Publié dans Travaux de logique 13, 147-158, 1999,
source qui doit être utilisée pour toute référence à ce travail

LES TYPES EN MATHÉMATIQUE ET EN LINGUISTIQUE

Jim LAMBEK

1. Les types en mathématique

Des types (fréquemment appelés catégories) ont toujours été présents dans les langues naturelles. En anglais, par exemple, on peut distinguer le type des personnes, comme dans *somebody* ou *who*, et le type des choses, comme dans *something* ou *what*.

Les types ont été introduits en mathématique par Bertrand Russell quand il découvrit son fameux paradoxe. Selon le schéma de compréhension de Frege, on pouvait former l'ensemble:

$$R = \{x \mid x \in x\}; \text{ mais alors } R \in R \Leftrightarrow R \notin R$$

qui est évidemment une contradiction. Pour la circonvenir, Russell insistait sur le fait que l'on doit assigner un type à chaque variable, comme à chaque autre terme. Si x est de type A, alors $x \in y$ est une expression bien formée *si et seulement si* y est d'un type approprié plus élevé que A, c'est-à-dire du type des sous-ensembles de A ou des fonctions $A \to 2 = \{\text{vrai, faux}\}$, du moins en mathématique classique. D'après Cantor, ce type – souvent désigné par 2^A – est tel que si $2^A \neq A$, alors $x \in x$ est une expression mal formée.

Les fondements type-théorétiques initiés par Russell et Whitehead dans leur célèbre «tour de force» des *Principia Mathematica* ont cependant été jugés trop compliqués par la plupart des mathématiciens et logiciens. Les mathématiciens adoptèrent de préférence la théorie de Gödel-Bernays, qui circonvient le paradoxe de Russell en distinguant les ensembles des classes: R est une classe et on ne peut pas la substituer à la variable x. Les logiciens, de leur côté, préférèrent la théorie du premier ordre de Zermelo-Fraenkel, selon laquelle le schéma de compréhension prend la forme suivante:

$$\{x \mid x \in a \wedge \varphi(x)\}$$

pour chaque terme donné *a*. On peut former, en particulier, l'ensemble russellien suivant:

$$R_a = \{x \mid x \in a \wedge x \notin x\}$$

et conclure alors que

$$R_a \in R_a \Leftrightarrow (R_a \in a \wedge R_a \notin R_a)$$

c'est-à-dire

$$R_a \notin R_a \wedge R_a \notin a$$

qui n'est pas une contradiction.

Bien que la théorie des types ait déjà été présentée d'une manière beaucoup plus simple et élégante par Church (1940) et Henkin (1950), elle n'est pas encore généralement acceptée. Plus récemment, Lambek et Scott (1986) ont essayé de restaurer cette théorie comme un fondement plus naturel pour les mathématiques.

Voici une brève esquisse de notre théorie des types. Considérons tout d'abord deux types de base: N pour les nombres naturels et Ω pour les valeurs de vérité. Nous avons également, pour chaque types donnés A et B, les types composés $A \times B$ et Ω^A. On dispose enfin de variables de chaque type ainsi que des termes suivants (où l'on écrit a : A pour indiquer que *a* est de type A):

$0, Sn : N$ *si* $n : N$;
$a = a', a \in \alpha : \Omega$ *si* $a, a' : A$ *et* $\alpha : \Omega^A$;
$\langle a, b \rangle : A \times B$ *si* $a : A$ *et* $b : B$;
$\{x : A \mid \varphi(x)\} : \Omega^A$ *si* $\varphi(x) : \Omega$

Quant aux connecteurs et aux quantificateurs usuels, il est possible d'en définir certains d'une manière directe:

$T \equiv 0 = 1$;
$p \wedge q \mathrel{.\equiv.} \langle p, q \rangle = \langle T, T \rangle$ où $p, q : \Omega$
$p \Rightarrow q \mathrel{.\equiv.} (p \wedge q) = p$
$\forall_{x:a} \varphi(x) \mathrel{.\equiv.} \{x : a \mid \varphi(x)\} = \{x : a \mid x = x\}$ où $\varphi(x) : \Omega$

