pour toute information complète I' de $\mathcal{S}$.

Remarque : Si $\mathcal{E}$ conserve la complétude, elle conserve évidemment la consistance.

Démonstration :

a) Supposons que $\mathcal{E}$ conserve la consistance.

. Si I est complète, $\mathcal{E}(I)$ est consistante ainsi que $\mathcal{E}(I) \cap F$ (2.2 Proposition 2). De plus $\mathcal{E}(I) \cap F$ contient I. Donc $\mathcal{R} \circ \mathcal{E}(I_0)$ contient I_0 et, puisque I_0 est maximale, $\mathcal{R} \circ \mathcal{E}(I_0) = I_0$.

. Si I est consistante, notons que

$$I_0 = \bigcap \left\{ J_0 \ ; \ J \text{ complète et } I \subset J \right\}.$$

Donc, $$\mathcal{R} \circ \mathcal{E}(I_0) \subset \bigcap \left\{ \mathcal{R} \circ \mathcal{E}(J_0) \ ; \ J \text{ complète et } I \subset J \right\}$$

$$= \bigcap \left\{ J_0 \ ; \ J \text{ Complète et } I \subset J \right\} = I_0.$$

D'autre part, $\mathcal{R} \circ \mathcal{E}(I_0)$ contient I_0, d'où l'égalité.

. Enfin, si $I = F$, $I_0 = F_0 = \mathcal{R} \circ \mathcal{E}(I_0)$.

b) Supposons maintenant que $\mathcal{E}$ conserve la complétude. Si I' est complète, $I' \cap F$ est également complète (2.2 proposition 5) ainsi que $\mathcal{E}(I' \cap F)$. Puisque I' contient $\mathcal{E}(I' \cap F)$, il vient

$$I'_0 = \mathcal{E} \circ \mathcal{R}(I'_0).$$

Exemple : Dans la structure des entiers (1.2.3) le quotient de deux nombres x et y peut être défini par la formule

$$y \neq 0 \supset \left[0 \leqslant x - y * \underline{\text{quotient}}(x,y) < y \right]$$

(où $\leqslant$ est défini au début du chapitre et où la notation $u \leqslant v < w$ est une abréviation de $u \leqslant v \wedge v < w$). On peut également écrire

$$y \neq 0 \supset \underline{\text{quotient}}(x,y) \equiv \underline{\text{si}}\ x < y\ \underline{\text{alors}}\ 0\ \underline{\text{sinon}}\ 1 + \text{quotient}(x-y,y)$$

Chacune de ces formules définit complètement le quotient de deux nombres m,n arbitraires, sauf pour n = 0. On obtient donc une information consistante et, si l'on ajoute, par exemple, l'axiome quotient(x,0) = 0, une information complète.

Plus généralement, considérons un alphabet L' contenant strictement L et un ensemble Y d'axiomes. On peut associer à L' et Y un problème dont les inconnues sont les symboles de L'-L et dont l'énoncé est constitué par les formules de Y. Chaque information complète I de $\mathcal{S}$ peut être considérée comme une donnée du problème.

On appelle résultat du problème toute information I" complète contenant $\mathcal{C}(I)$. On peut alors se poser les questions suivantes :

1) $\mathcal{C}(I)$ est-elle consistante ? Autrement dit, le problème admet-il des solutions (non nécessairement uniques) pour la donnée I ? Il suffit en fait de trouver une interprétation de $\mathcal{C}(I)$.

2) Existe-t'il une solution "constituée d'objets de la structure $\mathcal{S}$" ? Avant de préciser cette question, considérons l'exemple suivant :

Exemple : modes récursifs d'Algol 68

Soit M l'ensemble des modes constitués à partir du symbole 0-aire ent(mode entier), du symbole 1-aire rep(repère) et du symbole binaire $\text{struct}_{x,y}$ (structure à deux champs de sélecteurs x et y)

$$M = \{\text{ent}\} \cup \text{rep}\, M \cup \text{struct}_{x,y}\, M^2$$

On peut définir des modes récursifs comme le mode a défini par la déclaration

$$\text{mode a} = \text{struct}_{x,y}(\text{rep a} , \text{reel}) \qquad (D)$$

Un tel mode est différent, d'après le rapport (ALGOL 68), de tous les modes de M. Autrement dit, le "problème" d'inconnue a et d'énoncé (D) admet effectivement une solution, bien que a soit distinct de tous les modes de la structure initiale. Nous donnons, au paragraphe 3, une formalisation tenant compte de la sémantique d'Algol 68.

Nous étudions ce type d'extension au paragraphe 2.

3) $\mathcal{C}(I)$ est-elle complète ? Sinon peut-on choisir, parmi les résultats du problème, un résultat privilégié ? Nous étudions ce point sur 2 types de problèmes au paragraphe 3 et au chapitre 4.

3.2.- Extensions conservatrices

Définition 3 : Soit $\mathcal{S}'$ une extension de $\mathcal{S}$, I une information complète de $\mathcal{S}$. On dit que $\mathcal{C}$ est conservatrice en I s'il existe une information complète I" contenant $\mathcal{C}(I)$ et vérifiant la propriété :

(C) Pour tout terme u sur L', sans variable, il existe v appartenant à S tel que $u \equiv v \in I"$.

Proposition 2.- Soit I une information complète de $\mathcal{S}$ et I" une information consistante de $\mathcal{S}'$, contenant I et vérifiant la propriété (C') suivante

(C') Pour tout $f \in L'-L$, toute suite de termes $(u_1,\dots,u_n)$ sans variable appartenant à S tels que $fu_1 \dots u_n$ soit un terme, il existe $v \in S$ tel que

$$fu_1 \dots u_n \equiv v \in I".$$

Alors

i) I" vérifie la condition (C)

ii) I" est complète.

Démonstration :

i) Raisonnement par récurrence sur la longueur de u . Soit f appartenant à L', de profil (i,j), et $u_1,\ldots,u_n$ des termes de types $i_1,\ldots,i_n$, sans variable, appartenant à S'. Par récurrence, il existe $v_1,\ldots,v_n$, sans variable, appartenant à S tel que

$$u_k \equiv v_k \in I" \quad \text{pour} \quad k = 1,\ldots,n$$

Alors, si $f \in L$, $fv_1 \ldots v_n \in S$ et

$$fu_1 \ldots u_n \equiv fv_1 \ldots v_n \in I"$$

Si $f \in L'-L$, il existe d'après l'hypothèse (C'), un terme v tel que

$fv_1 \ldots v_n \equiv v \in I"$. Alors,

$fu_1 \ldots u_n \equiv v \in I"$.

ii) Soit u,u' des termes sur L', sans variable et v,v' des termes sur L, sans variable tels que $u \equiv v$ et $u' \equiv v'$ appartiennent à I". Alors

$u \equiv u' \notin I" \implies v \equiv v' \notin I" \implies v \neq v' \in I$ (puisque I est complète) $\implies u \neq u' \in I"$.

Donc I" est complète.

Conséquence : Soit I une information complète de $\mathcal{S}$ et I" une information de $\mathcal{S}'$ contenant I et vérifiant (C). Soit $\Gamma = \{fu_1 \ldots u_n \equiv v \in I" ;\ f \in L'-L,\ u_1,\ldots,u_n,v \in S\}$. Alors $I[\Gamma]$ est une information complète, contenue dans I". Donc, $I[\Gamma]$ et I" admettent les mêmes interprétations.

Proposition 3.- Soit I une information complète et R = (E,r) une interprétation de I. $\mathcal{C}$ est conservatrice en I si et seulement si l'on peut trouver un prolongement r' de r à L' tel que R' = (E,r') soit une interprétation de $\mathcal{C}(I)$.

Démonstration : La condition est suffisante. Soit r' un prolongement de r tel que R' = (E,r') soit une interprétation de $\mathcal{C}(I)$ et soit $I" = \{p \in F' ;\ \tilde{r}'(p) = \lambda x.\underline{vrai}\}$. I" est évidemment complète. De plus, si u est un terme sans variable de type j , $\hat{r}'(u) \in E_j = \hat{r}(S_j)$. Il existe donc $v \in S_j$ tel que $\hat{r}'(u) = \hat{r}(v) = \hat{r}'(v)$, donc tel que $u \equiv v \in I"$.

La condition est nécessaire. Pour tout $f \in L'-L$ de profil (i,j) et tout $x \in E_i$, soit $u \in S_i$ tel que $\hat{r}(u) = x$ et soit $v \in S_j$ tel que $fu \equiv v \in I"$. On pose alors

$$r'(f)(x) = \hat{r}(v)$$

Cette définition de r' ne dépend pas du choix de u et v. Soit (u',v') un autre couple tel que

$$\hat{r}(u) = \hat{r}(u') \qquad (1)$$

et $$fu' \equiv v' \in I" \qquad (2)$$

Puisque I est complète, (1) implique que $u_k \equiv u'_k \in I$ pour $k = 1,\ldots,n$, donc que $fu \equiv fu' \in I"$. En définitive $v \equiv v'$ appartient à I", donc à I puisque I est complète et $\hat{r}(v) = \hat{r}(v')$.

On pose, par ailleurs, $r'(f) = r(f)$ pour $f \in L$. Il est alors clair que $\hat{r}'(fu) = \hat{r}'(v)$ chaque fois que $fu \equiv v \in I"$ avec $v \in S_j$ et $u \in S_i$. R' = (E,r') est donc une interprétation de $I[\Gamma]$ (avec les notations précédentes), donc de I".

3.3.- Etude des modes récursifs d'Algol 68

Nous nous intéressons ici à un sous ensemble de l'ensemble des modes Algol 68, suffisant pour illustrer les résultats précédents. La généralisation ne présente pas de difficulté particulière et a été étudiée, en particulier, dans (FINANCE).

On considère la structure $\mathcal{M}$ définie de la manière suivante :

L'alphabet L de $\mathcal{M}$ est formé des symboles

- réel, ent, bool, 0-aires
- rep, unaire
- $\text{struct}_{x,y}$, binaire (x,y sont deux sélecteurs de champ).

En général, on devrait considérer un symbole $\text{struct}_{\bar{x}}$ pour chaque suite $\bar{x}$ de sélecteurs. On écrira struct (mx, ny) au lieu de $\text{struct}_{x,y}(m,n)$. On désigne par $\hat{L}$ l'ensemble des modes, c'est-à-dire des termes sur L.

Les axiomes de $\mathcal{M}$ sont tels que deux termes distincts représentent des modes différents.

① Pour tous symboles distincts de L, f k-aire et g ℓ-aire,

$$f\ x_1 \dots x_k \not\equiv g\ y_1 \dots y_\ell$$

② Pour tout symbole f, k-aire de L

$$f\ x_1 \dots x_k \equiv f\ y_1 \dots y_k \supset \left[x_1 \equiv y_1 \wedge \dots \wedge x_k \equiv y_k\right]$$

Soit T l'ensemble des théorèmes de $\mathcal{M}$. On montre immédiatement que I est consistante et que, pour tous m,n distincts de $\hat{L}$

$$m \not\equiv n \in I.$$

I est donc une information complète.

Considérons alors un symbole $a \notin L$, 0-aire et un terme h sans variable sur $L \cup \{a\}$ tel que $h \neq a$ et $h \notin \hat{L}$ et soit $\mathcal{M}'$ la structure d'alphabet $L' = L \cup \{a\}$ obtenue en ajoutant l'axiome $a \equiv h$. Soit enfin I' l'extension de I dans $\mathcal{M}'$.

Exemple : $a \equiv \text{struct}\ (\text{rep}\ a\ x, \text{reel}\ y)$

Proposition 4.- Pour tout mode m de $\hat{L}$ et pour tout mode n de $\hat{L}'-L$ $m \not\equiv n$ appartient à I'.

Autrement dit, un mode récursif est différent de tout mode obtenu par composition des modes élémentaires.

Démonstration : Montrons par récurrence sur la longueur de m que, pour tout mode $n \in \hat{L}'-\hat{L}$,

$$m \not\equiv n \in I' \qquad (1)$$

Soit $\ell \geq 1$. Supposons (1) vérifié pour $|m| < \ell$. Soit f un symbole k-aire $\in L$, $m_1,\dots,m_k$ des termes de longueur inférieure à ℓ et $n \in \hat{L}'-\hat{L}$. Si $n = a$, remplaçons a par h. 2 cas sont à envisager

. $n = g\ n_1 \dots n_\ell$ avec $g \neq f$ et $g \neq a$. Dans ce cas, il suit de l'axiome (1) que $m \not\equiv n \in I'$.

. $n = f\ n_1 \dots n_k$. Il existe $k' \in [1,k]$ tel que $n_{k'}$ appartienne à $\hat{L}'-\hat{L}$. Alors, par récurrence $m_{k'} \not\equiv n_{k'} \in I'$ et, en appliquant l'axiome (2), $m \not\equiv n \in I'$.

Pour démontrer que I' est consistante, il nous faut construire une interprétation de I'. Nous généralisons auparavant le problème.

Considérons pour cela un ensemble A de symboles 0-aires, disjoint de L et, pour chaque $a \in A$, un terme h_a appartenant à $\widehat{L \cup A} - \hat{L}$, de longueur stric-

tement plus grande que 1. On peut considérer la structure $\mathcal{M}'$ d'alphabet $L' = L \cup A$, obtenue en ajoutant le système

$$\{ a \equiv h_a \ ; \ a \in A \}$$

d'équations mutuellement récursives. Nous supposons de plus $h_a \neq h_b$ si $a \neq b$.

La proposition 4 se généralise immédiatement.

Pour démontrer la consistance de I', nous construisons une interprétation. Pour cela, nous commençons par associer à chaque mode m de $\hat{L}'$ une forme minimale $\hat{m}$ vérifiant les propriétés suivantes

. $\hat{h}_a = \hat{a} = a$ pour tout $a \in A$

. Si f est un symbole k-aire $(k \geqslant 1)$ et si $\hat{m}_i = \hat{n}_i$ pour $i = 1,\ldots,k$ alors

$$\widehat{f\ m_1 \ \ldots \ m_k} = \widehat{f\ n_1 \ \ldots \ n_k}$$

. Réciproquement, si $\hat{m}_i \neq \hat{n}_i$ pour un $i \in [1,k]$

$$\widehat{f\ m_1 \ \ldots \ m_k} \neq \widehat{f\ n_1 \ \ldots \ n_k}$$

. Si g est un symbole ℓ-aire distinct de f

$$\widehat{f\ m_1 \ \ldots \ m_k} = \widehat{g\ n_1 \ \ldots \ n_\ell} \ .$$

Il est clair, alors, que la relation d'équivalence $\sim$ définie par

$$m \sim n \iff \hat{m} = \hat{n}$$

est compatible avec la composition des symboles fonctionnels (2.2 définition 8) et que l'interprétation canonique associée à $\sim$ (2.2 définition 9) est une interprétation de I'.

Nous définissons $\hat{m}$ par récurrence sur la longueur de m,

. Si $m \in A$, $\hat{m} = m$

. Si $m = f\ m_1 \ \ldots \ m_k$ avec $f \in L$,

$$\hat{m} = \begin{cases} a \text{ si } f\ \hat{m}_1 \ \ldots \ \hat{m}_k = h_a \\ f\ \hat{m}_1 \ \ldots \ \hat{m}_k \text{ sinon} \end{cases}$$

Notons que $\hat{m}$ est défini sans ambiguïté puisque $h_a \neq h_b$ si $a \neq b$.

Exemple : Considérons le système

$$\begin{cases} a \equiv \underline{\text{struct}}\ (\underline{\text{rep}}\ a\ x,\ \underline{\text{rep}}\ b\ y) \\ b \equiv \underline{\text{struct}}\ (\underline{\text{rep}}\ b\ x,\ \underline{\text{rep}}\ a\ y) \end{cases}$$

et $\quad m = \underline{\text{rep}}\ \underline{\text{struct}}\ (\underline{\text{rep}}\ b\ x,\ \underline{\text{rep}}\ a\ y)$

$\quad n = \underline{\text{struct}}\ (\underline{\text{rep}}\ a\ x,\ \underline{\text{rep}}\ \underline{\text{struct}}\ (\underline{\text{rep}}\ b\ x,\ \underline{\text{rep}}\ a\ y)\ y)$

Alors $\quad \hat{m} = \underline{\text{rep}}\ b$ et $\hat{n} = a$.

