Histoire de λ-calcul

Qu'est-ce que c'est un λ-calcul ? Les codeurs sont familiers avec une notion de λ-fonction - une fonction anonyme qui est utilisée dans les morceaux du code qui ne méritent pas d'avoir un méthode nommé : clé de trie, les petits transformation dans les requêtes à la sql etc. Cépendant, la notion de λ-fonction a été pris d'une système de calcul aussi puisant que la machine de Turing (est inventé dans les mêmes années 30s). Dans cet article on va discuter l'histoire de son invention pour mieux comprendre le concept.

Plan

1. Crise des fondaments
2. Axiomes de Peano
3. 1ere version de Lambda calculs
4. Theoreme de Goedel
5. Machine de Turing et calculabilité
6. Thèse de Church-Turing
7. Impact et applications

Crise des fondaments

Disons qu'un ensemble est *simple* s'il n'appartient pas à lui-même. Par exemple, l'ensemble des tout les gens est simple, car cet ensemble n'est pas une personne. Ainsi, l'ensemble des tout les ensemble n'est pas simple par son définition. *L'ensemble de Russel* est un ensemble qui contient tout les ensembles simples et rien d'autre. Est-ce qu'un ensemble de Russel est simple ? Si c'est le cas, par construction il contient lui-même. Donc il n'est pas simple. Mais s'il n'est pas simple il doit contenir lui-même, ce que signifie qu'il est simple. *Contradiction.*

Ce paradoxe a été indépendement trouvé par Russel et Zermelo au début de XX siècle. Ce paradoxe a beaucoup d'autre formulations plutôt didactiques : *paradoxe du menteur*, *paradoxe du barbier*, etc. Cépendant dans une version de Russel ce paradoxe n'utilise que les constructions formelles de mathématique. Cela signifie que telles constructions sont contradictoires elles-mêmes : si nous avons prouvé qu'une formule propositionnelle est à la fois vrai et fausse, le même peut avoir lieu pour n'importe quel théorème. Si on ajoute qu'au début de XXème siècle paradoxe de Russel n'a pas été le seul

paradoxe connu, on voit bien qu'est-ce que c'était le *crise de fondaments* en mathématique.

Ce crise a été refleté sous le numéro 2 dans une liste des 23 fameux problèmes de Hilbert détérminants le développement du mathématique en XXème siècle.

Problème (2ème problème de Hilbert)**.** *Déterminer la consistence de l'arithmétique.*

Même si l'enoncé est simple, on peut dire que pour résoudre ce problème, il faut passer par les étapes suivants.

- Formuler les axiomes de l'arithmétique - i.e., trouver les proposition "minimales" telles que on peut en déduire tout ce que l'on connait jusqu'qu présent.
- Prouver que en partant de ces axiomes, il n'existe pas d'une proposition X, tel que les axiome implique X ainsi que "non X".

Axiomes de Peano

L'arithmétique est un domain de mathématique qui étude les nombres et rélations entre eux. Elle est appliquée partout de premiers années de l'école jusqu'à les conceptes modernes d'astrophysique. Cépendant, pour construire les bases de l'arithmétique, il est presque suffisant de bien déterminer les nombres naturels ainsi que les action qu'on peut faire avec. (Les nombres entiers est une extension pour que opération $x-y$ renvoit toujours un nombre valide, les nombres rationnels aparaissent si on étudie la division. Finalement, les nombres algébrique sont résponsable pour résoudre les équatinos polynomials et le reste - pour "fermer des trous"). Classiquement, les nombres naturels peuvent être définis de même façon qu'on fait quand les petits enfants apprennent à compter: ce résultat est connus dépuis la find de XIXème siècle comme les axiomes de Péano:

1. 1 est naturel;
2. le nombre suivant d'un nombre naturel est naturel;
3. rien ne suivi de 1;
4. si a suive b et a suive c, alorc $b = c$;
5. axiome de recurrence (i.e., si un prédicat $A(x)$ est vrai pour $x = 1$ ainsi que $A(n)$ implique $A(n + 1)$, alors $A(x)$ est vrai pour tout n naturel).