Sur cette base, on peut alors en définir d'autres. D'une manière classique, on posera par exemple pour la disjonction:

$$p \vee q \mathrel{.\equiv.} \neg(\neg p \wedge \neg q)$$

A la manière des intuitionnistes, on posera par contre:

$$p \vee q \mathrel{.\equiv.} \forall_{x:\Omega}(((p \Rightarrow x) \wedge (q \Rightarrow x)) \Rightarrow x)$$

Notre théorie comporte bien entendu des axiomes et des règles d'inférence. Nous les omettons dans ce bref exposé et renvoyons pour plus de détails à Lambek et Scott (1986).

Liés à cette théorie des types, on trouve le lambda-calcul de Church et la logique combinatoire équivalente de Schönfinkel et Curry. Pour le lambda-calcul, les types composés B^A pour des types donnés A et B sont décisifs, et pas uniquement lorsque $B = \Omega$. Aussi, introduisons-nous les termes suivants:

$$\varepsilon(f, a) : B \quad \textit{si} \quad a : A \quad \textit{et} \quad f : B^A$$
$$\lambda_{x:A}\, \varphi(x) : B^A \quad \textit{si} \quad \varphi(x) : B$$

et précisons que:

$$\varepsilon(f, x) = \varphi(x) \quad \textit{ssi} \quad f = \lambda_{x:A}\, \varphi(x).$$

Dans le cas où $B = \Omega$, on écrit habituellement:

$$a \in f \equiv \varepsilon(f, a), \quad \{x : A \mid \varphi(x)\} \equiv \lambda_{x:A}\, \varphi(x).$$

On écrit souvent $A \wedge B$ pour $A \times B$ et $A \Rightarrow B$ pour B^A et on identifie les types avec des formes du calcul propositionnel positif intuitionniste, sous la devise «les formules comme types». De la même manière, les preuves formelles sont alors conçues comme les entités des types correspondants.

Selon l'isomorphisme de Curry-Howard (Howard 1980), la théorie des preuves du calcul propositionnel positif intuitionniste équivaut au lambda-calcul typé. Même le type de base Ω peut correspondre à une contradiction (Lambek 1997), mais dans ce contexte on ne doit pas tenir compte du type de base N.

2. Les types en linguistique

Dans le lambda-calcul typé de Church (1940), il y a deux types de base:

e = entité t = valeur de vérité.

Dans son analyse syntaxique, Ajdukiewicz (1935), faisant référence aux travaux de Lesniewski, utilisait aussi deux types de base qu'il appelait «catégories sémantiques»:

S = Phrase (Sentence) N = Nom (Name)

Prenons l'exemple suivant:

John	*snores*
N	$N \Rightarrow S$

L'application du *modus ponens* montre qu'il s'agit d'une phrase. Cependant, Ajdukiewicz avait en vue une analogie arithmétique et non pas logique. Il écrivait donc S *sur* N plutôt que $N \Rightarrow S$.

Bar-Hillel (1953) faisait, pour sa part, une distinction supplémentaire entre foncteurs de droite (par exemple *snores*) et foncteurs de gauche (par exemple *poor*). En poursuivant l'analogie logique, on pourrait écrire:

poor	*John*	*snores*
$N \Leftarrow N$	N	$N \Rightarrow S$

mais Bar-Hillel visait également une analogie arithmétique.

En suivant notre notation (1958) une telle analyse devient:

poor	*John*	*snores*
N / N	N	$N \backslash S$

Nous introduisons ainsi une distinction entre deux sortes de divisions:

/ dessus \ dessous

Voici un exemple plus élaboré:

poor	*John*	*sees*	*Jane*	*today*
N / N	N	$N \backslash (S / N)$	N	$S \backslash S$

Dans notre calcul syntaxique, les types sont considérés comme les éléments d'un semi-groupe résiduel, c'est-à-dire un semi-groupe partiellement ordonné satisfaisant les conditions suivantes:

$$ab \rightarrow c \ \textit{ssi} \ a \rightarrow c / b \ \textit{ssi} \ b \rightarrow a \backslash c$$

en plus de la loi d'associativité

$$(ab)c = a(bc).$$

Ici $\rightarrow$ marque une relation transitive et réflexive que les logiciens peuvent lire comme la déductibilité mais que les linguistes voient comme la dérivabilité.