Vérifions, par exemple que $\wedge$ vérifie la troisième propriété

$$\hat{m}_i \neq \hat{n}_i \text{ pour un } i \in [1,k] \implies \widehat{f\ m_1 \ \ldots \ m_k} \neq \widehat{f\ n_1 \ \ldots \ n_k}$$

En effet $f\ \hat{m}_1 \ \ldots \ \hat{m}_k \neq f\ \hat{n}_1 \ \ldots \ \hat{n}_k$. Plusieurs cas sont alors à envisager

. Si $f\ \hat{m}_1 \ \ldots \ \hat{m}_k = h_a$ pour un $a \in A$, $\widehat{f\ m_1 \ \ldots \ m_k} = a$ et $\widehat{f\ n_1 \ \ldots \ n_k} \neq a$. Et de même pour $f\ \hat{n}_1 \ \ldots \ \hat{n}_k$.

. Sinon $\widehat{f\ m_1 \ \ldots \ m_k} = f\ \hat{m}_1 \ \ldots \ \hat{m}_k \neq f\ \hat{n}_1 \ \ldots \ \hat{n}_k = \widehat{f\ n_1 \ \ldots \ n_k}$.

Remarquons pour finir que

$$\hat{a} \neq \hat{b}$$

pour tout a,b de A tels que $a \neq b$.

Si $R = (E,r)$ désigne l'interprétation canonique associée à $\sim$, il vient $\tilde{r}(a \not\equiv b) = \underline{\text{vrai}}$, ce qui prouve que $a \equiv b$ n'est pas un théorème de I'.

Il est assez intuitif, dans l'exemple précédent, que $a \not\equiv b$ n'est pas non plus un théorème de I'. Pour le prouver rigoureusement, nous construisons maintenant une interprétation $R' = (E',r')$ de I' telle que $\tilde{r}'(a \equiv b) = \underline{\text{vrai}}$. Dans le cas d'un ensemble A quelconque, nous voulons que

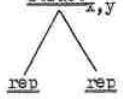
$$\tilde{r}(m \not\equiv n) = \underline{\text{vrai}} \iff m \not\equiv n \in I'$$

Dans le cas où $\tilde{r}(m \not\equiv n) = \underline{\text{faux}}$, nous dirons que m et n sont indiscernables.

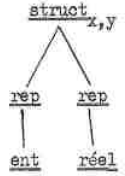
Nous associons cette fois à chaque mode m une description d(m) de ce mode. Une description est une suite $d_1,\ldots,d_n,\ldots$ de pseudo-arborescences sur L (PAIR-QUERE) définissant m de mieux en mieux. Donnons d'abord deux exemples.

1) $m = \underline{\text{struct}}\ (\underline{\text{rep}}\ \underline{\text{ent}}\ x\ ,\ \underline{\text{rep}}\ \underline{\text{réel}}\ y)$

On pose $d_0(m) = \underline{\text{struct}}_{x,y}$ $\quad d_1(m) =$

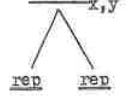


et pour $k \geq 2$ $\quad d_k(m) = d_2(m) =$

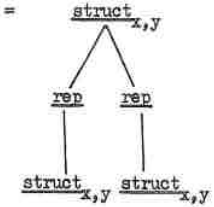


2) Considérons les symboles a,b définis précédemment. On pose

$d_0(a) = \underline{\text{struct}}_{x,y}$ $\quad d_1(a) =$



$d_2(a) =$



etc ...

De plus $d(a) = d(h_a) = d(h_b) = d(h_b) = d(b)$

Plus généralement, si $a \in L$ et si $d_1,\ldots,d_k$ sont des pseudo-arborescences sur L, on note $a\,x\,[d_1 + \ldots + d_k]$ l'enracinement de $(r_1,\ldots,r_k)$ par a.

Pour tout $m \in \hat{L}'$, $m \notin A$, il existe un symbole f k-aire et des modes $m_1 \ldots m_k$ tels que $m = f\ m_1 \ldots m_k$. On pose alors $d_0(m) = f$. Si $m \in A$, on pose $d_0(m) = d_0(h_m)$ (rappelons que $|h_m| > 1$).

Supposons définies, pour chaque m, les pseudo-arborescences $d_0(m)$, ... $d_{n-1}(m)$. On définit alors $d_n(m)$ par

$$d_n(m) = \begin{cases} f\ x\ [d_{n-1}(m_1) + \ldots + d_{n-1}(m_k)] & \text{si } m = f\ m_1 \ldots m_k \text{ et } f \in L \\ d_n(h_m) & \text{si } m \in A. \end{cases}$$

La suite $d(m) = (d_0(m), d_1(m),\ldots)$ est la description de m. On considère alors la relation $\sim$ définie pour tous m,n par

$$m \sim n \iff d(m) = d(n).$$

Proposition 5.- La relation $\sim$ vérifie les propriétés suivantes :

i) Soit $f \in L$ k-aire, $m_1,\ldots,m_k$, $n_1,\ldots,n_k$ des modes

$$f\ m_1 \ldots m_k \sim f\ n_1 \ldots n_k \iff (m_i \sim n_i \text{ pour } i = 1,\ldots,k)$$

ii) Si $f \neq g$ $\quad f\ m_1 \ldots m_k \sim g\ n_1 \ldots n$

iii) Pour tout $a \in A$ $\quad a \sim h_a$.

iv) $m \sim n \Rightarrow m \equiv n \in I'$.

Démonstration :

i) Soit $m = f\ m_1 \ldots m_k$ et $n = f\ n_1 \ldots n_k$

$$m \sim n \iff (d_p(m) = d_p(n) \text{ pour } p \geqslant 1)$$

$$\iff \text{pour } i = 1,\ldots,k \ (d_{p-1}(m) = d_{p-1}(n_i) \text{ pour } p \geqslant 1)$$

$$\iff (m_i \sim n_i \text{ pour } i = 1,\ldots k).$$

ii) et iii) sont évidents.

iv) Soit p le plus petit entier tel que $d_p(m) \neq d_p(n)$. On peut toujours supposer que m,n n'appartiennent pas à A (quitte à remplacer a par h_a). Montrons par récurrence sur p que $m \not\equiv n \in I'$. Si $p = 0$ $m = f\ m_1 \ldots m_k$ et $n = g\ n_1 \ldots n_\ell$. Donc $m \not\equiv n \in I'$. Si $p > 0$ $d_p(m) \neq d_p(n)$ si et seulement si $d_{p-1}(m_i) \neq d_{p-1}(n_i)$ pour un i. Par récurrence, $m_i \not\equiv n_i \in I'$. Donc $m \not\equiv n \in I'$.

Soit $R' = (E',r')$ l'interprétation canonique associée à $\sim$. On obtient immédiatement le résultat suivant.

Proposition 6.- R' est une interprétation de I'. Plus précisément soit $\Gamma = \{ m \equiv n \ ; \ m \not\equiv n \notin I' \}$ et $I'' = I'[\Gamma]$. R' est une interprétation de I''. I'' est donc consistante et complète.

Remarque : I'' formalise la sémantique des modes récursifs d'Algol 68.

Démonstration : Il résulte des points i), ii), iii) de la proposition précédente que $\tilde{r}'(p) = \lambda x.\underline{vrai}$ pour tout axiome de I'. De plus, si $m \not\equiv n \notin I'$, il résulte de iv) que $m \sim n$. Donc $\tilde{r}'(m \equiv n) = \underline{vrai}$ et R' est donc une interprétation de I''. I'' est donc consistante. De plus, pour tous m,n de $\hat{L}'$

$$m \not\equiv n \notin I'' \Rightarrow m \not\equiv n \notin I' \Rightarrow m \equiv n \in I''.$$

I'' est donc complète.

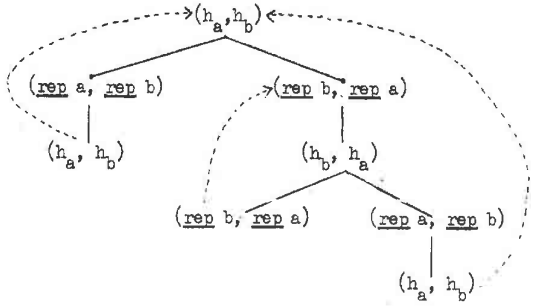
Pour conclure, donnons un algorithme permettant de décider si $m \sim n$ ou non. Appelons comparaisons d'un couple (m,n) de modes les pseudo-arborescences sur $\hat{L}' \times \hat{L}'$ construites en appliquant les règles suivantes. (Pour tout m de $\hat{L}'$, on pose $\tilde{m} = m$ si $m \notin A$ et $\tilde{a} = h_a$ si $a \in A$).

1) Pour tout (m,n), $(\tilde{m},\tilde{n})$ est une comparaison de (m,n).

2) Si $\tilde{m} = f\ m_1 \ldots m_k$ et $\tilde{n} = f\ n_1 \ldots n_k$ et si, pour $i \in [k]$ c_i est une comparaison de (m_i,n_i), $(\tilde{m},\tilde{n}) \times [c_1 + \ldots + c_k]$ est une comparaison de (m,n).

Exemple : Soient a,b les deux modes définis précédemment ; (a,b) admet une infinité de comparaisons. Considérons en particulier la pseudo-arborescence.

c(a,b) =



Notons que $c(a,b)$ vérifie la propriété suivante.

Pour toute feuille x de $c(a,b)$, il existe un point situé au-dessus de x, de même valeur que x.

Plus généralement, pour tout couple (m,n), il existe une comparaison standard $c(m,n)$ unique vérifiant les propriétés suivantes :

1) Pour toute feuille (m',n') de $c(m,n)$, 3 situations seulement sont possibles

a) m' et n' commencent par des symboles distincts.

b) $m' = n'$.

c) il existe un ascendant de (m',n') de même valeur.

2) Les situations b) et c) ne se produisent qu'aux feuilles de $c(m,n)$.

La démonstration de l'existence de $c(m,n)$ est basée sur le fait que pour chaque (m,n) il existe un nombre maximum $A(m,n)$ de couples distincts dans une comparaison de (m,n).

Nous pouvons montrer alors que $m \sim n$ si et seulement si toutes les feuilles de $c(m,n)$ vérifient b) ou c). Pour cela nous introduisons la définition suivante. Nous dirons qu'une comparaison est juste si, pour toute feuille (m',n'), m' et n' commencent par le même symbole. Nous avons besoin de deux lemmes.

Lemme 1.- $d(m) = d(n)$ si et seulement si toutes les comparaisons de (m,n) sont justes.

Démonstration : La condition est nécessaire. Raisonnons par récurrence sur la hauteur h d'une comparaison $c'(m,n)$. Si $h = 1$, $c'(m,n) = (\tilde{m},\tilde{n})$. Puisque $d(m) = d(n)$, $\tilde{m}$ et $\tilde{n}$ commencent par le même symbole. Si $h > 1$ on peut écrire

$m = f\, m_1 \ldots m_k$ et $n = f\, n_1 \ldots n_k$ et $c'(m,n) = (\tilde{m},\tilde{n}) x \left[c'(m_1,n_1)+..+c'(m_k,n_k)\right]$. Par récurrence $c'(m_i,n_i)$ est juste pour $i = 1,\ldots,k$. Donc $c'(m,n)$ est juste.

La condition est suffisante. Supposons que $d_p(m) \neq d_p(n)$ pour un $p \geq 0$. Si $p = 0$ (m,n) ne commence pas par le même symbole. Si $p > 0$, il existe $i \in [k]$ tel que $d_{p-1}(m_i) \neq d_{p-1}(n_i)$. Par récurrence, il existe une comparaison $c'(m_i,n_i)$ non juste. Pour $j \neq i$, posons $c'(m_j,n_j) = (\tilde{m}_j,\tilde{n}_j)$. Alors $(\tilde{m},\tilde{n})$ x $\left[c'(m_1,n_1) + \ldots + c'(m_k,n_k)\right]$ n'est pas juste.

Lemme 2.- Si la comparaison standard de (m,n) est juste, toutes les comparaisons de (m,n) sont justes.

Démonstration : Raisonnons par récurrence sur la hauteur h de la comparaison standard $c(m,n)$ de (m,n). Si $h = 1$, $\tilde{m} = \tilde{n}$. Si $h > 1$, on peut écrire $c(m,n) = (\tilde{m},\tilde{n})$ x $\left[c'(m_1,n_1) + \ldots + c'(m_k,n_k)\right]$. Pour chaque i de $[k]$, ou bien $c'(m_i,n_i)$ est standard ou bien il existe des feuilles de la forme $(\tilde{m},\tilde{n})$. Dans ce cas, on obtient la comparaison standard en remplaçant ces feuilles par $(\tilde{m},\tilde{n})$ x $\left[(\tilde{m}_1,\tilde{n}_1)+\ldots+(\tilde{m}_k,\tilde{n}_k)\right]$. Dans chaque cas, la comparaison standard de (m_i,n_i) est juste. Donc toute comparaison de (m_i,n_i) est juste et, de même, toute comparaison de (m,n).

4

équations récursives dans les structures d'information

4.- EQUATIONS RECURSIVES DANS LES STRUTURES D'INFORMATION

4.1.- Introduction

Considérons, dans $\mathcal{F}(\mathbb{N}, \mathbb{N})$, l'équation

$$F(n) \text{ si } n = 0 \text{ alors } 1 \text{ sinon } n * F(n-1) \quad (n \in \mathbb{N})$$

qu'il est plus correct d'écrire, en utilisant la λ-notation,

$$F = \lambda n.\ (\text{si } n = 0 \text{ alors } 1 \text{ sinon } n * F(n-1)).$$

Cette équation admet une solution unique Fact (On note encore Fact(n) = n !). Fact est l'unique point fixe de la fonctionnelle τ_1 définie sur $\mathcal{F}(\mathbb{N}, \mathbb{N})$ par

$$\tau_1(F) = \lambda n.\ (\text{si } n = 0 \text{ alors } 1 \text{ sinon } n * F(n-1)).$$

La fonctionnelle τ_2 :

$F \longmapsto \tau_2(F) = \lambda n.(\text{si } F(n) = 0 \text{ alors } 1 \text{ sinon } 0)$ n'admet pas de point fixe dans $\mathcal{F}(\mathbb{N}, \mathbb{N})$. On aimerait dire alors que τ_2 admet comme point fixe la fonction jamais définie. Introduisons pour cela un élément $\perp$ (élément non défini) et prolongeons l'application Cond sur $\mathbb{N}_+^4$(*) en posant :

$$\text{Cond}(x,x,z,t) = z \qquad x \in \mathbb{N}$$

---

* Si E est un ensemble, on notera généralement $E_+ = E \cup \{\perp\}$.

$$\mathrm{Cond}(x,y,z,t) = t \qquad x,y \in \mathbb{N} \qquad x \neq y$$

$$\mathrm{Cond}(\perp,y,z,t) = \mathrm{Cond}(x,\perp,z,t) = \perp$$

$\mathcal{T}_2$ induit naturellement une fonctionnelle $\mathcal{T}_2'$ sur $\mathcal{F}(\mathbb{N}_+, \mathbb{N}_+)$ dont l'unique point fixe est l'application $\perp = \lambda n.\perp$

Enfin la fonctionnelle $\mathcal{T}_3$

$$\mathcal{T}_3(F)(x_1,x_2) = \underline{\text{si}}\ x_1 = x_2\ \underline{\text{alors}}\ x_2{+}1\ \underline{\text{sinon}}\ F(x_1,F(x_1{-}1,x_2{+}1))$$

admet comme points fixes les applications

$$F = \lambda x_1,x_2.(\underline{\text{si}}\ x_1 = x_2\ \underline{\text{alors}}\ x_2{+}1\ \underline{\text{sinon}}\ x_1{+}1)$$

$$G = \lambda x_1,x_2.(\underline{\text{si}}(x_1 \geqslant x_2)\ \underline{\text{alors}}\ x_1{+}1\ \underline{\text{sinon}}\ x_2{-}1)$$

$$H = \lambda x_1,x_2.(\underline{\text{si}}(x_1 \geqslant x_2)\ \text{et}\ (x_1{-}x_2\ \underline{\text{pair}})\ \underline{\text{alors}}\ x_2{+}1\ \underline{\text{sinon}}\ \perp)$$

Notons que pour $x_1 < x_2$ $H(x_1,x_2) = \perp$ tandis que $F(x_1,x_2) \neq G(x_1,x_2)$. On peut dire que H est moins définie que F et G et donc que H est le plus petit point fixe de $\mathcal{T}_3$, si l'on munit $\mathcal{F}(\mathbb{N}_+^2, \mathbb{N}_+)$ de la relation "moins définie que ".