Heureusement, la preuve d'une consistence des axiomes de Péano est un problème beaucoup plus sophistiqué que l'invention de ses axiomes, et l'histoire n'est donc que commencée.

1ère version de Lambda calcul

En 1932 Church a proposé une autre construction qui est connue comme **λ-calcul non-typé**. Malheureusement, son étudiant, Kleene a prouvé que cette construction n'a pas été consistente.

λ-calcul a formalisé une application d'une fonction. L'écriture envisage la compréhension d'une fonction comme une "règle". Et l'écriture classique $f(x)$ pointe plutôt sur le résultat de ce règle.

Rappelon brièvement, qu'est ce que c'est. (Sinon, wiki et les autres articles ou "Eggs and crocodiles") Le brique principal est une fonction. Au lieu de $f(x)$ on écrit $\lambda x.f$. Si on parle de la valeur de $f(x)$ quand $x = a$, on écrit $\lambda x.fa$. Naturelement, ion peut définir une composition... Pour transformer des propositions on a une règle de β-reduction.

Malgré sa simplicité et abstraité, cette construction permet néanmoins rédefinir tout les opérations arithmétiques, la logique Booléen etc. Est-ce que λ-calcul non-typé est un bon candidat pour le rôle de fondament de mathématique ? La réponse est **non**: à cause de Paradoxe de Kleene-Rosser proposé en 1935 par J. B. Rosser et Stephen Kleene qui a été un étudiant de Church. Bien que la propre énoncé de ce paradoxe est trop compliqué pour cet article, nous pouvons décrire la raison d'un problème. Commençons par une phrase "si cette phrase est vrai, alors X", où X est un énoncé qui est évidement faux, e.g., "Allemand et Chine ont une frontière commune". Après, par une analyse logique (**todo**), on peut déduire que n'importe quel énoncé X est vrai. C'est une version non-formel de paradoxe basée sur l'autoréférence. Il peut être formulé en termes de λ-calculs.

Considerons une fonction r définie comme $r = \lambda x.((xx) \to y)$. (rr) β-se réduit en $(rr) \to y$. Si (rr) est faux, alors $(rr) \to y$ est vrai par le principe d'explosion, mais cela est contradictoire avec la β-réduction. Donc (rr) est vrai. On en déduit que y est aussi vrai. Comme y peut être arbitraire, on a prouvé que n'importe quel proposition est vrai. Contradiction.

Théorème de Gödel

Les deux paradoxes discutés ci-dessus, sont basés sur le même concept de l'autoréférence : une proposition ou n'importe quel objet qui réference lui-même (e.g., ensemble des tout les ensembles). Faut-il intérdir l'autoréference dans les constructions mathématiques ? L'idée n'est pas séduisant si on rappel que avec les paradoxes, nous avons jeté dans la poubelle tout les construcitons récursives.

Néanmoins, l'autoréférence a une influence forte sur le fondement de mathématique. Un résultat clé et le plus connu comme la théorème de

l'incomplétude a été prouvé par Kurt Gödel en 1930. Une des intérpretations prétends que la consistence d'une système d'axiomes ne peut pas être prouvée en n'utilisent que ces axiomes (voici l'autoréference !). En particulier, pour prouver la consistence d'arithmétique il faut ajouter les axiomes supplémentaires (qui a été vite fait, en 1936). Le seul problème est que maintenant il faut prouver une autre système...

Pour ceux qui veulent plonger dans le sujet de l'autoréférence, nous pouvons conseiller un livre "Gödel, Escher, Bach : Les Brins d'une Guirlande Éternelle" de Douglas Hofstadter.

La crise des fondements a déclenché plusieurs études sur le sujet. Nous avons brièvement présenté deux modèles qui ont été les candidats sur le rôle de base minimale de l'arithmétique. Cépéndant, le λ-calcul non typé est contradictoire car il contient des paradoxes. La consistence de l'arithmétique Péano a été prouvé un an après, en utilisant la récurrence transfinie par Gerhard Gentzen. D'après le théorème de Gödel, l'ajout d'un proposition supplémentaire dans le système des axiomes a été nécessaire. C'était une idée qui a été manquant pendant presque 50 ans entre la publication des axiomes de Péano et la preuve de Gentzen.