Avec le calcul syntaxique, on peut alors prouver nombre d'inégalités, par exemple:

$(c/b)\ b \rightarrow c$	application
$b \rightarrow (c/b) \setminus c$	élévation du type
$(c/b)\ (b/a) \rightarrow c/a$	composition
$c/b \rightarrow (c/a) / (b/a)$	règle de Geach

cette dernière ayant été redécouverte par Geach (1971).

Jusqu'ici, notre attention s'est portée essentiellement sur les aspects algébriques du calcul syntaxique. Il est cependant également possible de considérer celui-ci comme un système déductif. Cet aspect prédominait lorsque je me suis rendu compte que la méthode d'élimination des coupures de Gentzen pouvait y être appliquée afin de résoudre le problème de décision. En fait, le calcul syntaxique est essentiellement identique au calcul propositionnel positif intuitionniste, excepté le fait qu'il ne convient pas pour les trois règles structurales de Gentzen.

Il s'agit ici des trois règles suivantes:

$ab \rightarrow ba$	permutation,
$a \rightarrow aa$	contraction,
$ab \rightarrow b$	affaiblissement.

Remarquons que certaines d'entre elles sont également absentes d'autres logiques – dites substructurales: l'affaiblissement manque en logique relevante, la contraction en logique BCK. Enfin ces deux règles manquent en logique linéaire (dans un contexte où la juxtaposition indique la multiplication, ajoutons encore que la permutation est une conséquence de la contraction et de l'affaiblissement).

De manière tout à fait indépendante, Haskell Curry (1961) développa sa propre méthode. Celle-ci, qui apparaît désormais comme un calcul des types sémantiques, est essentiellement identique à un calcul positif intuitionniste. Dès lors que la théorie de la preuve peut être identifiée au lambda-calcul (ou plutôt, comme l'a montré Curry, à une logique combinatoire équivalente), elle apparaît comme un instrument adéquat pour exprimer le sens des mots en anglais.

Reprenons l'exemple *John snores*. Celui-ci est de type $e\ (e \Rightarrow t) \rightarrow t$, et l'on peut alors écrire:

$$snores = \lambda_x\ (x\ snores)$$

Cette idée, plus développée, est connue sous le nom de grammaire de Montague (1974).

Le calcul syntaxique a été récemment reconnu comme le fragment multiplicatif d'une version intuitionniste de logique linéaire non commutative, autrement dit, la logique linéaire moins la règle de permutation. Nous parlerons ici de préférence de logique bilinéaire.

Claudia Casadio a eu l'idée – nouvelle et intéressante – d'appliquer la logique bilinéaire classique à la linguistique. Cette logique développée par Grishin (1983), Abrusci (1991), et moi-même (1993) peut être obtenue plus facilement par le calcul syntaxique en y introduisant un objet dualisant 0 tel que:

$$(0 / a) \setminus 0 = a = 0 / (a \setminus 0)$$

En posant:

$$0 / a = a^{\ell}, \quad a \setminus 0 = a^{r}$$

on peut aisément vérifier que:

$$(b^{\ell}\ a^{\ell})^{r} = (b^{r} a^{r})^{\ell}$$

pour lequel il est plus simple d'écrire a+b. Cette opération est associative et satisfait les formules suivantes:

$$a + 0 = a = 0 + a$$

$$(a + b)\ c \rightarrow a + bc, \quad c\ (a + b) \rightarrow ca + b.$$

Ces dernières règles sont appelées, dans différents ouvrages, règles d'associativité mixte ou de distributivité faible.