Au paragraphe 2, nous étudions l'existence d'un plus petit point fixe lorsque la fonctionnelle vérifie une propriété de continuité et nous donnons une construction de ce plus petit point fixe. Décrivons cette méthode dans le cas de n!. La fonctionnelle $\mathcal{T}_1$ induit une fonctionnelle $\mathcal{T}_1'$ sur $\mathcal{F}(\mathbb{N}_+, \mathbb{N}_+)$ (On pose $n * \perp = \perp * n = \perp$; $0{-}1 = \perp - 1 = \perp$). Soit

$$F_0 = \lambda n.\ \perp = \perp \qquad \text{et}$$

$$F_n = \mathcal{T}_1'(F_{n-1}) \quad \text{pour} \quad n \geqslant 1$$

On vérifie immédiatement que

$$F_n(\perp) = \perp \qquad \text{pour} \qquad n \geqslant 0$$

et $$F_n(x) = x\,! \qquad \text{pour} \qquad n > x$$

La suite $(F_n)_{n \geqslant 0}$ est donc croissante pour l'ordre "moins définie que" et converge vers Fact.

Cette construction du plus petit point fixe ne correspond pas cependant au calcul informatique de factorielle, tel qu'il découle, par exemple, de la sémantique de la procédure Algol 68 suivante

proc fact = (ent n)ent : si n = 0 alors 1 sinon n * fact(n-1)fsi

Par exemple, Fact(2) peut être calculé en écrivant successivement que

Fact(2) = si 2 = 0 alors 1 sinon 2 * Fact(1) = 2 * Fact(1)

= 2 * (si 1 = 0 alors 1 sinon 1 * Fact(0)) = 2 * (1 * Fact(0))

= 2 * (si 0 = 0 alors 1 sinon 0 * Fact(0-1)) = 2 * 1 = 2.

Nous formalisons au paragraphe 3 la notion de fonctionnelle dans une structure d'information $\mathcal{S}$ en introduisant un symbole f (de profil (i,j)) n'appartenant pas à L. Si τ (resp p) est un terme (resp une formule) sur $L' \cup \{f\}$ et h un terme de même profil que f, on définit alors la substitution de h à f dans τ (resp p) ; cette opération correspond à la substitution complète développée dans (VUILLEMIN, CADIOU).

Si τ est un terme sur $L' \cup \{f\}$, de profil (i,j), on définit au paragraphe 4 un nouveau symbole f par l'axiome

$$f\,x \equiv \tau(f)\,[x] \qquad (1)$$

Cette adjonction définit une extension de $\mathcal{S}$ dans une structure $\mathcal{S}'$. Pour chaque information complète I, soit I' l'extension de I dans $\mathcal{S}'$. Si u est un terme de S_i, calculer $\underline{f}u$ dans I' c'est démontrer dans I' un théorème $\underline{f}u \equiv v$ avec $v \in S_j$. On se pose alors les questions suivantes :

i) Peut-on toujours calculer $\underline{f}u$? La réponse est négative. Considérer par exemple l'équation $\underline{f}x \equiv \underline{f}x$.

ii) Si $\underline{f}u$ est calculable, existe-t-il un calcul utilisant uniquement :

les théorèmes égalitaires de I,

l'équation (1),

la règle de substitution,

la règle d'égalité (cf 2.1.4).

Là encore, la réponse est partiellement négative. Nous montrerons en effet que l'équation $a \equiv cond(a,0,1,0)$ admet comme seul point fixe $a \equiv \omega$ mais que ce résultat ne peut être obtenu par une suite de substitutions. Cependant, si $fu \equiv v \in I'$ avec $v \not\equiv \omega \in I$, les axiomes et règles précédents suffisent.

4.2.- Théorie du point fixe (MANNA - NESS - VUILLEMIN, SCOTT)

Soit E un ensemble, $\perp$ un élément de cet ensemble. E peut être ordonné par la relation

$$x < y \iff x = y \text{ ou } x = \perp$$

On dit que E est muni de la relation d'ordre discrète.

Soit E' un deuxième ensemble. A toute fonction F (non partout définie) de E' dans $E - \{\perp\}$, nous associons l'application $\tilde{F}$ de E' dans E définie par

$$\tilde{F}(x) = \begin{cases} F(x) & \text{si } x \in Dom(F) \\ \perp & \text{sinon} \end{cases}$$

L'ensemble $\mathcal{F}(E',E)$ des applications de E' dans E peut, lui-même, être ordonné par

$$F < G \iff \text{pour tout } x \text{ de } E' \quad F(x) < G(x)$$

$\mathcal{F}(E',E)$ possède un plus petit élément : $\perp = \lambda x. \perp$. Enfin, toute suite croissante $(F_n)_{n \geq 0}$ de $\mathcal{F}(E',E)$ admet une borne supérieure F. Il suffit de poser

$$F(x) = \begin{cases} \perp & \text{si } F_n(x) = \perp \text{ pour tout } n \geq 0 \\ F_n(x) & \text{si } F_n(x) \neq \perp \end{cases}$$

Définition 1 : Un ensemble ordonné E est complètement inductif s'il admet un plus petit élément (noté $\perp$) et si toute suite croissante de E possède une borne supérieure.

Définition 2 : Soit $(E, <)$ $(E', <')$ deux ensembles ordonnés, F une application de E dans E'. F est croissante si

$$\forall x,y \in E \quad x < y \implies F(x) <' F(y)$$

F est continue si, pour toute suite croissante $(x_n)_{n \geq 0}$ de E admettant une borne supérieure, la suite $(F(x_n))_{n \geq 0}$ admet une borne supérieure et

$$F(\sup_n x_n) = \sup_n F(x_n)$$

On note $\mathcal{F}_c(E,E')$ l'ensemble des applications continues de E dans E'.

Remarque : toute application continue est croissante. La réciproque est fausse.

Théorème 1.- (du point fixe) Soit E un ensemble complètement inductif, F une application continue de E dans E. F admet un plus petit point fixe x_F qui vérifie

$$x_F = \sup_n F^n(\perp)$$

Démonstration :

Pour tout $n \geqslant o$ $F^n(\perp) < F^{n+1}(\perp)$

Cela est vrai pour $n = o$. Si $F^n(\perp) < F^{n+1}(\perp)$, alors

$$F^{n+1}(\perp) < F^{n+2}(\perp)$$

Soit $x_F = \sup_n F_n(\perp)$

$$F(x_F) = F(\sup_n F_n(\perp)) = \sup_n F^{n+1}(\perp) = x_F.$$

x_F est donc un point fixe de F. De plus, si un élément x de E vérifie $F(x) < x$, on a :

$$\perp < x$$

et si $F^n(\perp) < x$, alors

$$F^{n+1}(\perp) < F(x) < x$$

Donc $x_F < x$

Remarques :

1) Ce théorème affirme l'existence d'un point fixe et montre comment le construire ; de plus celui qui est donné est le plus petit. En fait x_F est

le plus petit des éléments qui vérifient $F(x) < x$.

2) La fonctionnelle $\mathcal{T}_3$ définie en 4.1 par

$$\mathcal{T}_3(F) = \lambda x_1,x_2.\ \mathrm{Cond}(x_1,x_2,x_2+1,F(x_1,F(x_1-1,\ x_2+1)))$$

est continue sur $\mathcal{F}_c(\mathbb{N}_+^2, \mathbb{N}_+)$. Remarquons tout d'abord que $\mathcal{T}_3$ est définie à partir des applications suivantes

$$P : x \longmapsto \begin{cases} \perp & \text{si } x = \perp \text{ ou } x = 0 \\ x-1 & \text{si } x \geqslant 1 \end{cases}$$

$$S : x \longmapsto \begin{cases} \perp & \text{si } x = \perp \\ x+1 & \text{si } x \geqslant 0 \end{cases}$$

$$P_1 : (x_1,x_2) \longmapsto x_1$$

$$P_2 : (x_1,x_2) \longmapsto x_2$$

et Cond.

Chacune de ces applications est continue. Par exemple,

$$P_1(\perp, x_2) = \perp \quad \text{pour tout } x_2$$

et $$P_1(x_1,x_2) = P_1(x_1, \perp) = x_1$$

De manière générale, soit $E_1,\ldots,E_m$ des ensembles, munis de la relation d'ordre discrète. Pour tout $i = (i_1,\ldots,i_n) \in [m]^*$, E_i est ordonné par

$$x < y \iff (x_k < y_k \text{ pour } k \in [1,n])$$

Une application F de E_i dans E_j $(j \in [m])$ est continue si et seulement si elle vérifie la propriété :

Pour tout $k \in [1,n]$, tout $(x_1,\ldots,x_{k-1},x_{k+1},\ldots,x_n) \in \prod_{\ell \neq k} E_{i_\ell}$

ou bien $F(x_1,\ldots,x_{k-1}, \perp, x_{k+1},\ldots,x_n) = \perp$

ou bien, pour tout $x_k \in E_{i_k}$, $F(x_1,\ldots,x_n) = F(x_1,\ldots,x_{k-1}, \perp, x_{k+1},\ldots,x_n)$

Le fait que τ_3 soit une fonctionnelle continue est alors une conséquence du résultat général de (CADIOU), énoncé plus loin (4.3. Proposition 1).

4.3.- Notion de fonctionnelle

Définition 3 : Une structure $\mathcal{S}$ est ordonnée si

i) pour tout $j \in [m]$, L contient un symbole de constante ω_j de type j (dans la suite l'indice sera toujours omis).

ii) pour tout $i \in [m]^*$ tel que $q = |i| > 0$, pour tout g de L tel que source$(g) = i$ et pour tout $k \in [q]$, il existe un théorème de $\mathcal{S}$ de la forme

$$gy_1 \ldots y_{k-1}\, \omega\, y_{k+1} \ldots y_q \equiv \omega \vee gy_1 \ldots y_q \equiv gy_1 \ldots y_{k-1}\omega y_{k+1}\ldots y_q$$

Dans la suite du paragraphe, nous supposerons que $\mathcal{S}$ est une structure ordonnée. Il résulte alors du paragraphe précédent que, pour toute interprétation $R = (E,r)$ de $\mathcal{S}$, les applications $r(g)$ sont continues.

Définissons une structure $\mathcal{S}(f)$ en ajoutant à l'alphabet un symbole f de profil (i,j) $(i = (i_1,\ldots,i_n))$ et en n'ajoutant aucun axiome non logique. Les termes sur $L' \cup \{f\}$ formalisent la notion de fonctionnelle définie sur $\mathcal{F}(E_i,E_j)$. Précisément, soit I une information de $\mathcal{S}$ et $R = (E,r)$ une interprétation de I. Pour tout $F \in \mathcal{F}(E_i,E_j)$, on note R_F l'interprétation de $S(f)$ définie par

$$R_F = (E, r_F)$$

$$r_F(g) = r(g) \quad \text{pour } g \in L$$

$$r_F(f) = F$$

R_F est évidemment une interprétation stricte de $S(f)$. On a alors la définition suivante :

Définition 4 : (Fonctionnelle associée à un terme sur $L' \cup \{f\}$)

Soit $R = (E,r)$ une interprétation de I et τ un terme de profil (i',j') sur $L' \cup \{f\}$. La fonctionnelle associée à τ dans l'interprétation R est l'application $\hat{\hat{r}}(\tau)$ de $\mathcal{F}(E_i,E_j)$ dans $\mathcal{F}(E_{i'},E_{j'})$ définie par

$$\hat{\hat{r}}(\tau)(F) = \hat{r}_F(\tau) \quad \text{pour } F \in \mathcal{F}(E_i,E_j)$$

Nous pouvons énoncer un principe de récurrence pour les termes sur $L'(f)$. Pour cela, adoptons l'abréviation suivante : Si P est une propriété et si $\tau = (\tau_1,\ldots,\tau_n)$, on pose $P(\tau) = P(\tau_1)$ et ... et $P(\tau_n)$. En particulier, si $n = o$, $P(\tau) =$ vrai.

Principe de récurrence

Soit $x = (x_1,\ldots,x_n) \in V^n$ et $P(\tau)$ une propriété sur l'ensemble des termes construits sur $L(x) \cup \{f\}$.

On suppose que :

. $P(x)$ est vraie

. Si g est un symbole distinct de f et si $P(\tau)$ est vraie, $P(g\tau)$ est vraie

. Si $P(\tau)$ est vraie, $P(f\tau)$ est vraie.

Alors $P(\tau)$ est vraie pour tout terme τ sur $L(x) \cup \{f\}$.

Notations : On notera par la suite τ, σ, ρ les termes et, plus généralement, les suites de termes sur $L' \cup \{f\}$. Pour une interprétation donnée de $\mathcal{S}$, on pose $\mathcal{T} = \hat{\hat{r}}(\tau)$. Si $\tau = (\tau_1, \ldots, \tau_n)$, on pose $\hat{\hat{r}}(\tau) = (\hat{\hat{r}}(\tau_1), \ldots, \hat{\hat{r}}(\tau_n))$.

Avec ces notations, on peut écrire :

si $\tau = x_k$, $\mathcal{T} = \lambda F, x. x_k$

si $\tau = g\tau'$, $\mathcal{T} = r(g)(\mathcal{T}')$

si $\tau = f\tau'$, $\mathcal{T} = \lambda F. F(\mathcal{T}'(F))$

Définition 5 : (profil d'un terme)

Soit $i = (i_1, \ldots, i_n) \in [m]^*$, $j \in [m]$, h un terme sur $L' \cup \{f\}$. On dira que h est de profil (i,j) si h est de type j et s'il existe un n-uple $x = (x_1, \ldots, x_n)$ de V_i tel que h soit un terme sur $L(x) \cup \{f\}$.

Proposition 1 : Soit τ un terme de profil (i',j') sur $L' \cup \{f\}$. $\mathcal{T} = \hat{\hat{r}}(\tau)$ est une fonctionnelle continue de $\mathcal{F}_c(E_i, E_j)$ dans $\mathcal{F}_c(E_{i'}, E_{j'})$.

La démonstration, par récurrence sur τ , est faite dans (CADIOU)

Définition 6 : (substitution par un terme de même profil que f)

Soit τ et h des termes sur $L' \cup \{f\}$ tels que h soit de profil (i,j). $\tau(h)$ est définie par récurrence sur la longueur de τ .

Si $\tau \in V$, $\tau(h) = \tau$

Si $\tau = g\tau_1 \ldots \tau_q$, $\tau(h) = g\tau_1(h) \ldots \tau_q(h)$

Si $\tau = f\tau_1 \ldots \tau_n$, $\tau(h) = h[\tau_1(h), \ldots, \tau_n(h)]$

Si $\tau = (\tau_1, \ldots, \tau_n)$, on pose $\tau(h) = (\tau_1(h), \ldots, \tau_n(h))$

Proposition 2 : Si h est un terme sur L', $\tau(h)$ est un terme sur L', de même profil que τ et

$$\hat{r}(\tau(h)) = \mathcal{T}(\hat{r}(h)) \quad \text{pour toute interprétation.}$$

Démonstration : par récurrence sur τ

Si $\tau \in V$, $\tau(h) = \tau$ et $\mathcal{T}(\hat{r}(h)) = \hat{r}(\tau)$

Si $\tau = g\tau'$, $\tau(h) = g\tau'(h)$

$$\hat{r}(\tau(h)) = r(g)(\hat{r}(\tau'(h)))$$

$$= r(g)(\mathcal{T}'(\hat{r}(h))) \quad \text{par hypothèse de récurrence.}$$

$$= \mathcal{T}(\hat{r}(h))$$

Si $\tau = f\tau'$, $\tau(h) = h[\tau'(h)]$

$$\hat{r}(\tau(h)) = \hat{r}(h)(\hat{r}(\tau'(h)))$$

$$= \hat{r}(h)(\mathcal{T}'(\hat{r}(h))) \quad \text{par hypothèse de récurrence.}$$

$$= \mathcal{T}(r(h))$$

Nous avons également besoin du résultat suivant.

Proposition 3 : (composition des substitutions)

Soit σ et τ deux termes sur $L' \cup \{f\}$ tels que τ soit de profil (i,j). Notons $\sigma \circ \tau$ le terme $\sigma(\tau)$. Alors pour tout terme h sur $L' \cup \{f\}$, de profil (i,j)

$$\sigma \circ \tau\,(h) = \sigma(\tau(h)).$$

Démonstration :

Par récurrence sur σ (Si $\sigma' = (\sigma_1,\dots,\sigma_q)$, on pose $\sigma' \circ \tau = (\sigma_1 \circ \tau, \dots, \sigma_q \circ \tau)$).

