Rapport de projet : TP2 IFT2035

Jaydan ALADRO, Karim BOUMGHAR

20 janvier 2023

1 Introduction

Ce document a pour but de témoigner de notre cheminement à travers le TP2 du cours IFT 2035. Nous raconterons donc notre expérience (qui est, par définition, subjective), de ce que nous avons appris, ce qui pourrait être améliorer, etc. Le rapport sera divisé en sections, pour rendre la lecture plus facile.

2 Problèmes rencontrés

Suite au premier projet, nous pensions que le deuxième projet, qui était sur un sujet similaire, serait plus facile. Malheureusement, ce ne fut pas le cas, en effet, le projet était encore une fois assez difficile (probablement beaucoup plus que le premier), dans un langage encore "plus" nouveau que l'était Haskell. Les problèmes rencontrés ont donc été nombreux, nous en énumérons quelques uns dans cette section.

1) Le premier problème rencontré dans ce projet est l'environnement de développement que propose Prolog. Le langage est peu utilisé, et cela parrait. Il n'y a donc pas de "bon" environnement de développement pour ce langage (qui n'apparait même pas dans notepad++ par exemple). Bien que cela ne soit pas un énorme problème, surtout pas pour un projet de petite envergure comme celui-ci, ne pas avoir accès à un "IDE" performant, comme on est habitué avec des langages populaires comme Java, Python, C,... est assez dérangeant.

2) Le deuxième problème, que l'on a rencontré fréquemment tout au long du projet, est que certaines erreurs (surtout de syntaxe) sont très difficiles à remarquer. Par exemple, après avoir écrit la fonction lookup (qui regarde dans l'environnement si une variable est présente et renvoit son type), nous avons fait un appel du genre $T = lookup(Env, X)$, qui ne fonctionnait pas. Bien sur, le problème était qu'une variable était oublié, et qu'on aurait du donc écrire $lookup(Env, X, T)$, mais Prolog ne nous avertissait pas que l'on faisait ce genre d'erreur, et puisque c'étais la première fois qu'on écrivait dans ce langage, on ne savait pas trop comment utiliser les write pour débugger ce genre de problème, cela nous a donc pris quelques heures.

3) Le troisième problème rencontré, et qui est similaire à celui du TP1, est la compréhension de l'énoncé. On aurait pu croire qu'après avoir fait le premier TP, qui était sur un problème presque identique, comprendre le deuxième TP soit assez facile, mais ce ne fut malheureusement pas le cas. L'utilisation de deux langages, le fait d'utiliser des types comme des valeurs, sans qu'on est de relation de transitivité, le fait que l'on cache le tout sous une couche de Prolog rendait la compréhension du projet simplement difficile. Beaucoup de choses qui étaient extrêmement importantes étaient écrites dans de petites phrases cachées un peu partout dans le texte.

4) Le quatrième problème rencontré, et de loin celui qui était le plus gros, était le problème de "récursion dans le coerce". Plus que ça, le coerce a été de loin ce qu'on a passé le plus de temps sur, sans même réussir à le résoudre à la fin. Après avoir passé des dizaines d'heures sur le problème, parlé aux deux démos, envoyer des messages au professeur, on peut quand même affirmé que nous ne comprenons pas vraiment comment le coerce était sensé fonctionné. En effet, après avoir trouvé une forme "à peu près correct" (qui fonctionne pour des cas vraiment simple), on c'est rendu compte que le nouveau problème était qu'il fallait appeler coerce. Après avoir inventer une règle qui ferait en sorte que coerce serait appelé (on nous a dit plus tard qu'il faudrait plutôt faire un "catch" all qui appelerait coerce, qu'il faudrait plutôt trouver nous même ou appeler le coerce et puis finalement le professeur nous a dit que l'appel dans check devrait être suffisant, alors que nous avons jamais réussi à faire fonctionner ce cas, même pour les cas les plus simples), nous ne sommes rendu compte que notre coerce ne fonctionnait pas pour des cas où il devait être appeler plusieurs fois (après plusieurs heures de debuggage (où un debuggeur de haut niveau comme ceux auxquels on a accès pour des langages plus populaire nous aurait sauvé plusieurs heures)). Malgré le fait que nous avons essayé des dizaines de variantes dans notre coerce pour que l'on puisse résoudre le problème, et parler aux deux tpistes / au prof, nous n'avons jamais réussi à résoudre le problème. Nous pensons cependant que nous n'étions pas trop loin de trouver la solution, et qu'il s'agissait probablement d'une des dernières parties que l'on devait compléter pour finir le projet, mais nous ne sommes pas certains.

3 Choix faits

Puisque nous avons pas vraiment eu le temps de finir le projet / de se consacrer au détails, peu de choix ont été fait (autre que ceux qui était décrit dans la donnée). On remarque cependant qu'on a tout de même décidé d'implémenter un forall récursif (même si le professeur a dit qu'on avait pas besoin, puisqu'il n'étais pas dans l'énoncé). On peut aussi également dire que nous avons fait le "choix" de faire un deuxième infer de app qui appelait check à l'intérieur (qui était notre façon de gérer les cas de coerce), mais c'étais plus par nécessité qu'autre chose (bien que le professeur nous a assuré que l'on avait pas besoin d'appeler coerce, et que les règles suffisaient). Si nous avions plus de temps, je pense qu'il aurait été interessant d'implémenter certains addons, comme par exemple le fait que l'on puisse appeler une fonction "fun" avec une liste (permettant ainsi un autre sucre syntaxique qui n'aurait certainement pas été trop difficile à implémenter).