Lorsque j'ai assisté à la conférence où Mme Casadio présentait ses travaux, je fus immédiatement convaincu qu'elle était sur la bonne voie. Cependant, l'interprétation linguistique de $a + b$ restait plutôt mystérieuse (Lambek 1993) et il m'apparût alors qu'il fallait identifier:

$$a + b = ab$$

et travailler plutôt à une logique bilinéaire «compacte» dont une version commutative en termes de catégories existait déjà dans

la littérature (Barr 1979). Cette logique peut être présentée plus simplement comme suit (le signe $\rightarrow$ désignant une relation d'ordre partiel qui préserve la multiplication):

$(ab)\,c = a\,(bc), \quad a1 = a = 1a,$

$a^{\ell}\,a \rightarrow 1 \rightarrow aa^{\ell}, \quad aa^{r} \rightarrow 1 \rightarrow a^{r}a$

On comprend alors comment le type de:

poor John saw Jane today

peut être calculé dans (a) le calcul syntaxique, (b) la logique bilinéaire et (c) la logique bilinéaire compacte.

(a)
$$\frac{\dfrac{\dfrac{\underline{N/N\;N} \qquad N\setminus(S/N)}{N} \qquad N}{S/N} \qquad S\setminus S}{S}$$

(b)
$$\frac{\dfrac{\dfrac{\underline{(N+N^{\ell})\;N} \qquad (N^{r}+S+N^{\ell})}{N} \qquad N}{S+N^{r}} \qquad (S^{r}+S)}{S}$$

(c)
$$\begin{array}{llll} (N\;N^{\ell})N & (N^{r}\;S\;N^{\ell})N & (S^{r}\;S) \\ \quad \mathbf{I—I} & \quad \mathbf{I—I} & \\ \mathbf{I----------I} & \mathbf{I----------I} & \end{array}$$

La partie soulignée en (c) représente la contraction, par exemple $N^{\ell}\,N \rightarrow 1$, $N\,N^{r} \rightarrow 1$. C'est tout ce qui reste des réseaux de preuves de la logique linéaire. Il a été démontré dans Lambek (à paraître) que les expansions comme $1 \rightarrow N\,N^{\ell}$ et $1 \rightarrow N^{r}\,N$ ne sont pas nécessaires tant que l'on est seulement intéressé par la validation de la caractéristique de phrase.

Bien entendu, les types de base N et S ne suffisent pas pour évaluer la grammaire de l'anglais d'une manière réaliste. Pour ce faire, nous avons besoin d'un ensemble plus élaboré de types de base qui peut être ordonné. A partir d'un petit nombre d'exemples, nous adopterons les types de base suivants:

π_1, π_2, π_3	les trois personnes du singulier (il est à noter qu'en anglais les trois personnes du pluriel se comportent comme π_2)
s_1, s_2, s	les phrases aux temps présent, passé, ou dans lesquelles le temps importe peu
q	les questions oui-non
q'	toutes les questions (incluant les questions oui-non et celles commençant par *wh*)
p_1, p_2	les participes présents et passés
o	les objets.

Ces types sont ordonnés de la manière suivante:

$$s_i \to s, \qquad q \to q'.$$

Voici quelques phrases anglaises simples:

I	*go*		*you went*			*he goes*	
π_1	$(\pi_1{}^r\, s_1)$		π_2	$(\pi_2{}^r\, s_2)$		π_3	$(\pi_3{}^r\, s_1)$
I	*am*	*going*		*she*	*has*		*gone*
π_1	$(\pi_1{}^r\, s_1\, p_1{}^\ell)$	p_1		π_3	$(\pi_3{}^r\, s_1\, p_2{}^\ell)$		p_2

Remarquons que les verbes *aller* et *être* – en anglais *go* et *be* – ont les matrices suivantes:

go	*go*	*goes*	*am*	*are*	*is*
went	*went*	*went*	*was*	*were*	*was*

Ces verbes sont respectivement des types $\pi_k{}^r s_i$ et $\pi_k{}^r s_i\, p_1{}^\ell$, où i = 1 ou 2 représente le temps et k = 1, 2 ou 3 représente la personne. Alors qu'en anglais, la matrice de conjugaison possède $2 \times 3 = 6$ entrées, elle a $7 \times 6 = 42$ entrées en français et $3 \times 5 \times 6 = 90$ en latin. En arabe et en sanskrit, les matrices ont des entrées beaucoup plus nombreuses alors qu'en chinois il n'y en a qu'une ou deux.