. Si $\sigma \in V$, $\sigma \circ \tau = \sigma$. Donc $\sigma \circ \tau\ (h) = \sigma = \sigma(\tau(h))$

. Si $\sigma = g\,\sigma'$ $(g \neq f)$, $\sigma \circ \tau = g(\sigma' \circ \tau)$, donc

$$\sigma \circ \tau\,(h) = g(\sigma' \circ \tau(h)) = g(\sigma'(\tau(h))) = \sigma(\tau(h))$$

. Si $\sigma = f\,\sigma'$, $\sigma \circ \tau = \tau[\sigma' \circ \tau]$.

$\sigma(\tau(h)) = \tau(h)\,[\sigma'(\tau(h))]$. Il suffit donc de prouver que

$$\tau[\sigma' \circ \tau](h) = \tau(h)\,[\sigma' \circ \tau\,(h)]$$

Lemme 1 : $\tau[\rho](h) = \tau\,(h)\,[\rho(h)]$.

Démonstration : par récurrence sur τ.

. Si $\tau = x_k$ et $\rho = (\rho_1,\dots,\rho_q)$, $\tau[\rho] = \rho_k$ et $\tau(h) = x_k$.

$$\tau[\rho](h) = \rho_k(h) = \tau(h)\,[\rho(h)]$$

. Si $\tau = g\,\tau'$, $\tau[\rho] = g\,\tau'[\rho]$

$$\tau[\rho](h) = g\,\tau'\ [\rho](h) = g\,\tau'(h)\,[\rho(h)] = \tau(h)\,[\rho(h)]$$

. Si $\tau = f\,\tau'$, $\tau[\rho] = f\,\tau'[\rho]$

$$\tau[\rho](h) = h\,[\tau'\,[\rho]\ (h)] = h\,[\tau'(h)\ [\rho(h)]]$$

(hypothèse de récurrence)

$$= h\,[\tau'(h)]\,[\rho(h)] \quad \text{(associativité de la substitution)}$$

$$= \tau(h)\,[\rho(h)].$$

Règle de substitution généralisée

Nous avons vu que si p est une formule sur $L(x)$ et si $a_1,\dots,a_n$ sont des symboles de constantes nouveaux, de même type que $x_1,\dots,x_n$ respectivement, p est un théorème de $\mathcal{F}$ si et seulement si $p[a_1,\dots,a_n]$ est un théorème de $\mathcal{F}[a_1,\dots,a_n]$ (lemme des constantes (2.3)). De plus, si $u \in S_i'$ et si p est un théorème de $\mathcal{F}$, $p[u]$ est un théorème de $\mathcal{F}$. Nous démontrons maintenant un résultat du même type pour le symbole f. Nous avons besoin de la définition suivante.

Définition 7 : Soit h un terme de profil (i,j), p une formule sur $L' \cup \{f\}$; $p(h)$ est définie par récurrence sur p par les formules :

$$(\sigma \equiv \tau)(h) = \sigma(h) \equiv \tau(h)$$

$$(\neg p)(h) = \neg p(h)$$

$$(p \supset q)(h) = p(h) \supset q(h)$$

Si Γ est un ensemble de formules, on pose $\Gamma(h) = \{p(h)\ ;\ p \in \Gamma\}$

Proposition 4 : (Règle de substitution)

Soit I une information de $\mathcal{S}$, Γ un ensemble de formules et q une formule sur $L' \cup \{f\}$. Si $q \in I[\Gamma]$, $q(h) \in I[\Gamma(h)]$ pour tout terme h de profil (i,j).

Démonstration : Par récurrence sur les démonstrations.

. Si $q \in \Gamma$, $q(h) \in \Gamma(h)$

. Si q est un axiome propositionnel, $q(h)$ est un axiome propositionnel.

. Si q est un axiome d'égalité de $\mathcal{S}$, $q(h) = q$.

. Si $q = x_1 \equiv y_1 \supset \ldots \supset x_n \equiv y_n \supset fx \equiv fy$,

$$q(h) = x_1 \equiv y_1 \supset \ldots \supset x_n \equiv y_n \supset h[x] \equiv h[y]$$

et $q(h)$ est un théorème de I.

. Si $p(h)$ et $p(h) \supset q(h)$ appartiennent à $I(\Gamma(h))$, il en est de même de $q(h)$.

. Si $q = p[u]$ et si $p(h) \in I(\Gamma(h))$, il faut vérifier que

$$q(h) = p(h)[u(h)]$$

Il suffit, pour cela, de montrer que pour tout terme σ

$$\sigma[u](h) = \sigma(h)[u(h)].$$

Ce dernier point résulte directement du lemme 1.

Corollaire : Soit p une formule et τ un terme de profil (i,j) sur $L' \cup \{f\}$. Si $p(\tau) \in I[p]$, alors $p(\tau(h)) \in I[p(h)]$ pour tout terme h de profil (i,j).

Il suffit de remarquer que $p(\tau(h)) = p(\tau)(h)$, ce qui résulte immédiatement de la proposition 3.

4.4.- Equations récursives

Définition 8 : Soit $\mathcal{S}$ une structure d'information ordonnée. Une équation récursive sur $\mathcal{S}$, de profil (i,j), est un couple $(\underline{f}, X_{\underline{f}})$ où $\underline{f}$ est

un symbole n'appartenant pas à $L \cup \{f\}$, de profil (i,j) et $X_{\underline{f}} = \{\underline{f}x \equiv \tau(\underline{f})[x]\}$ (*) (τ est un terme sur $L(x) \cup \{f\}$).

Exemple : Dans la structure $\mathcal{N}$ des entiers considérons l'équation récursive

$$\text{fact } x \equiv \underline{\text{si}}\ \ x \equiv 0\ \ \underline{\text{alors}}\ \ 1\ \ \underline{\text{sinon}}\ \ x * \text{fact}(x-1)$$

associée au terme $\tau = \underline{\text{si}}\ \ x = 0\ \ \underline{\text{alors}}\ \ 1\ \ \underline{\text{sinon}}\ \ x * f(x-1)$, noté encore $\text{cond}(x, 0, 1, x * \text{fact}(x-1))$.

Les axiomes de cond doivent être modifiés de manière à rendre la structure $\mathcal{N}$ ordonnée. On peut considérer les axiomes suivants :

$$x \not\equiv \omega \supset \text{cond}(x,x,z,t) \equiv z$$

$$x \not\equiv \omega \wedge y \not\equiv \omega \wedge x \not\equiv y \supset \text{cond}(x,y,z,t) \equiv t$$

$$\text{cond}(\omega,y,z,t) \equiv \omega$$

$$\text{cond}(x,\omega,z,t) \equiv \omega$$

Une équation récursive $(\underline{f}, X_{\underline{f}})$ définit une extension $\mathcal{S}'$ de $\mathcal{S}$ (cf 3.1). Pour toute information I de $\mathcal{S}$, soit I' l'extension de I dans $\mathcal{S}'$. Si $u \in S_i$, $\underline{f}u$ est calculable dans I' s'il existe $v \in S_j$ tel que $\underline{f}u = v \in I'$. Un calcul de $\underline{f}u$ est une démonstration de $\underline{f}u \equiv v$.

Si I est consistante, soit $R = (E,r)$ une interprétation fixée de I. Pour toute application F de $\mathcal{F}(E_i,E_j)$, notons encore R_F le prolongement

---

* Nous utiliserons la notation $\tau(\underline{f})[x]$ au lieu de $\tau(\underline{f}x)$

de R obtenu en posant $r(\underline{f}) = F$. Alors R_F est une interprétation de I' si et seulement si F est un point fixe de la fonctionnelle τ associée à τ.

Nous pouvons donc énoncer le théorème suivant.

Théorème 2.- Si I est consistante, I' est consistante. Si τ admet plusieurs points fixes distincts, I' n'est pas complète. Plus précisément, soit $u \in S_i$, $x = \hat{r}(u)$. S'il existe deux points fixes F_1 et F_2 de τ tels que $F_1(x) \neq F_2(x)$, alors $\underline{f}u$ n'est pas calculable ; de plus $\underline{f}u \neq \omega \notin I'$.

Démonstration :

1) Si I est consistante, elle admet une interprétation R. D'après la proposition 1, τ est une fonctionnelle continue et admet un plus petit point fixe F_τ. R_{F_τ} est donc une interprétation de I'.

2) Soit $u \in S_i$, $x = \hat{r}(u)$. Si $\underline{f}u \equiv v \in I'$, pour tout point fixe F, $F(x) = \hat{r}(v)$. Si $\underline{f}u \neq \omega \in I'$, $F_\tau(x) \neq \perp$. Donc, pour tout point fixe F de τ, $F(x) = F_\tau(x)$.

En définitive si F_1, F_2 sont deux points fixes distincts de τ, soit $x \in E_i$ tel que $F_1(x) \neq F_2(x)$ et $u \in S_i$ tel que $x = \hat{r}(u)$. Alors $\underline{f}u \equiv \omega \notin I'$ et $\underline{f}u \neq \omega \notin I'$. I' est donc incomplète.

Commentaire : Si τ admet plusieurs points fixes, I' n'est pas complète. Réciproquement, si τ admet un point fixe unique, I' est-elle complète ? On ne peut apporter ici qu'une réponse partielle : si $F_\tau(x) \neq \perp$ (et si $\hat{r}(u) = x$), $\underline{f}u$ est calculable (théorème 4). Par contre, si tous les points fixes de τ vérifient $F(x) = \perp$, nous ne pouvons assurer que $\underline{f}u \equiv \omega \in I'$. Le problème est le suivant : existe-t-il des interprétations R' de I' qui ne soient pas de la forme $R' = R_F$ où R est une interprétation de I ?

Points fixes calculables

Définition 9 : Soit $u \in S_i$. $\underline{f}u$ est S-calculable s'il existe un calcul $(a_0,\ldots,a_n)$ de $\underline{f}u$, formé de formules égalitaires tels que, pour $k = 0,1,..,n$

- $a_k \in I \cup X_{\underline{f}}$ ou bien
- $a_k = a_j\ [u]$ $(j < k)$ ou bien
- a_k est déduite de formules $a_j, a_{i_1},\ldots,a_{i_n}$ précédentes

en utilisant la règle d'égalité (cf 2.1.4)

On peut définir une suite $(f_n)_{n \geqslant 0}$ de termes sur $L(x)$ en posant

$$f_0 = \omega$$

et $$f_{n+1} = \tau(f_n) \quad \text{pour } n \geqslant 0$$

Exemple : Considérons l'équation récursive

$$\text{fact } n \equiv \underline{si}\ n = 0\ \underline{alors}\ 1\ \underline{sinon}\ n * \text{fact}(n-1)$$

associée au terme $\tau = \underline{si}\ n = 0\ \underline{alors}\ 1\ \underline{sinon}\ n * f(n-1)$

Alors,

$$f_0[1] = \omega$$

$$f_1[1] = \tau(\omega)[1] = \underline{si}\ 1 = 0\ \underline{alors}\ 1\ \underline{sinon}\ 1 * \omega$$

$$f_2[1] = \tau^2(\omega)[1] = \underline{si}\ 1 = 0\ \underline{alors}\ 1\ \underline{sinon}\ 1 * [\underline{si}\ 1-1 = 0\ \underline{alors}\ 1\ \underline{sinon}\ (1-1) * \omega]$$

Calculons également $\tau^2[1]\ (= \tau(\tau)[1])$

$$\tau^2[1] = \underline{si}\ 1 = 0\ \underline{alors}\ 1\ \underline{sinon}\ 1 * [\underline{si}\ 1-1 = 0\ \underline{alors}\ 1\ \text{sinon}\ (1-1) * f(1-1-1)]$$

mais, $\text{cond}(1-1,0,1,(1-1) * \omega) = 1 \in I$ et

$$cond(1\text{-}1,0,1,(1\text{-}1) * f(1\text{-}1\text{-}1)) \equiv 1 \in I(f)$$

où $I(f)$ désigne l'extension de I dans $\mathcal{S}(f)$.

Donc $\tau^2(\omega)[1] \equiv 1 \in I$ et $\tau^2[1] \equiv 1 \in I(f)$

Et donc, en utilisant la proposition 4

$f_n[1] \equiv 1 \in I$ pour $n \geqslant 2$ (En effet, $f_n[1] = \tau^2(\tau^{n-2}(\omega))[1]$)

et $fact(1) \equiv 1 \in I'$ (En effet $fact(1) \equiv \tau^2(fact)[1] \in I'$)

$fact(1)$ est donc S-calculable.

De plus, on voit sur cet exemple le lien entre un calcul ascendant de $fact(1)$ (Calcul de $f_0[1]$, $f_1[1]$, ... $f_n[1]$, ...) et un calcul descendant de $fact(1)$ (Calcul de $fact(1)$, $\tau(fact)[1]$, ..., $\tau^n(fact)[1]$, ...).

Le calcul de $fact(1)$ repose sur le fait suivant : $\mathcal{S}$ étant une structure ordonnée,

$$cond(0,0,1,\omega) \equiv 1 \in I \Longrightarrow cond(0,0,1,x) \equiv 1 \in I$$

Théorème 3.- Soit I une information complète, $u \in S_i$ et $n \geqslant 0$. Si $f_n[u] \neq \omega \in I$, $\underline{f}u \equiv f_n[u] \in I'$. De plus $\underline{f}u$ est S-calculable et $f_{n+k}[u] \equiv f_n[u] \in I$.

Démonstration :

Par définition $f_n = \tau_0 \ldots \circ \tau(\omega) = \tau^n(\omega)$. D'autre part, en appliquant n fois l'équation récursive, $\underline{f}u \equiv \tau^n(\underline{f})[u] \in I'$. Nous avons donc besoin du résultat suivant.

Lemme 2 : Soit σ un terme sur $L(x) \cup \{f\}$. Si $\sigma(\omega)[u] \neq \omega \in I$, alors $\sigma[u] \equiv \sigma(\omega)[u] \in I(f)$.

Démonstration : par récurrence sur σ

. Si $\sigma \in \{x_1, \ldots, x_n\}$, $\sigma = \sigma(\omega)$

. Si $\sigma = g\sigma_1 \ldots \sigma_q$ $(g \neq f)$,

$$\sigma(\omega)[u] = g(\sigma_1(\omega)[u], \ldots, \sigma_q(\omega)[u])$$

Rappelons qu'il existe $y = (y_1 \ldots y_q)$ tel que

$$g(y_1, \ldots, \omega, \ldots, y_q) \neq \omega \supset g(y_1, \ldots, y_q) \equiv g(y_1, \ldots, \omega, \ldots, y_q)$$

soit un théorème de I. Donc, pour toute suite $(u_1, \ldots, u_{k-1}, u_{k+1}, \ldots, u_q)$ de termes sur $L(x)$, si $g(u_1, \ldots, \omega, \ldots, u_q) \neq \omega \in I$, alors

$$g(u_1, \ldots, y_k, \ldots, u_q) \equiv g(u_1, \ldots, \omega, \ldots, u_q) \in I$$

Supposons, pour simplifier, que, pour $k = 1, \ldots, r$, $\sigma_k(\omega)[u] \equiv \omega \in I$ et, pour $k = r+1, \ldots, q$ $\sigma_k(\omega)[u] \equiv \omega \notin I$ et donc $\sigma_k(\omega)[u] \neq \omega \in I$ (puisque I est complète).

Alors, par hypothèse de récurrence,

$$\sigma_k[u] \equiv \sigma_k(\omega)[u] \in I(f) \text{ pour } r < k \leqslant q$$

et, donc

$$g(\sigma_1(\omega)[u], \ldots, \sigma_q(\omega)[u]) \equiv g(y_1, \ldots, y_r, \sigma_{r+1}[u], \ldots, \sigma_q[u]) \in I(f)$$

et $$\sigma(\omega)[u] \equiv \sigma[u] \in I(f).$$

. Enfin, si $\sigma = f\sigma'$, $\sigma(\omega) = \omega$

Revenons à la démonstration du théorème. D'après la proposition 4, puisque $\tau^n[u] \equiv \tau^n(\omega)[u] \in I(f)$, $\tau^n(h)[u] \equiv \tau^n(\omega)[u] \in I$ pour tout terme h sur $L(x)$. En particulier $f_{n+k} = \tau^n(\tau^k(\omega))$. Donc $f_{n+k}[u] \equiv f_n[u] \in I$ pour $k \geqslant 0$.