4 Surprises

Quelques surprises ont été rencontré lors de la création du projet, elles seront décrites dans cette section :

1) La première surprise rencontrée était sans doute le langage Prolog en tant que tel. Bien que le langage était assez frustrant, certaines choses que nous avons pu faire (comme la "magie" qui permettait d'unifier dans notre coerce) était assez surprenante. Bien que nous ne recommanderions à personne d'utiliser ce langage, on sort de cette expérience pour une certaine admiration par rapport à certaines choses que Prolog peut faire (qui serait difficile à imiter dans un autre langage).

2) Une autre grosse surprise que nous avons rencontré était la difficulté de la règle du coerce de forall. Bien qu'elle soit présentée comme une règle ordinaire et comme les autres (et qu'elle ne soit pas décrite en particulier), cette règle était de loin la plus difficile à saisir / implémenter.

3) La dernière grosse surprise que nous avons rencontré était les sucres syntaxiques. L'énoncé "cache" un mentionnant brièvement les possibilités, mais en lisant bien les tests qui étaient proposés dans le code, on se rendant compte qu'il y avait plusieurs sucres syntaxiques qui étaient soient très peu décrit dans l'énoncé, ou simplement non-présent et qu'il fallait voir nous même. Ceux du "let" était particulièrement suprenant, car nous avons compris à la fin qu'il y avait vraiment plusieurs cas et que les gérer devenait assez difficile (une chance qu'il n'était pas récursif entre eux).

5 Ce que nous avons appris

Plusieurs choses ont été apprises lors de l'élaboration de ce TP.

1) Prolog : la première chose que l'on retire de ce TP est sans aucun doute une meilleure connaissance de Prolog. Puisque nous avons pas vraiment pratiqué Prolog en classe, ce projet a permis de pratiquer tout ce qui touche à la base d'un nouveau langage (soit la syntaxe, certains petits "trucs", etc). Comme décrit dans la section surprise, il était intéressant de faire des choses qui seraient assez verboses à faire dans des langages populaires en très peu de ligne.

2) Une "compréhension" d'un nouveau système de typage. N'ayant jamais eu à intéragir avec un système qui mélange les types et les valeurs, on a appris quelques choses (bien qu'elle reste très élémentaire, parce que la compréhension était extrêmement difficile) sur ce système.

3) Une autre chose que nous avons appris, est que le nombre d'heures à essayer de comprendre quelque chose ne paye pas toujours. En effet, généralement, dans les autres cours, passer plusieurs heures à essayer de comprendre un concept permet de le comprendre. Cependant, dans ce projet, essayer de comprendre semblait tout simplement impossible, parce qu'on nous demandait quelque chose de quand même avancé, sans même nous avoir appris les bases. C'étais une expérience assez frustrante, et je suis simplement heureux que je n'aurais plus jamais à faire ça (Karim).

6 Ce que nous aurions aimé

Cette section sera une collection que nous aurions aimé avoir / faire et qui aurait grandement améliorer notre apprentissage / appréciation du projet :

1) Avoir des exemples concrets (même si petit) du comportement que l'on s'attend pour le programme aurait été vraiment utile. Par exemple, certains sucres syntaxiques n'étaient pas du tout décrit dans la donnée, alors que ceux-ci étaient essentiels lors de la réalisation. Un petit exemple pour chacun des sucres syntaxiques dans les données auraient donc été apprécié.

2) Avoir un peu plus de temps / que le TP soit donné à une autre période. Lorsque le TP2 a été rendu disponible, il fallait également faire le devoir 10, se préparer pour le final, et se préparer pour les intras / finaux des autres cours. Il n'était donc pas vraiment possible (ou assez difficile) de mettre beaucoup de temps sur le projet, bien que celui-ci en prenne énormément.

3) Avoir un TP différent du premier TP (opinion personnelle de Karim). J'ai trouvé assez décevant que le thème du deuxième travail soit le même que celui du premier. N'ayant pas particulièrement aimé le premier TP (pour toutes les raisons qui sont décrites dans mon premier rapport), je trouvais hyper déprimant de devoir faire quelque chose d'identique (côté thématique), dans un langage pas particulièrement plaisant. Comme indiqué dans mon premier rapport, j'aurais trouvé bien plus pédagogique de nous donner un TP que l'on fait de A à Z (sans pleins de fonctions qui nous ont fourni et qu'on doit utiliser un peu au hasard, sans vraiment comprendre le "flow" du programme), plutôt que ce que nous devions faire.

4) Avoir une meilleure explication sur comment le coerce était sensé fonctionner. Comme décrit dans la section problème, nous avons eu 3 explications différentes de comment le coerce devrait fonctionné (par les 2 tpistes et le professeur lui-même). Il aurait été vraiment utile si cette règle, qui était la difficulté du TP, était décrite en plus grand détail dans la donnée.

7 Remerciement

On tient à grandement remercié les deux tpistes (Antoine et Maxim), qui nous ont grandement aidé (et on été là tout le long) lors de la réalisation de ce TP. Sans eux, l'expérience aurait été horrible.