Nous présentons ci-dessous quelques exemples qui montrent les possibilités de notre théorie des types:

I	*have*		*seen*	*her*			
π_1	$(\pi_1{}^r\, s_1\, p_2{}^\ell)$		$(p_2 o^\ell)$	o			
Have		*I*	*seen*	*her*	*?*		
$(q\, p_2{}^\ell\, \pi_1{}^\ell)$		π_1	$(p_2 o^\ell)$	o			

Il est à noter que la deuxième occurrence de *have* est accompagnée d'une intonation montante et qu'elle a un type différent de la première. Nos deux derniers exemples illustrent comment la théorie des types comprend la théorie des traces de Chomsky, celles-ci étant indiquées ici par un trait:

She	*had*	*been*	*seen*	__
π_3	$(\pi_3{}^{r}\, s_2\, p_2{}^{\ell})$	$(p_2 o^{\ell\ell}\, p_2{}^{\ell})$	$(p_2 o^{\ell})$	

Whom	*have*	*I*	*seen*	__ ?
$(q'\, o^{\ell\ell}\, q^{l})$	$(q\, p_2{}^{\ell}\, \pi_1{}^{\ell})$	π_1	$(p_2\, o^{\ell})$	

On remarque encore que $o^{\ell r} = o = o^{r\ell}$, mais que $o^{\ell\ell} \neq o$.

3. Le rapport entre les catégories en mathématique

Les catégories sont des structures mathématiques abstraites, présentées par Eilenberg et Mac Lane (1945). Sam Eilenberg m'a assuré qu'ils ne furent pas influencés dans le choix qu'ils firent de ce mot «catégorie» par l'utilisation qu'en faisait Aristote pour type, de même que Kant et Ajdukiewicz. Bien qu'il faille distinguer entre «catégoriel» (typé) et «catégorique» (au sens de Eilenberg et Mac Lane), les systèmes de types ont des analogies catégoriques. Ainsi:

1) La théorie intuitionniste des types (Lambek & Scott 1986) correspond aux topoï élémentaires de Lawvere (1975);
2) celle du lambda-calcul typé (connue aussi sous le nom de théorie des preuves du calcul propositionnel positif intuitionniste) correspond aux catégories cartésiennes fermées de Lawvere (1969);
3) celle du calcul syntaxique (Lambek 1958) correspond aux catégories monoïdes doublement fermées (Lambek 1969);
4) celle de la théorie des preuves en logique bilinéaire classique correspond aux catégories non commutatives *-autonomes (Barr 1979);
5) la théorie des preuves en logique compacte bilinéaire correspond à celle des catégories non commutatives *-autonomes compactes.

Lors des questions, après mon exposé à Neuchâtel, on m'a demandé quelques précisions sur le point (1). Lambek & Scott (1986) ont démontré que la théorie intuitionniste des types est essentiellement identique à celle des topoï élémentaires de Lawvere. Plus précisément, à chaque topos **T** est associé un langage interne L(**T**) dont les types sont les objets de **T** (en général ceux-ci ne forment pas une hiérarchie générée librement) et dont les termes de type B – disons avec une variable libre de type A – sont des flèches A→B (quand il n'y a pas de variable libre, prendre A = 1, l'objet terminal de **T**). Par ailleurs, chaque théorie des types (logique d'ordre supérieur) **L** génère un topos T(**L**) dont les objets sont les termes α de type Ω^A pour quelques types A de **L** et dont les flèches $\alpha \to \beta$ sont les termes ζ de type A×B pour lesquels on peut prouver en **L** que:

$$\forall_{x:A}\,(x \in \alpha \Rightarrow \exists_{y:B}\,(y \in \beta \wedge \langle x, y\rangle \in \zeta)).$$

Ainsi **L** et **T** sont des foncteurs entre les catégories, et **L** est adjoint à gauche de **T**. Enfin TL(**T**) est équivalent à **T** et **L** → LT(**L**) est une extension conservative.