Par ailleurs $\tau^n(\underline{f})[u] \equiv \tau^n(\omega)[u] \in I'$ et donc $\underline{fu} \equiv f_n[u] \in I'$. Enfin, il ressort de la démonstration du lemme que $\underline{fu}$ est S-calculable.

En étudiant la démonstration du théorème 3, on peut préciser le calcul de $\underline{fu}$ lorsque $\underline{fu}$ est S-calculable. Reprenant la terminologie de (CADIOU), nous adoptons les définitions suivantes.

Soit σ un terme sur $L \cup \{\underline{f}\}$. Effectuer une substitution dans σ c'est remplacer certains termes dans σ de la forme $\underline{fu}$ par $\tau(\underline{f})[u]$. Nous avons étudié précédemment les substitutions complètes ou remplacements de toutes les occurrences de $\underline{f}$.

Soit σ' un sous terme de σ, de la forme $g\,\sigma_1 \dots \sigma_q$. Si les occurrences de $\underline{f}$ dans σ' sont contenues dans $\sigma_{i_1}, \dots, \sigma_{i_r}$, soit σ'' le terme obtenu en remplaçant σ_{i_1} par $y_1, \dots, \sigma_{i_r}$ par y_r ($y_1, \dots, y_r$ étant des variables de type convenable). On dit que σ' est simplifiable s'il existe $v \in S$ tel que $\sigma'' \equiv v \in I$. On simplifie alors σ' en le remplaçant par v dans σ (le choix de v n'est pas unique).

Exemple : $\sigma' = \mathrm{cond}(1,1,3,\underline{fu})$ est simplifiable. On peut alors remplacer σ' par 3.

$\sigma' = \mathrm{cond}(\underline{fu}, \underline{fu}, 3, 7)$ n'est pas simplifiable.

Il ressort alors de la démonstration que, si $f_n[u] \not\equiv \omega$ appartient à I pour n assez grand, il existe un calcul de $\underline{fu}$ composé uniquement de substitutions et de simplifications.

Théorème 4.- (théorème fondamental)

Soit I une information complète, R = (E,r) une interprétation de I, τ la fonctionnelle associée à τ dans R. Soit $u \in S_i$ et $x = \hat{r}(u)$.

Alors les propriétés suivantes sont équivalentes :

i) $\underline{fu} \not\equiv \omega \in I'$

ii) $F_\tau(x) \neq \perp$

iii) il existe $n \geq 0$ tel que $f_n[u] \not\equiv \omega \in I$ et donc tel que $f_n[u] \equiv \underline{fu} \in I'$.

Démonstration :

i) $\Rightarrow$ ii) est évident puisque R_{F_τ} est une interprétation de I'

ii) $\Rightarrow$ iii) Soit $(F_n)_{n \geq 0}$ la suite définie par

$$F_0 = \perp$$

$$F_n = \tau(F_{n-1}) \quad n > 0$$

Puisque $F_\tau(x) \neq \perp$, il existe n tel que $F_n(x) \neq \perp$. D'après la proposition 2, $F_n = \hat{r}(f_n)$; donc $F_n(x) = \hat{r}(f_n[u])$. Donc $f_n[u] \equiv \omega \notin I$ et, puisque I est complète, $f_n[u] \not\equiv \omega \in I$. La seconde partie de l'assertion résulte du théorème 3.

iii) $\Rightarrow$ i) est évident.

On déduit immédiatement du théorème fondamental le théorème suivant.

Théorème 5.- (Théorème du plus petit point fixe)

Soit I une information complète de $\mathcal{S}$, $\underline{fx} \equiv \tau(\underline{f})[x]$ une équation récursive sur $\mathcal{S}$ et I' l'extension de I associée. Soit $\Gamma = \{\underline{fu} = \omega, \underline{fu} \not\equiv \omega \notin I'\}$. Alors $I'' = I'(\Gamma)$ est une extension complète de I'. De plus, si R est une interprétation de I et si F est le plus petit point fixe de la fonctionnelle associée à τ dans R, R_F est l'unique interprétation de I'' prolongeant R.

Démonstration :

. I" est consistante. En effet, soit R une interprétation de I et $F = F_{\tau}$. D'après le théorème fondamental, $\underline{fu} \neq \omega \in I' \Longrightarrow F(\hat{r}(u)) = \perp$. Donc R_F est une interprétation de I".

. I" est complète. Il suffit pour cela de vérifier que, pour tout $u \in S_i$, il existe $v \in S_j$ tel que $\underline{fu} \equiv v \in I"$. Or, si $\underline{fu} \neq \omega \in I'$, il existe $n \geq 0$ tel que $\underline{fu} \equiv f_n[u] \in I'$ et si $\underline{fu} \neq \omega \notin I'$, $\underline{fu} \equiv \omega \in I"$.

. Si R' est une interprétation de I", prolongeant R

$$r'(\underline{f})(x) = \hat{r}'(\underline{fu}) = \hat{r}'(v) = \hat{r}(v)$$

$r'(\underline{f})$ est donc déterminé de manière unique.

Règle point fixe

Nous décrivons maintenant une règle d'inférence qui est l'équivalent de la "règle calculatoire" de Scott, Manna, Vuillemin.

Théorème 6.- Soit p une formule sur $L' \cup \{f,\underline{f}\}$. On suppose que

i) $p(\omega) \in I"$

ii) $p(\tau(f)) \in I"[p]$ (autrement dit $p(\tau(f))$ peut être déduit de p et de I")

Alors, pour toute suite de termes u sans variable sur $L \cup \{\underline{f}\}$, $p(\underline{f})[u]$ appartient à I".

Démonstration : Il suit des hypothèses que $p(f_n) \in I"$ pour tout $n \geq 0$. Il suffit donc de démontrer le lemme suivant.

Lemme 3 : Soit p une formule sans variable sur $L \cup \{f,\underline{f}\}$. Il existe $\ell_0 \geq 0$ tel que $p(\underline{f}) \longleftrightarrow p(f_\ell) \in I"$ pour tout $\ell \geq \ell_0$.

Démonstration : Démontrons tout d'abord que pour tout terme sans variable τ sur $L \cup \{f,\underline{f}\}$, il existe $\ell_0 \geq 0$ tel que $\tau(\underline{f}) \equiv \tau(f_\ell) \in I"$ pour $\ell \geq \ell_0$.

. Si $g \in L \cup \{\underline{f}\}$ et $\tau = g\tau_1 \ldots \tau_q$, il existe $\ell_0 \geq 0$ tel que pour $\ell \geq \ell_0$ et pour $k = 1,\ldots,q$, $\tau_k(\underline{f}) \equiv \tau_k(f_\ell) \in I"$. Donc pour $\ell \geq \ell_0$, $\tau(\underline{f}) \equiv \tau(f_\ell) \in I"$.

. Si $\tau = f\tau'$ avec $\tau' = (\tau_1,\ldots,\tau_n)$, $\tau(\underline{f}) = \underline{f}\,\tau'(\underline{f})$ et $\tau(f_\ell) = f_\ell\,[\tau'(f_\ell)]$. Il existe ℓ_0 tel que, pour $\ell \geq \ell_0$ et pour $k = 1,\ldots,n$, $\tau_k(\underline{f}) \equiv \tau_k(f_\ell) \in I"$.

Pour $\ell \geq \ell_0$, $\tau(\underline{f}) \equiv \underline{f}\ \tau'(f_\ell) \in I"$.

Soit $u = \tau'(f_{\ell_0})$.

Deux cas sont à envisager :

. Si $\underline{fu} \neq \omega \in I"$, il existe ℓ_1 tel que, pour $\ell \geq \ell_1$, $\underline{fu} \equiv f_\ell[u] \in I"$. Alors pour $\ell \geq \max(\ell_0, \ell_1)$,

$$\underline{fu} \equiv \underline{f}\ \tau'(f_\ell) \in I"$$

$$\underline{fu} \equiv f_\ell[u] \in I"$$

$$f_\ell[u] \equiv f_\ell\,[\tau'(f_\ell)] \in I".$$

Donc $\tau(\underline{f}) \equiv \tau(f_\ell) \in I"$ pour $\ell \geq \max(\ell_0, \ell_1)$.

. Si $\underline{fu} \equiv \omega \in I"$, alors, pour tout $\ell \geq 0$, $f_\ell[u] \equiv \omega \in I"$ et donc, pour $\ell \geq \ell_0$, $\tau(\underline{f}) \equiv \tau(f_\ell) \in I"$.

Démontrons maintenant, par récurrence sur p, le résultat du lemme.

. Si $p = \sigma \equiv \tau$, il existe $\ell_0 \geq 0$ tel que, pour $\ell \geq \ell_0$,

$$\sigma(\underline{f}) \equiv \sigma(f_\ell) \in I" \text{ et } \tau(\underline{f}) \equiv \tau(f_\ell) \in I".$$

Donc, pour $\ell \geq \ell_o$, $p(\underline{f}) \leftrightarrow p(f_\ell) \in I''$.

. Si $p = \neg q$ il existe $\ell_o \geq 0$ tel que, pour $\ell \geq \ell_o$, $q(\underline{f}) \leftrightarrow q(f_\ell) \in I''$. Alors, pour $\ell \geq \ell_o$, $p(\underline{f}) \leftrightarrow p(f_\ell) \in I''$.
De même si $p = q \supset r$.

Donnons, pour finir un exemple d'équation récursive pour laquelle on peut trouver u tel que $\underline{f}u$ soit calculable sans être S-calculable.

Exemple :

Considérons la structure d'information $\mathcal{S}$ d'alphabet $L = \{0, 1, \omega, \text{cond}\}$ avec un seul type d'objet et admettant pour axiomes la formule $0 \neq 1 \wedge 0 \neq \omega \wedge 1 \neq \omega$ et les axiomes de cond décrits à la page 4.15 . Soit I l'ensemble des théorèmes de $\mathcal{S}$. I est complète.

Considérons alors le symbole 0-aire a et l'équation récursive

$$a \equiv \text{cond}(a, 0, 1, 0) \quad .$$

Montrons que a est calculable et, plus précisément, que $a \equiv \omega \in I'$.

Nous écrivons $\vdash p$ au lieu de $p \in I'$.

D'après les axiomes de cond,

$$\vdash a \not\equiv \omega \wedge a \equiv 0 \supset \text{cond}(a, 0, 1, 0) \equiv 1$$

soit $$\vdash a \equiv \omega \vee a \not\equiv 0 \vee \text{cond}(a, 0, 1, 0) \equiv 1.$$

Puisque $\vdash a \equiv \text{cond}(a, 0, 1, 0)$, il vient

$$\vdash a \equiv \omega \vee a \not\equiv 0 \vee a \equiv 1$$

et, puisque $\vdash a \equiv 1 \supset a \not\equiv 0$, on peut simplifier :

$$\vdash a \equiv \omega \vee a \not\equiv 0$$

On obtient de même

$$\vdash a \equiv \omega \vee a \equiv 0$$

soit $$\vdash a \equiv \omega \wedge (a \equiv 0 \vee a \not\equiv 0)$$

Donc, $$\vdash a \equiv \omega \quad .$$

a est donc calculable. Il est clair, cependant, que a n'est pas S-calculable. En effet aucun axiome ne permet de simplifier $\text{cond}(a, 0, 1, 0) = \tau(a)$, pas plus que $\tau^2(a)$, $\tau^3(a)$, ... , $\tau^n(a)$.

Conclusion : La théorie des équations récursives dans les structures d'information peut être comparée avec plusieurs autres théories.

1) Dans l'étude des schémas récursifs (GARLAND-LUCKHAM) on s'intéresse à un point de vue purement syntaxique. Seule la conditionnelle possède sa signification habituelle. Tous les autres symboles fonctionnels ou propositionnels sont interprétés librement. Certains auteurs (NIVAT) n'assignent pas à la conditionnelle sa signification habituelle.

2) Dans les travaux de (CADIOU, VUILLEMIN) on considère au contraire, une seule interprétation $\mathcal{D}$, chaque élément $\underline{c}$ de $\mathcal{D}$ étant d'ailleurs représenté par un symbole de constante c. On ne s'intéresse pas à la manière d'accéder à chaque élément de $\mathcal{D}$.

3) Dans le présent travail, on étudie les équations récursives sur des structures d'information. Cela signifie essentiellement que

a) On considère une classe d'interprétations, satisfaisant les axiomes de la structure. Si ces axiomes se réduisent aux formules définissant le symbole cond, on retrouve l'étude des schémas récursifs. Inversement, si l'on ajoute assez d'axiomes pour obtenir une information complète, on obtient

les équations récursives sur un domaine $\mathcal{D}$.

b) l'étude d'une équation récursive est entièrement menée au niveau formel. Ceci est possible parce que nous avons associé à une donnée une information complète. Ceci permet en particulier d'énoncer les règles de simplification au moyen d'axiomes de la structure d'information.

D'autre part, les équations récursives permettent de définir des accès nouveaux. Donnons en un exemple typique.

Accès associatif dans une liste

L'accès associatif est une fonction associant à une place x et une valeur y la première place x', si elle existe, derrière x, telle que v(x') = y ou la valeur nil si la liste s'achève sans que l'on ait rencontré y. Cet accès peut être défini formellement par un symbole ass et l'équation

ass(x,y) ≡ si vx = y alors x sinon si sx = nil alors nil sinon ass(sx,y)

En particulier, ass(nil,y) ≡ nil

Considérons une information I complète. On peut alors distinguer 2 cas.

1) si I est une liste finie, ass(x,y) est calculable (et même S-calculable) pour tout x,y et donc I' est une extension complète de I.

2) si I est une liste infinie et si $vs^k x \neq y \in I$ pour $k \geqslant 0$, ass(x,y) n'est pas calculable et I' n'est pas une extension complète de I.

Notons, pour finir, que notre formalisation permet de traiter essentiellement le cas où l'équation récursive admet un point fixe défini de manière unique. Nous obtenons cependant des résultats généraux sur le plus petit point fixe en considérant une ω-extension de I (théorème 5) et nous démontrons alors une règle d'induction du même type que la règle de Scott.

Il reste encore de nombreux points à étudier, en particulier à propos des différents types de calcul associés à une équation récursive (appel par valeur, par nom ...).

4.5.- Systèmes d'équations mutuellement récursives

L'extension à un système d'équations mutuellement récursives est absolument standard. Nous énonçons dans ce paragraphe les définitions et résultats correspondants sans effectuer les démonstrations.

Soit $f_1,\dots,f_k$ des symboles n'appartenant pas à L, de profil respectif $(i_1,j_1),\dots,(i_k,j_k)$ et τ un terme sur $L(x) \cup \{f_1,\dots,f_k\}$, de profil (i',j'). Posons $i = (i_1,\dots,i_k)$ et $j = (j_1,\dots,j_k)$. Si $R = (E,r)$ est une interprétation de S, soit $\mathcal{F}_{i,j} = \mathcal{F}(E_{i_1},E_{j_1}) \times\dots\times \mathcal{F}(E_{i_k},E_{j_k})$. Notons également S(f) l'ensemble des termes sans variable engendrés par $L \cup \{f_1,\dots,f_n\}$. Enfin si $F = (F_1,\dots,F_k)$ appartient à $F_{i,j}$, on note r_F l'extension de r défini par

$$r_F(f_\ell) = F_\ell \qquad \text{pour} \quad \ell \in [1,k]$$

$R_F = (E,r_F)$ est une interprétation de S(f).

Alors, si R est une interprétation fixée, on peut associer au terme τ une application $\mathcal{T}$ de $\mathcal{F}_{i,j}$ dans $\mathcal{F}(E_{i'}, E_{j'})$ en posant, pour tout $F \in \mathcal{F}_{i,j}$

$$\mathcal{T}(F) = r_F(\tau)$$

Plus généralement, si $\tau_1,\dots,\tau_k$ sont des termes de même profil que $f_1,\dots,f_k$ respectivement, on pose $\tau = (\tau_1,\dots,\tau_k)$ et $\mathcal{T} = (\mathcal{T}_1,\dots,\mathcal{T}_k)$.

τ est une fonctionnelle de $\mathcal{F}_{i,j}$ dans $\mathcal{F}_{i,j}$.