Pour résumer le sujet de l'arithmétique, disons que dans la version moderne on construit le fondement toujours à partir des axiomes de Péano. Pour la consistence, au lieu de la récurrence transfinie, on rajout la théorie des ensemble de Zermelo-Fraenkel avec l'axiome de choix. Néanmoins, chez les mathématiciens il n y a pas de consensus si le deuxième problème de Hilbert est résolu ou non.

Machine de Turing et calculabilité

Cépéndant, comme il est souvent en science, il faudrait étudier le même domaine de point de vue un peu différent. Cela a été fait sur l'autre continent par un jeun étudiant Alan Turing. Il a cherché une solution pour une problème de la décision posé en 1928 par Hilbert et Ackermann : "trouver un algorithme qui détermine dans un temps fini, s'il un énoncé est vrai ou faux". La formalisation d'un terme algorithme a conduit au concept de machine de Turing connu par tout le monde. Entre outre, le théorème de Gödel a été reformuler en thermes d'une machine de Turing. Le résultat a été aussi négative, connu comme une théorème de Turing-Church: "il existe les énoncés pour lesquels on ne peux pas déterminer" (vérifier l'enoncé et le nom d'un théorème).

Thèse de Church-Turing

Le résultat positif. S'il existe les fonctions, qu'il peuvent pas être décidées, on se pose la question, qu'est-ce que ce sont les fonction simple, i.e. les fonctions que l'on peut effectivement calculer. Intuitivement, c'est dont la valeur peut être calculée avec un crayon si on a suffisament de papier et du temps. Mais vous comprenez déjà que les mathématiciens n'acceptent pas les solutions intuitives... Le problème de décision est lié avec une problème de calculabilité. Qu'est-ce que signifie qu'une fonction peut être calculée ? Souvent on se refère sur "des méthodes d'un crayon et de papier". Indépendement, chaque des deux a proposé que toute fonction calculable en thèrmes de crayon et papier peut être calculé par son méthode (lambda-calcul ou la machine de Turing). Les deux propositions - ne sont pas les théorèmes, peuvent pas être prouvées car on ne peut pas formaliser autrement calculabilité. (On doit remarquer ici qu'il y avait le troisième mechanisme de détérminer la calculabilité - les fonction récursive primitive). Relativement vite il a été prouvé que tout les 3 méchanismes sont équivalent. Donc, n'importe lequel peut être utilisé comme une définition de function effectivement calculable.

Résumé. Impacts de λ-calcul

Le concept de λ-calcul a joué une rôle tellement importante dans l'informatique théorique que l'on peut voir ses échos en pratique : dans la plupart des langages de programmation une notion de λ-fonction représente une fonction "anonyme". Cette notion rends la terme connue par des ingénieurs mais la plupart ne connaît pas les détails cachés derrière.

En réalité le concept a eu quatre impacts principaux.

1. *Formalisation d'une notion de calculabilité.* Avant les années 1930s, la définition de calculabilité pouvait être caricaturisée comme "calculable à l'aide du papier, crayon et suffisament du temps". En plus il y avait une intuition que les fonctions récursives doivent définir la classe des fonctions calculables. L'invention de λ-calcul et machine de Turing a relancé une discussion sur la notion de calculabilité. Comme tout les trois concept ont été prouvés équivalents, les mathématiciens se sont mis d'accord a les utiliser comme une définition formelle de calculabilité.

2. *Preuves de calculabilité.* Car les trois concept sont équivalents, n'importe lequel peut être utilisé pour prouver la calculabilité d'un nouvaux objet. Donc on peut considerer λ-calcul comme un outil de plus (en réalité outilisé plus souvant pour prouver qu'un objet est n'est pas calculable).

3. *Preuves formelles.* La version des λ-calcul typés peut être appliquée dans la théorie des preuves. Ainsi, les certains langages de preuves formelles tels que Coq ou AUTOMATH sont basés sur ce modèle.

4. λ-calul est un *langage de programmation* primitif (en nombre de constructions). Comme la machine de Turing est le fondament de tout les langages impératifs, λ-calcul est une base pour les langages fonctionnels tels que Haskell ou OCaml.