Si une théorie des types **L** est présentée comme nous l'avons fait ici, ce n'est pas encore un lambda-calcul typé, mais son extension conservative LT(**L**) l'est. Il en va de même du langage interne de n'importe quel topos **T** ≈ TL(**T**). Effectivement, si $\varphi(x)$ est un terme de type B avec une variable libre x de type A dans L(**T**), nous considérons cela comme une flèche f: A → B et nous pouvons alors définir $\lambda_{x:A}\,\varphi(x)$ comme la flèche correspondante $1 \to B^A$.

Department of Mathematics and Statistics
McGill University
Burnside Hall
805 Sherbrooke Street West
MONTREAL QC H3A 2K6 Canada

Références bibliographiques

ABRUSCI V. M. (1991). Phase semantics and sequent calculus for pure non commutative classical linear logic. *Journal of Symbolic Logic* 56, 1403-1451.

AJDUKIEWICZ K. (1935). Die syntaktische Konnexität. *Studia Philosophica* 1, 1-27.

BAR-HILLEL Y. (1953). A quasi-arithmetical notation for syntactic description. *Language* 29, 47-58.

BARR M. (1979). **- Autonomous categories.* Springer LNM 752.

BUSZKOWSKI W., MARCISZEWSKI W. & VAN BENTHEM J. (eds) (1988). *Categorial Grammar.* Amsterdam: John Benjamins Publishing Co.

CHURCH A. (1940). A foundation for the simple theory of types. *Journal of Symbolic Logic* 5, 56-68.

CURRY H. B. (1961). Some logical aspects of grammatical structure. In: R. Jakobson (ed), *Structure of language and its mathematical aspects*, Proc. Symposia Applied Mathematics 12, A.M.S. Providence R.I., 56-58.

EILENBERG S. & MAC LANE S. (1945). General theory of natural equivalences. *Trans. Amer. Math. Soc.* 58, 231-294.

GEACH P.T. (1971). A program for syntax. *Synthese* 22, 3-17.

GRISHIN V.N. (1983). On a generalization of the Ajdukiewicz-Lambek system. In: *Studies in nonclassical logics and formal systems*. Moscow: Nauka, 315-343.

HENKIN L. A. (1950). Completeness in the theory of types. *Journal of Symbolic Logic* 15, 81-91.

HOWARD W.A. (1980). The formulae-as-types notion of construction. In: J. P. Seldin and J. R. Hindley to H. B. Curry, *Essays on Combinatory Logic, Lambda Calculus and Formalism.* London: Academic Press, 479-490.

LAMBEK J. (1958). The mathematics of sentence structure. *Amer. Math. Monthly* 65, 154-169.

LAMBEK J. (1969). *Deductive systems and categories II.* Springer LNM 86, 76-122.

LAMBEK J. (1993). From categorial grammar to bilinear logic. In: K. Dosen & P. Schroeder-Heister, *Substructural logics*. Oxford: Clarendon Press, 207-237.

LAMBEK J. (1997). An extension of the formulas-as-types paradigm. *Dialogue* 36, 33-43.

LAMBEK J. (à paraître). *Type grammar revisited*. Springer LNAI.

LAMBEK J. & SCOTT P. J. (1986). *Introduction to higher order categorical logic*. Cambridge: CUP.

LAWVERE F. W. (1969). Adjointness in foundations. *Dialectica* 23, 281-296.

LAWVERE F.W. (1975). Introduction to Part I. In: F. W. Lawvere, C. Maurer & G. C. Wraith (eds). *Model theory and topoi.* Springer LNM 445, 3-14.

MONTAGUE R. (1974). *Formal Philosophy*. New Haven: Yale University Press.