Réciproquement, étant donné des ensembles $E_1,\ldots,E_m$ et un sous-ensemble $\mathcal{L}$ de $\bigcup_{i,j} \mathcal{F}(E_i,E_j)$, on peut définir la notion de fonctionnelle engendrée par $\mathcal{L}$. Considérons pour cela un ensemble L de symboles tel qu'il existe une bijection r de L sur $\mathcal{L}$ pour laquelle $R = (E_1,\ldots,E_m,r)$ soit une interprétation de l'ensemble S des termes construits sur L.

Définition 10 : Soit i,j, i',j' comme ci-dessus. τ est une fonctionnelle de $\mathcal{F}_{i,j}$ dans $\mathcal{F}(E_{i'}, E_{j'})$ engendrée par $\mathcal{L}$ s'il existe des symboles $f_1,\ldots,f_k$ et un terme τ sur $L' \cup \{f_1,\ldots,f_k\}$ tel que τ soit la fonctionnelle associée à τ dans l'interprétation R.

Si $\tau = (\tau_1,\ldots,\tau_k)$ et si, pour $\ell = 1,\ldots,k$, τ_ℓ est une fonctionnelle de $\mathcal{F}_{i,j}$ dans $\mathcal{F}(E_{i\ell}, E_{j\ell})$ engendrée par $\mathcal{L}$, on dira encore que τ est une fonctionnelle (de $\mathcal{F}_{i,j}$ dans lui-même) engendrée par $\mathcal{L}$.

Nous utiliserons cette définition au chapitre suivant.

Si σ est un terme sur $L' \cup \{f_1,\ldots,f_k\}$ et si $\tau_1,\ldots,\tau_k$ sont des termes sur $L' \cup \{f_1,\ldots,f_k\}$ de même profil que $f_1,\ldots,f_k$ on note $\sigma(\tau_1,\ldots,\tau_k)$, ou encore $\sigma(\tau)$ ou $\sigma \circ \tau$, le terme obtenu en substituant dans σ τ_1 à $f_1,\ldots,\tau_k$ à f_k. On vérifie immédiatement que $\sigma \circ \tau\,(h) = \sigma(\tau(h))$ où $h = (h_1,\ldots,h_k)$.

La règle de substitution s'étend de même immédiatement (proposition 4).

Définition 11 : Soit $i = (i_1,\ldots,i_k)$, $j = (j_1,\ldots,j_k)$ comme ci-dessus. Un système d'équation récursive sur $\mathcal{S}$, de profil (i,j) est la donnée

d'un k-uple $\underline{f} = (\underline{f}_1,\ldots,\underline{f}_k)$ de symboles n'appartenant pas à $L \cup \{f_1,\ldots,f_k\}$, de profil respectif $(i_1,j_1),\ldots,(i_k,j_k)$ et d'un système d'axiomes

$$X_{\underline{f}} = \{\, \underline{f}_\ell\, x_\ell \equiv \tau_\ell(\underline{f}_\ell)[x_\ell] \;;\quad 1 \leqslant \ell \leqslant k \,\}$$

où pour $\ell = 1,\ldots k$, τ_ℓ est un terme sur $L(x_\ell) \cup \{f_1,\ldots,f_k\}$ et $x_\ell \in V_{i\ell}$.

Le système $(\underline{f}, X_{\underline{f}})$ définit une extension $\mathcal{S}'$ de $\mathcal{S}$. Pour toute information I de $\mathcal{S}$, soit I' l'extension de I dans $\mathcal{S}'$. Pour $\ell \in [1,k]$ et $u \in S_{i\ell}$ $\underline{f}_\ell u$ est calculable dans I' s'il existe $v \in S_{j\ell}$ tel que $\underline{f}_\ell u \equiv v \in I'$. On définit de même le S-calculabilité.

Enfin, on définit une suite $(f_n)_{n\geqslant 0}$ de k-uples de termes sur L' en posant

$$f_o = (\omega,\ldots,\omega)$$

et $$f_{n+1} = \tau(f_n) \qquad n \geqslant 0$$

Pour $n \geqslant 0$, on note $f_n = (f_{n1},\ldots,f_{nk})$.

Soit $(\underline{f}, X_{\underline{f}})$ une équation récursive, I une information de $\mathcal{S}$ et $R = (E,r)$ une interprétation de I. Désignons par $\mathcal{F}^c_{i,j}$ le sous-ensemble de $\mathcal{F}_{i,j}$ des fonctions continues. La fonctionnelle τ associée à τ dans R est une application continue de $\mathcal{F}^c_{i,j}$ dans lui-même. Le théorème 1 s'applique donc à τ et l'on peut en déduire le théorème suivant.

Théorème 2'.- Si I est consistante, I' est consistante. Si τ admet plusieurs points fixes distincts, I' n'est pas complète. Plus précisément, soit $\ell \in [1,k]$, $u \in S_{i\ell}$ et $x = \hat{r}(u)$. S'il existe deux points fixes F et G de τ tels que $F_\ell(x) \neq G_\ell(x)$, alors $\underline{f}_\ell u$ n'est pas calculable ; de plus $\underline{f}_\ell u \neq \omega \notin I'$.

On obtient de même

Théorème 3'.- Soit I une information complète, $\ell \in [1,k]$, $u \in S_{i\ell}$ et $n \geqslant 0$. Si $f_{n\ell}[u] \not\equiv \omega \in I$, $\underline{f}_\ell u \equiv f_{n\ell}[u] \in I'$. De plus $\underline{f}_\ell u$ est S-calculable et $f_{p\ell}[u] \equiv f_{n\ell}[u] \in I$ pour $p \geqslant n$.

Théorème 4'.- Soit I une information complète, $R = (E,r)$ une interprétation de I, $\ell \in [1,k]$, $u \in S_{i\ell}$ et $x = \hat{r}(u)$. Soit F le plus petit point fixe de τ. Alors, les propriétés suivantes sont équivalentes

i) $\underline{f}_\ell u \not\equiv \omega \in I'$

ii) $F\ell(x) \neq \perp$

iii) il existe $n \geqslant 0$ tel que $f_{n\ell}[u] \not\equiv \omega \in I$ et, donc, tel que $\underline{f}_\ell[u] \equiv f_{n\ell}[u] \in I'$.

Théorème 5'.- Soit I, I' comme précédemment et $\Gamma = \{\underline{f}_\ell u \equiv \omega \ ; \ \ell \in [1,k]$ et $\underline{f}_\ell u \not\equiv \omega \notin I'\}$; $I'' = I'[\Gamma]$ est une extension complète de I'. De plus, si R est une interprétation de I et F le plus petit point fixe de τ, R_F est l'unique interprétation de I'' prolongeant R.

Enfin, nous pouvons énoncer la règle point fixe suivante.

Théorème 6'.- Soit p une formule sur $L' \cup \{f_1,\ldots,f_k\} \cup \{\underline{f}_1,\ldots,\underline{f}_k\}$. On suppose que

1) $p(\omega,\ldots,\omega) \in I''$

2) $p(\tau(f)) \in I''(p)$

Alors pour toute suite de termes u, sans variable, $p(\underline{f})[u]$ appartient à I''.

5

modifications,

calculs, problèmes

5.- MODIFICATIONS, CALCULS, PROBLEMES

Comme nous l'avons dit dans l'introduction, le présent travail vise à donner un outil d'étude à la notion de problème et de programme. Cette étude n'est qu'à peine ébauchée et nous présentons simplement dans ce chapitre les définitions qui nous semblent les plus adéquates.

5.1.- Modifications élémentaires

Pour définir une structure d'information, il n'est pas suffisant de préciser les types d'objets manipulés et les accès à ces objets. Il est aussi important de spécifier les modifications permettant de passer d'une information à une autre. Rappelons les notations introduites au paragraphe 1.2.9.

Notations : Soit $\mathcal{F}$ un système fonctionnel, $\mathcal{I}(\mathcal{F})$ l'ensemble des informations de $\mathcal{F}$. Pour tout $n \geq 0$, on note $\mathcal{A}_n(\mathcal{F})$ l'ensemble des applications de $\mathcal{I}(\mathcal{F})^n$ dans $\mathcal{I}(\mathcal{F})$. Soit $\mathcal{A}(\mathcal{F}) = \bigcup_{n \geq 0} \mathcal{A}_n(\mathcal{F})$.

Définition 1.: Une structure d'information $\mathcal{S}$ est la donnée d'un système fonctionnel $\mathcal{F}$ et d'un sous ensemble $\mathcal{M}$ de $\mathcal{A}(\mathcal{F})$. Les éléments de $\mathcal{M}$ sont les modifications élémentaires de $\mathcal{S}$.

On pose $\mathcal{M}_n = \mathcal{M} \cap \mathcal{A}_n(\mathcal{F})$.

Une modification élémentaire associe à des ensembles $I_1,\ldots,I_n$ de théorèmes un autre ensemble I' de théorèmes. Une manière particulièrement intéressante de réaliser cette association est de "plonger" $I_1,\ldots I_n$ et I' dans une même structure d'information et de définir la modification par un système d'axiomes.

Etudions d'abord un exemple : on veut adjoindre, dans une liste un élément derrière $s^k t$ et lui assigner la valeur a. Intuitivement, cela revient à remplacer $s^{k+1}t$ par $s^{k+2}t$ etc ... Pour traduire cela, on introduit un nouveau symbole s_1 et on considère la bijection σ_1 de L sur $L_1 = L \cup \{s_1\} \setminus \{s\}$ telle que $\sigma_1(s) = s_1$ et laissant fixes les autres éléments de L. Il existe alors une et une seule application, encore notée σ_1, définie sur S' telle que

$$\sigma_1(x) = x \quad \text{pour} \quad x \in V$$

et $$\sigma_1(gu_1 \ldots u_n) = \sigma_1(g)\,\sigma_1(u_1)\ldots\sigma_1(u_n)$$

pour tout $g \in L$ (avec $\underline{\text{source}}(g) = (i_1,\ldots,i_n)$) et toute suite $(u_1,\ldots,u_n)$ de termes de type respectif $i_1,\ldots,i_n$. σ_1 peut encore être prolongée aux formules de $\mathcal{F}$ de manière naturelle.

Considérons alors le système fonctionnel $\mathcal{F}''$, défini par

. l'alphabet $L'' = L \cup L_1$

. et l'ensemble X" d'axiomes propres :

- $X'' = X \cup \sigma_1(X) \cup Y$

où $Y = \left\{ sss_1^k t \equiv s_1^{k+1}t,\ x \not\equiv s_1^k t \supset sx \equiv s_1 x,\ vs^{k+1}t \equiv a \right\}$

On désigne également par $\mathcal{F}_1$ le système d'alphabet L_1 et d'axiomes propres $\sigma_1(X)$. Intuitivement, les termes sur L_1 représentent l'information

initiale tandis que les termes sur L représentent l'information modifiée. De manière précise, étant donnée une information I de $\mathcal{F}$, on peut définir successivement

. l'information I_1 de $\mathcal{F}_1$, transcrite de I par σ_1 et notée $\sigma_1(I)$,

. l'information I", extension de I_1 à $\mathcal{F}''$ et notée $\mathcal{E}''(I_1)$ et

. l'information I', restriction de I" à $\mathcal{F}$, notée $\mathcal{R}(I'')$.

I' est une information de $\mathcal{F}$, définie par

$$I' = \mathcal{R} \circ \mathcal{E}'' \circ \sigma_1(I) = m(I)$$

Nous dirons que la modification m est associée à la bijection σ_1 et au système d'axiomes Y. Nous noterons par la suite $m = adj(s^k t, a)$. Remarquons que adj est en fait un schéma de modifications, associant à chaque "place" $s^k t$ et à chaque "valeur" a une modification $adj(s^k t, a)$.

Plus généralement, voici une manière de définir une modification à n arguments. On se donne des bijections $\sigma_1,\ldots,\sigma_n$, respectivement de L dans $L_1,\ldots,L_n$ vérifiant les propriétés suivantes :

1) $\forall x,y \in L \quad \forall k \in [n] \quad (\sigma_k(x) = y \Rightarrow x = y)$

2) $\forall x,y \in L \quad \forall k,\ell \in [n] \quad (k \neq \ell \quad \text{et} \quad \sigma_k(x) = \sigma_\ell(y) \Rightarrow x = y)$

Soit $L'' = L \cup \bigcup_{k=1}^{n} L_k$. Compte tenu des hypothèses 1) et 2), on peut définir une application <u>profil</u> sur L", unique, telle que

$$\underline{\text{profil}}(\sigma_k(x)) = \underline{\text{profil}}(x) \quad \text{pour tout } x \text{ de } L \text{ et tout } k \in [n].$$

Pour chaque k de [n], σ_k définit naturellement une transcription, encore notée σ_k, de l'ensemble des formules sur L' dans l'ensemble des formules sur

L'_k. On se donne enfin un ensemble Y de formules sur L" et on note $\mathcal{F}$" le système fonctionnel d'alphabet L" et dont l'ensemble X" d'axiomes propres est défini par

$$X'' = X \cup Y \cup \bigcup_{k=1}^{n} \sigma_k(X).$$

On considère d'autre part, pour chaque k de [n], le système fonctionnel $\mathcal{F}_k$ transcrit de $\mathcal{F}$ par σ_k. Etant données des informations $I_1,\ldots,I_n$ de $\mathcal{F}$, on définit successivement

. les informations $\sigma_1(I_1),\ldots,\sigma_n(I_n)$ de $\mathcal{F}_1,\ldots,\mathcal{F}_n$,

. l'information I", extension à $\mathcal{F}$" des informations précédentes et notée $\mathcal{E}''(\sigma_1(I_1),\ldots,\sigma_n(I_n))$,

. et l'information I', restriction de I" à $\mathcal{F}$, notée $\mathcal{R}(I'')$.

On peut alors énoncer la définition suivante.

Définition 2 : Soit $\sigma_1,\ldots,\sigma_n$, Y comme ci-dessus. La modification m associée à $\sigma_1,\ldots,\sigma_n$, et à Y est définie par

$$m(I_1,\ldots I_n) = \mathcal{R} \circ \mathcal{E}''(\sigma_1(I_1),\ldots \sigma_n(I_n)).$$

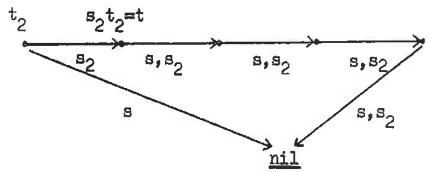
On dit encore que m est définie axiomatiquement par $\sigma_1,\ldots,\sigma_n$ et Y.

Exemples : Nous décrivons ici 3 modifications élémentaires de la structure de liste, représentant respectivement les opérations CAR, CDR et APPEND de LISP. On considère pour cela 2 bijections σ_1, σ_2 remplaçant s,t respectivement par s_1,t_1 et par s_2,t_2 et laissant fixes les autres symboles. On suppose $s_1 \neq s_2$ et $t_1 \neq t_2$.

a) La modification tête associe à toute liste I non vide la liste réduite

à l'élément de tête ; elle peut être définie par σ_1 et Y_1 où $Y_1 = \{t \equiv t_1 , st \equiv nil\}$.

b) La modification queue associe à chaque liste sa queue, c'est-à-dire la liste obtenue en enlevant l'élément de tête. Queue est simplement définie par σ_2 et $Y_2 = \{t \equiv s_2 t_2,\ x \neq t_2 \supset sx \equiv s_2 x\}$



La figure ci-dessus schématise les places d'une liste I à 5 éléments et de son extension I". Notons que la valeur de $s\,t_2$ n'est pas déterminée par les axiomes mais peut être prise égale à nil. Notons aussi qu'on ne peut poser $s\,t \equiv s_2 t$ sans entrer en contradiction avec l'axiome des listes

$$s\,x \equiv t \supset t \equiv nil$$

En effet, de $s\,t \equiv s_2 t$ et de $t \equiv s_2 t_2$, on déduit $st_2 \equiv t$ donc $t \equiv nil$.

c) La modification ajoute associe à deux listes I_1 et I_2 une liste I' définie de la manière suivante :

. si I_1 est la liste vide, $I' = I_2$

. si I_1 est la liste réduite à un élément, I' est obtenue en ajoutant I_1 devant I_2.

. sinon I' est l'information minimale, c'est-à-dire l'information réduite aux théorèmes de la structure de liste.

Ajoute peut être associée au triplet $(\sigma_1, \sigma_2, Y_3)$ où

$$Y_3 = \{ t_1 \equiv \underline{nil} \supset (t \equiv t_2 \wedge sx \equiv s_2 x) \}$$

$$\cup \{ (t_1 \not\equiv \underline{nil} \wedge s_1 t_1 \equiv \underline{nil}) \supset (t \equiv t_1 \wedge st \equiv t_2 \wedge (x \not\equiv t \supset sx \equiv s_2 x)) \}$$

Un autre exemple de modification : la conditionnelle

Soit $\mathcal{S}$ une structure d'information quelconque d'alphabet L. Pour toute formule p sans variable, introduisons une modification cond(p), à trois arguments, telle que, pour toutes I_1, I_2, I_3, I_1 consistante,

$$\text{cond}(p)(I_1, I_2, I_3) = \begin{cases} I_2 & \text{si } p \in I_1 \\ I_3 & \text{si } \neg p \in I_1 \\ T & \text{sinon} \end{cases}$$

où T désigne l'ensemble des théorèmes de $\mathcal{S}$.

Pour $j = 1,2,3$ soit σ_j une bijection de L sur un ensemble disjoint L_j telle que L_1, L_2, L_3 soient également disjoints deux à deux. Posons $f_j = \sigma_j(f)$ pour $f \in L$ et $j = 1,2,3$. Enfin, pour tout $i = (i_1 \ldots i_q)$ de $[m]^*$ soit $x_{i1}, \ldots, x_{iq}$ une suite de variables distinctes de type respectif $i_1, \ldots, i_q$. Soit $p_1 = \sigma_1(p)$. Cond(p) peut être défini par les bijections $\sigma_1, \sigma_2, \sigma_3$ et l'ensemble $Y = \{ p_1 \supset f x_{i1} \ldots x_{iq} \equiv f_2 x_{i1} \ldots x_{iq} \; ; \; f \in L, \underline{source}(f) = i \}$

$$\cup \{ \neg p_1 \supset f x_{i1} \ldots x_{iq} \equiv f_3 x_{i1} \ldots x_{iq} \; ; \; f \in L, \underline{source}(f) = i \}$$

On vérifie, en effet, par récurrence, que

$$p \in I_1 \implies u \equiv \sigma_2(u) \in I''$$

$$\neg p \in I_1 \implies u \equiv \sigma_3(u) \in I''$$

pour tout terme u de S'.

Exemple : Soit $\mathcal{S}$ la structure de liste et p la formule $t \equiv \underline{nil}$. Notons $\wedge$ l'information de $\mathcal{S}$ engendrée par p, c'est-à-dire la liste vide. Si I_1 est complète, ou bien I_1 contient p, donc $\wedge$, ou bien I_1 contient $\neg p$. On peut donc écrire, pour I_1 complète

$$\text{cond}(p)(I_1, I_2, I_3) = \underline{si} \; I_1 \supset \wedge \; \underline{alors} \; I_2 \; \underline{sinon} \; I_3$$

$\wedge$ étant complète, les conditions I_1 consistante et $I_1 \supset \wedge$ impliquent $(I_1)_0 = \wedge_0$ (où I_0 désigne l'ensemble des formules sans variable de I).

De manière générale, si I est une information complète, engendrée par un nombre fini d'axiomes sans variable $p_1, \ldots p_n$, soit $p = p_1 \wedge \ldots \wedge p_n$. Alors, si I_1 est complète,

$$p \in I_1 \iff I \subset I_1 \iff I_0 = (I_1)_0$$

et $$p \in I_1 \iff \neg p \notin I_1$$

Donc $$\text{cond}(p)(I_1, I_2, I_3) = \underline{si} \; (I_1)_0 = I_0 \; \underline{alors} \; I_2 \; \underline{sinon} \; I_3$$

Pour une information I_1 quelconque, on écrira

$$\underline{si} \; p \in I_1 \; \underline{alors} \; I_2 \; \underline{sinon} \; I_3$$

au lieu de $\text{cond}(p)(I_1, I_2, I_3)$

5.2.- Modifications engendrées par un ensemble de modifications élémentaires

Il est possible de développer une étude des modifications engendrées par $\mathcal{M}$ semblable à l'étude des accès d'une structure d'information.

Rappelons tout d'abord que l'ensemble $\mathcal{I}(\mathcal{F})$ des informations de $\mathcal{F}$, ordonné par la relation d'inclusion, est un ensemble complètement inductif (2.2.1). Nous pouvons donc appliquer aux modifications de $\mathcal{S}$ la théorie du point fixe

présentée en 4.2. Nous supposons pour cela que toutes les modifications élémentaires de $\mathcal{S}$ sont des applications continues sur $\mathcal{I}(\mathcal{F})$. On vérifie immédiatement que ceci est bien le cas lorsque ces modifications sont définies axiomatiquement.

Si $\mathcal{B}$ est un ensemble d'applications continues sur $\mathcal{I}(\mathcal{F})$ ($\mathcal{B} \subset \mathcal{A}(\mathcal{F})$) et si $\mathcal{T}$ est une fonctionnelle engendrée par $\mathcal{B}$ (4.5), nous savons que $\mathcal{T}$ est continue et donc admet un plus petit point fixe. Nous dirons que $\mathcal{B}$ est stable par l'opérateur plus petit point fixe si, pour toute fonctionnelle $\mathcal{T}$ engendrée par $\mathcal{B}$, les composantes du plus petit point fixe de $\mathcal{T}$ appartiennent à $\mathcal{B}$.

On dira que $\mathcal{B}$ est stable par composition si, pour tout $m \in \mathcal{B} \cap \mathcal{A}_n(\mathcal{F})$, tous $m_1,\ldots,m_n \in \mathcal{B} \cap \mathcal{A}_k(\mathcal{F})$, $m(m_1,\ldots,m_n) \in \mathcal{B}$, où $m(m_1,\ldots,m_n)(I_1,\ldots,I_k) = m(m_1(I_1,\ldots,I_k),\ldots,m_n(I_1,\ldots,I_k))$. Enfin, pour tout $n > 0$ et tout k de $[n]$, on note π^n_k la modification $\lambda I_1 \ldots I_n . I_k$. Soit $\mathcal{P}$ l'ensemble de ces projections.

Définition 3 : L'ensemble $\hat{\mathcal{M}}$ des modifications de $\mathcal{S}$ est le plus petit sous-ensemble de $\mathcal{A}(\mathcal{F})$ contenant $\mathcal{M} \cup \mathcal{P}$ et stable par composition et par l'opérateur plus petit point fixe.

Exemple : Considérons, dans la structure de liste, les modifications élémentaires tête, queue, ajoute et cond ($t \equiv$ nil). Alors, l'application concat, définie par l'équation ci-dessous, est une modification de la structure de liste

$$\text{concat}(I_1,I_2) = \underline{\text{si}}\ t \equiv \text{nil} \in I_1\ \underline{\text{alors}}\ I_2\ \underline{\text{sinon}}$$
$$\underline{\text{ajoute}}\ (\underline{\text{tête}}(I_1),\ \text{concat}\ (\underline{\text{queue}}(I_1),I_2))$$

La définition de $\hat{\mathcal{M}}$ conduit naturellement au type de récurrence suivant. Soit P une propriété sur $\hat{\mathcal{M}}$, telle que P(m) soit vérifiée, pour $m \in \mathcal{M} \cup \mathcal{P}$ et stable par composition et par l'opérateur plus petit point fixe. Alors P(m) est vérifiée pour tout m de $\hat{\mathcal{M}}$. On démontre en particulier, en raisonnant par récurrence, le résultat suivant.

Proposition 1.- m appartient à $\hat{\mathcal{M}}$ si et seulement si m est la première composante du plus petit point fixe d'une fonctionnelle engendrée par $\mathcal{M}$.

Nous pouvons alors adopter la définition suivante.

Définition 4 : Un programme sur une structure d'information $\mathcal{S}$ est un système à point fixe π sur $\mathcal{M}$

$$\pi : (m_1,\ldots,m_n) = \mathcal{T}(m_1,\ldots,m_n)$$

La modification m_π définie par π est la première composante du plus petit point fixe de $\mathcal{T}$.

Calculs sur une structure d'information

Considérons d'abord le cas où toutes les modifications élémentaires sont des applications de $\mathcal{I}(\mathcal{F})$ dans $\mathcal{I}(\mathcal{F})$ (cas monadique). On appelle alors calcul sur $\mathcal{S}$ toute suite finie de modifications élémentaires. Un calcul est donc un élément de $\mathcal{M}^*$.

Exemple : On peut considérer, sur les listes, le calcul

(queue, queue, tête)

A ce calcul est associée la modification $m = \underline{\text{tête}} \circ \underline{\text{queue}} \circ \underline{\text{queue}}$ qui à toute liste I associe la liste formée du 3ème élément de I, si cet élément existe, et la liste vide sinon. De manière générale, si $c = m_1 \ldots m_n$ est un calcul sur $\mathcal{S}$, on associe à c la modification $m = m_n \circ \ldots \circ m_1$. Si $n = 0$, m est l'identité de $\mathcal{I}(\mathcal{F})$ dans lui-même (c'est-à-dire l'application π^1_1).

Pour généraliser la définition des calculs à des modifications arbitraires, remarquons qu'on peut également associer à un calcul $c = m_1 \dots m_n$ le schéma fonctionnel $c' = m_n \dots m_1\ X$ sur $\mathcal{M} \cup \{X\}$ où X est une variable. Nous confondrons par la suite c et c'.

Dans le cas général, soit $\mathcal{V} = \{X_1, X_2, \dots, X_n, \dots\}$ un ensemble infini de variables. Chaque variable est un symbole 0-aire. D'autre part, chaque modification de $\mathcal{M}_n$ peut être considérée comme un symbole d'arité n. Nous pouvons alors poser la définition suivante.

Définition 5 : Un calcul sur $\mathcal{S}$ est un schéma fonctionnel sur $\mathcal{M} \cup \mathcal{V}$.

Exemple : ajoute.tête.X_1.queue.X_1 est le calcul associé à la modification $m = \lambda I.$ ajoute (tête(I), queue(I)).

Soit c un schéma fonctionnel sur $\mathcal{M} \cup \{X_1, \dots, X_n\}$. La modification m associée à c est définie récursivement par les règles :

1) Pour $k = 1, \dots, n$ Π^n_k est associée à X_k

2) Si $m \in \mathcal{M}_k$ et si, pour $j = 1, \dots, k$, m_j est associée à c_j, alors $m(m_1, \dots, m_k)$ est associée au calcul $m c_1 \dots c_k$.

Une telle modification est obtenue par composition des modifications élémentaires et des projections. Nous n'avons pas étudié dans le présent travail le calcul généralisé qui devrait être associé à une modification définie récursivement et donc également à un programme. Il s'agit de plonger l'ensemble des calculs dans un ensemble complètement inductif. Le calcul généralisé associé à un programme $m = \mathcal{C}(m)$ quelconque serait alors le plus petit point fixe d'une fonctionnelle $\widetilde{\mathcal{C}}$ déduite de $\mathcal{C}$. Des études ont été faites dans ce sens dans (SCOTT), (NIVAT) et (FINANCE).

Avant de donner des exemples de programmes, nous introduisons la notion de problème : un programme (ou un algorithme) calcule en effet la solution (ou une solution), si elle existe, d'un problème donné.

5.3.- Formalisation de la notion de problème

5.3.1.- Introduction

En informatique, un problème s'exprime généralement sous forme d'un ensemble de relations entre des objets composés. On peut distinguer parmi les objets

. des objets constants, dont la "valeur" est déterminée dans l'énoncé du problème.

. des objets dont la valeur est laissée libre par l'énoncé du problème mais qui doivent éventuellement vérifier certaines conditions ; ces objets constituent la donnée du problème.

. des objets dont la valeur dépend de la valeur des données ; certains de ces objets servent simplement à effectuer des calculs intermédiaires. Les autres sont les résultats du problème. Il est clair que le choix des données et des résultats fait partie intégrante de la définition du problème.

Donnons un premier exemple dans lequel l'énoncé est un ensemble d'égalités. Il s'agit de calculer la paie mensuelle d'un salarié relevant du régime général de la Sécurité Sociale connaissant le nombre d'heures de travail effectuées et le salaire horaire. Le problème peut être défini par le système d'équations

net à payer = brut - retenue

retenue = si brut > plafond alors (ss1 + ss2) sinon brut x 0,065

plafond = 2320

ss1 = brut x 0,01

ss2 = plafond x 0,055

brut = nombre d'heures x salaire horaire

Dans cet exemple, les données sont nombre d'heures et salaire horaire. Rien n'empêche de modifier le problème en enlevant par exemple l'équation

plafond = 2320

et en considérant plafond comme une donnée, changeant chaque année. Le résultat peut être simplement net à payer ou, si l'on désire plus de renseignements, l'ensemble $\{$ net à payer, retenue $\}$.

Exemple 2 : Définition du p.g.c.d. de deux entiers

Etant donnés deux nombres entiers a et b le pgcd de a,b peut être défini par la relation :

pgcd divise a $\wedge$ pgcd divise b $\wedge$ [x divise a $\wedge$ x divise b] $\supset$ x divise pgcd

Dans ce problème, les données sont a,b , le résultat est pgcd et l'énoncé est la formule précédente.

L'intérêt de ce type de problème (et la difficulté) est qu'on ne peut en déduire de manière immédiate un algorithme de calcul du pgcd. Cette recherche d'algorithmes calculant la solution d'un problème donné est quelquefois appelée synthèse de programmes [MANNA-WALDINGER]. Il est connu que cette synthèse de programme ne peut être, dans le cas général, entièrement automatisée. Un certain nombre de groupes de recherche s'attachent cependant à définir une méthodologie dans l'écriture des programmes et parallèlement à développer des techniques de preuves assurant la correction du programme trouvé.

Donnons un programme calculant le pgcd de deux entiers a,b. Il s'agit simplement de traduire en terme de modification l'algorithme d'Euclide.

pgcd(a,b) = si b = 0 alors a sinon pgcd(b, r(a,b))

r(a,b) désignant le reste de la division de a par b.

Introduisons pour cela les affectations. Il s'agit en réalité d'un schéma de modifications.

Si $a_1,\ldots,a_n$ sont des identificateurs c'est-à-dire des symboles de constantes et $u_1,\ldots,u_n$ des termes de même type, la modification $aff(a_1,\ldots,a_n ; u_1,\ldots,u_n)$ associe à toute information I, l'information I' contenant les théorèmes $a_i \equiv u_i$ pour $1 \leqslant i \leqslant n$. De manière précise $aff(a_1,\ldots,a_n ; u_1,\ldots,u_n)$ peut être définie par le couple (σ, Y) où σ est une bijection associant à chaque a_i un symbole a_i' et laissant fixes les autres symboles et où Y est l'ensemble des formules $a_i \equiv \sigma(u_i)$ pour $i = 1,\ldots,n$.

Compte tenu de ces définitions l'algorithme d'Euclide peut être traduit par le programme

m(I) = si b $\equiv$ 0 $\in$ I alors aff(pgcd ; a) (I) sinon

m$\circ$aff(a,b ; b,r(a,b))(I)

5.3.2.- Définitions

Définition 6 : Un problème Π sur une structure $\mathcal{S}$ est la donnée d'un ensemble $L(\Pi)$ d'accès nouveaux et d'un ensemble $X(\Pi)$ d'axiomes, formules sur l'alphabet $L' \cup L(\Pi) . L(\Pi)$ et $X(\Pi)$ sont respectivement l'ensemble des inconnues et l'énoncé du problème.

Exemples : Nous avons étudié au chapitre 4 les problèmes récursifs. Dans ce type de problème, $L(\Pi) = \{f_1,\ldots,f_n\}$ et $X(\Pi)$ est un système d'équations récursives.

L'énoncé du problème peut être plus complexe (définition du pgcd).

On appelle <u>donnée</u> de $\mathcal{P}$ toute information de $\mathcal{S}$. $L(\mathcal{P})$ et $X(\mathcal{P})$ déterminent une extension $\mathcal{E}$ de $\mathcal{S}$ à une structure $\mathcal{S}'$ d'alphabet $L \cup L(\mathcal{P})$ et d'ensemble d'axiomes propres $X \cup X(\mathcal{P})$. On appelle alors <u>résultat</u> de $\mathcal{P}$ associé à une donnée I de $\mathcal{S}$ toute information I'_0 complète contenant $\mathcal{E}(I_0)$ (où I_0 est la restriction de I à $\mathcal{F}_0$ (2.1)).

On dira qu'un problème $\mathcal{P}$ est <u>déterministe</u> si, pour toute information I complète, $\mathcal{E}(I)$ est également complète, donc si $\mathcal{P}$ associe à chaque donnée complète un et un seul résultat.

conclusion

CONCLUSION

Cette étude peut être poursuivie dans un certain nombre de directions.

a) Jusqu'à présent, nous n'avons cherché à formaliser que des domaines très simples tels que les entiers ou les listes. On doit pouvoir traiter des domaines aussi complexes que les banques de données ou la gestion des travaux d'un système. Un premier travail (FINANCE) utilise les structures d'information pour formaliser la sémantique des langages de programmation. (PAIR) étudie également les structures d'information les plus courantes, mais sans utiliser les systèmes fonctionnels. Dans les preuves de programmes, la formalisation présentée ici prend bien en compte l'axiomatique du domaine de base.

b) Pour étudier des structures d'information complexes, il semble indispensable de définir sur celles-ci des opérations algébriques. C'est ainsi qu'on doit pouvoir ramener l'étude des listes de listes ou des arborescences à celle des listes.

c) Au chapitre 3, nous construisons plusieurs interprétations de la structure des modes Algol 68. Il nous semble que ces constructions sont assez généralisables. On pourra ainsi démontrer la consistance de certaines structures non triviales.

d) Les notions de calcul, de programme et de problème n'ont été qu'abordées ici. Le premier point est de définir les calculs associés à un programme récursif

ou simplement itératif. Une première étude a été faite par (FINANCE) dans le cas de modifications à une variable. Des recherches ont aussi été effectuées, indépendamment des structures d'information par (SCOTT) et (NIVAT). Les idées qui y sont développées peuvent certainement servir de point de départ.

e) Pour les problèmes, nous nous sommes efforcés d'écrire leur énoncé (par exemple : définition du p.g.c.d) indépendamment de tout algorithme. Cela est-il toujours possible ? Sinon, comment faut-il modifier les définitions? Par exemple, nous n'avons obtenu jusqu'à présent d'énoncé très satisfaisant pour l'interclassement de deux listes triées.

index

INDEX DES NOTATIONS

I) Notations attachées à une structure d'information

Une structure $\mathcal{S}$ est définie par un système formel $\mathcal{F}$ et un ensemble de modifications élémentaires $\mathcal{M}$.

$\mathcal{F}$ est complètement déterminé par un alphabet L de symboles fonctionnels et un ensemble X d'axiomes propres. Précisément $\mathcal{F}$ est un quadruplet $(\mathcal{L}, F, \mathcal{X}, R)$ où

. $\mathcal{L} = L \cup V \cup \{ \equiv, \neg, \supset, (,) \}$

. $V = \bigcup_{j=1}^{m} V_j$ est l'ensemble des variables

. $m \geqslant 1$ est le nombre de types distincts de $\mathcal{F}$. On pose $[m] = \{1, \ldots, m\}$

. S'_j (resp S_j) désigne l'ensemble des termes (resp des termes sans variable) de type j $(j \in [m])$.

. $A = \bigcup_{j=1}^{m} S_j \equiv S_j$ est l'ensemble des formules atomiques

. l'ensemble F des formules est solution de l'équation

$$F = A \cup \neg F \cup (F \supset F)$$

. $\mathcal{X} = Prop(A) \cup Eg \cup X$ est l'ensemble des axiomes de $\mathcal{F}$.

. R est l'ensemble des règles du système.

On désigne par T l'ensemble des théorèmes de $\mathcal{F}$, par $\mathcal{I}(\mathcal{F})$ (ou $\mathcal{I}$) l'ensemble des informations de $\mathcal{F}$.

On note $\mathcal{A}_n(\mathcal{F})$ l'ensemble des applications de $(\mathcal{I}(\mathcal{F}))^n$ dans $\mathcal{I}(\mathcal{F})$ et $\mathcal{A}(\mathcal{F}) = \bigcup_{n \geqslant 0} \mathcal{A}_n(\mathcal{F})$.

L'ensemble $\mathcal{M}$ des modifications élémentaires est un sous-ensemble de $\mathcal{A}(\mathcal{F})$. On pose $\mathcal{M}_n = \mathcal{M} \cap \mathcal{A}_n(\mathcal{F})$.

On désigne généralement par I, I_1, I_2 ... les informations de $\mathcal{F}$. On note F_o l'ensemble des formules sans variable de $\mathcal{F}$ et $I_o = I \cap F_o$.

Une interprétation est une suite $R = (E_1,\ldots,E_m,r) = (E,r)$ où les ensembles E_j sont non vides et r est une application de L dans l'ensemble des applications définies sur les E_j. On note $\hat{r}$(resp $\tilde{r}$) le prolongement de r à S' (resp à F).

Enfin aux chapitres 1, 2, 4 on pose $L' = L \cup V$ et, si x est une suite $(x_1,\ldots,x_n)$ de variables distinctes, $L(x) = L \cup \{x_1,\ldots,x_n\}$. De même, $S_j(x)$ désigne l'ensemble des termes de type j sur L(x).

On pose

$$S' = \bigcup_{j=1}^{m} S'_j \quad , \quad S = \bigcup_{j=1}^{m} S_j \quad , \quad S(x) = \bigcup_{j=1}^{m} S_j(x).$$

Exceptionnellement, au chapitre 3, L'(resp S') désigne l'alphabet (resp l'ensemble des termes sans variable) d'une structure $\mathcal{S}'$.

II) Notations syntaxiques

Un certain nombre de lettres sont utilisées au cours de ce travail, représentant chacune un type d'objets donné. Rappelons que m est un entier fixé, désignant le nombre de types du système fonctionnel.

j désigne un entier de [m]

i désigne un n-uple $(i_1,\ldots,i_n)$ de $[m]^*$

f,g désignent des symboles fonctionnels.

u,v,w,h désignent des termes de S' et plus généralement des n-uples de termes.

x,y,z,t désignent des variables ou des n-uples de variables. De manière générale, si $x = (x_1,\ldots,x_n) \in V^*$, on suppose les x_k tous distincts.

p,q,r désignent des formules arbitraires.

Au chapitre 4, τ, σ, ρ désignent des termes sur l'alphabet étendu $L' \cup \{f\}$.

III) Notations et abréviations fréquemment utilisées

Pour alléger l'écriture des formules, on s'est efforcé d'étendre chaque notation, définie pour un type d'objets, aux suites d'objets de ce type. Nous donnons ici une liste des principales notations introduites et des conventions correspondantes.

Notations introduites au chapitre 1

. Si $A_1,\ldots,A_m$ sont des ensembles et si $i = (i_1,\ldots,i_n) \in [m]^*$, on pose $A_i = A_{i_1} \times \ldots \times A_{i_n}$.

. Soit $i = (i_1,\ldots,i_n) \in [m]^*$, $x = (x_1,\ldots,x_n) \in V_i$, $u = (u_1,\ldots,u_n) \in S'_i$, $h \in S'$, f un symbole fonctionnel tel que source(f) = i.

On pose

type(u) = i =(type (u_1),...,type(u_n)) (p. 1.8)

$fu = fu_1 \ldots u_n$

$h_x[u]$ (ou h[u]) désigne le terme obtenu en remplaçant dans h chaque x_k par u_k pour $k = 1,\ldots,n$ (p. 1.8).

$\hat{r}(u) = (\hat{r}(u_1),\ldots,\hat{r}(u_n))$.

Notations introduite au chapitre 4

On considère un symbole f n'appartenant pas à L. On désigne par $\mathcal{S}(f)$ l'extension de $\mathcal{S}$ d'alphabet $L(f) = L \cup \{f\}$ et d'ensemble d'axiomes propres

X (p. 4.8). Si I est une information de $\mathcal{S}$, I(f) est l'extension de I dans $\mathcal{S}(f)$. On note ρ, π, τ les termes sur L'(f) et, plus généralement, les suites de tels termes.

Si R = (E,r) est une interprétation de $\mathcal{S}$, on note $\hat{\tau}(\tau)$ ou τ la fonctionnelle associée à τ dans R. Si $\tau = (\tau_1,\ldots,\tau_n)$,

$$\tau = \hat{\tau}(\tau) = (\hat{\tau}(\tau_1),\ldots,\hat{\tau}(\tau_n)) = (\tau_1,\ldots,\tau_n) \quad \text{(p. 4.9)}$$

Si h est un terme de même profil que f, on note $\tau(h)$ le terme obtenu en "remplaçant f par h" (p. 4.10). Si $\tau = (\tau_1,\ldots,\tau_n)$, $\tau(h) = (\tau_1(h),\ldots,\tau_n(h))$. On obtient

$$\hat{\tau}(\tau(h)) = \tau(\hat{\tau}(h)) \qquad \text{(p. 4.11)}$$

En particulier, si σ contient des occurrences de f, on pose $\tau \circ \sigma = \tau(\sigma)$

Alors,

$$\tau \circ \sigma\,(h) = \tau(\sigma(h)) \qquad \text{(p. 4.12)}$$

On introduit un symbole ω et l'on définit, pour τ fixé, une suite (f_n) de termes :

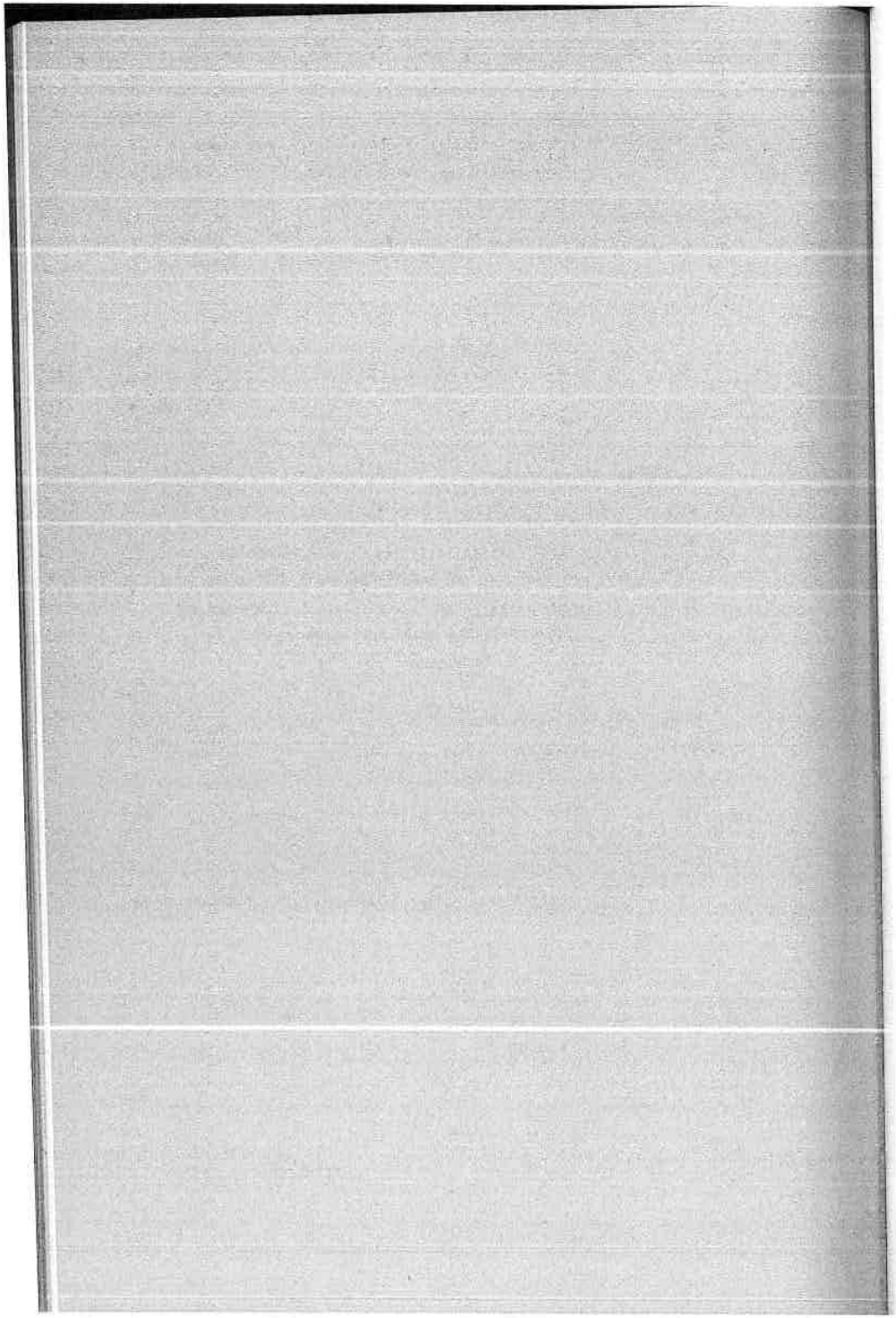
$$\begin{cases} f_0 = \omega \\ f_n = \tau(f_{n-1}) \qquad n \geqslant 1. \end{cases}$$

Il suit de la relation précédente que $f_n = (\tau \circ \ldots \circ \tau)(\omega) = \tau^n(\omega)$ (p. 4.17)

IV) Autres notations

INDEX TERMINOLOGIQUE

références



BIBLIOGRAPHIE

ALGOL 68, Définition du Langage Algorithmique ALGOL 68 - Présentation et traduction française du "Report on the Algorithmic Language Algol 68", Groupe Algol de l'AFCET, HERMANN (1972).

DE BAKKER J.W. et DE ROOVER W.P., A calculus for recursive program schemas, dans Automata, Languages and Programming (M. Nivat, ed.), p. 167-196, North-Holland, Amsterdam, 1973.

CADIOU J.M., Recursive definitions of partial functions and their computations, MEMO AIM 163, Standford University, 1972.

FINANCE J.P, Contribution à la formalisation de la sémantique d'un langage de programmation. Application à Algol 68, thèse de 3ème cycle, Nancy, 1974.

GARLAND D.J et LUCKHAM D.C, Program schemes, Recursion schemes and Formal languages, JCSS, 7 (1973), p. 119-160.

HOPCROFT J.E. et ULLMAN J.D., Formal languages and their relations to automata, Addison Wesley, London, 1969.

KREISEL G. et KRIVINE J.L., Eléments de logique mathématique. Théorie des modèles, Dunod, Paris, 1967.

MANNA Z., NESS S. et VUILLEMIN J., Inductive methods for proving properties of programs, C.ACM, 18 (1973), p. 491-502

MANNA Z. et WALDINGER J.R., Towards automatic program syntheses, dans Symposium on Semantics of Algorithmic Languages (E. Engeler, ed.), p. 270-310, Lecture Notes in Mathematics, Vol. 188, Springer, Berlin, 1971.

NIVAT M., Langages Algébriques sur le magma libre et Sémantique des schémas de programme, dans Automata, Languages and Programming (M. Nivat, ed.), p. 293-357, North-Holland, Amsterdam, 1973.

PAIR C., Structures d'information, Cours de l'école d'été de l'AFCET, Alès, 1971.

PAIR C. et FINANCE J.P., Formalisation des notions d'information, de structure d'information et de problème, Université de Nancy II, 1973.

PAIR C. et QUERE A., Définition et étude des bilangages réguliers, Information and Control 13 (1968), p. 565-593.

REMY J.L. et FINANCE J.P., Structures d'information et sémantique d'un langage de programmation, Université de Nancy, 1973.

SHOENFIELD J.R., Mathematical Logic, Addison Wesley, London, 1967.

SCOTT D., Outline of a mathematical theory of computation, Proc. Ann. Princeton Conf. on Information Sciences and Systems, Princeton U., 1970, p. 169-176.

SCOTT D., The lattice of flow diagrams, dans Symposium on Semantics of Algorithmic Languages (E. Engeler, ed.), p. 311-366, Lecture Notes in Mathematics, Vol. 188, Springer, Berlin, 1971.

VUILLEMIN J., Proof techniques for recursive programs, Notes de travail, IRIA, 1973.

NOM DE L'ETUDIANT : REMY Jean Luc

NATURE DE LA THESE : Doctorat de Spécialité en Mathématiques Appliquées (Informatique)

VU, APPROUVE

& PERMIS D'IMPRIMER

NANCY, le 14 Juin 1974

LE PRESIDENT DE L'UNIVERSITE DE NANCY I

J.R. HELLIET