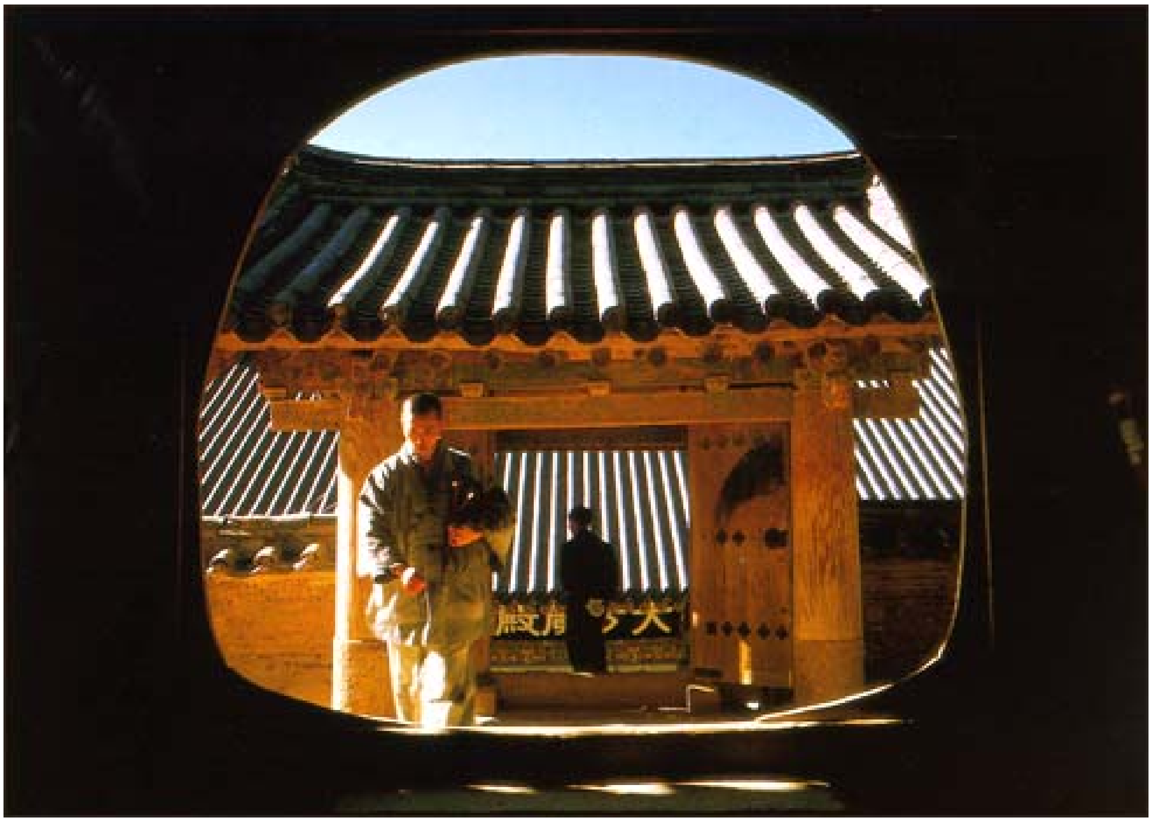

Interprétation Abstraite en Corée du Sud

Charles Hymans

CHARLES.HYMANS@ENS.FR

14 mars 2000

Résumé

Ce rapport rend compte de mon activité au sein du groupe de recherche en analyse statique du Korean Advanced Institute of Science and Technology.[1] Après une brève exposition de mes motivations, je m'efforce de présenter le laboratoire du professeur Kwangkeun Yi, mon hôte. Enfin, je trace les grandes lignes du sujet de recherche qui m'a occupé pendant la durée de ce stage.

1 Pourquoi partir en Corée ?

L'année de cours théoriques prodiguée aux élèves en informatique de l'École Normale Supérieure est complétée par un stage de trois à six mois dans un laboratoire étranger. On peut véritablement considérer ce stage comme un premier contact avec le monde international de la recherche en informatique. Ainsi, nul ne devrait ignorer l'importance des laboratoires américains dans ce domaine.

Alors pourquoi prendre la décision inconsidérée d'effectuer ce baptême de recherche dans un laboratoire sud-coréen ?

Simplement pour lier l'utile à l'agreable : je désirai travailler dans le domaine de l'analyse statique de programmes et de plus je voulai mettre à essai mes connaissances de la langue japonaise. À ma requête, le professeur Patrick Cousot a rétorqué que rares étaient les chercheurs japonais de qualité en analyse statique. Je me suis donc rapidement laissé convaincre qu'il valait mieux

1. Ou de façon plus concise K.A.I.S.T.

Le drapeau coréen est appelé **taegukki**. Il symbolise les principes du Yin et du Yang selon la philosophie orientale. Le centre du drapeau est composé de deux parties égales. La section rouge supérieure représente les forces positives cosmiques du Yang. A l'inverse, la section bleue inférieure représente les forces cosmiques du Yin. Les deux forces accouplées symbolisent les concepts du mouvement continuel; l'équilibre et l'harmonie qui caractérisent la sphère de l'infini. Le cercle est entouré de quatre trigrammes répartis aux quatre coins du drapeau. Chaque trigramme représente l'un des quatre éléments universels : le ciel, la terre, le feu et l'eau.[3]

---

passer la période du premier août 1999 au premier février 2000 à consommer du kimchi[2] plutôt que des sushis.

2 R.O.P.A.S dans le K.A.I.S.T

2.1 Les video-bans de Taejon

Une volonté politique déterminée a fait du Korean Advanced Institute of Science and Technology le premier centre de recherche scientifique de la Corée du sud. Située à distance respectueuse des missiles de la Corée du nord, le K.A.I.S.T n'a donc rien à envier à la réputée Seoul National University. Cependant, Taejon ne demeure pas moins la sixième ville de Corée. J'ai donc dû me contenter de video-bans[4] de seconde zone.

2.2 Le projet LET

La création du Research On Program Analysis System date d'Octobre 1998. Les fonds du laboratoire proviennent d'un programme de développement du "Korean Ministry of Science and Technology". À la tête de ce groupe d'une douzaine de personnes, on trouve le jeune et dynamique professeur Kwangkeun Yi. La figure 1 donne un aperçu de son curriculeum vitae. Réparti

2. Ce plat de chou fermenté et pimenté constitue la base de la nourriture coréenne.
3. Texte de l'Office du Tourisme Coréen.
4. Un video-ban est une petite pièce où deux personnes peuvent regarder une cassette video de leur choix tout en sirotant du thé au ginseng.

- ❖ 1993 - 1995 Member of Technical Staff, Bell Labs., Murray Hill
- ❖ 1993, Ph.D. Computer Science, CSRD, Univ. of Illinois at Urbana-Champaign
- ❖ 1990, M.S. Computer Science, Univ. of Illinois at Urbana-Champaign
- ❖ 1987, B.S. Computer Science & Statistics, Seoul National University

FIG. 1: *Antécédants du professeur Yi.*

dans trois piéces, le groupe est constitué, à un peu moins de la moitié, de post-docs et d'élèves en thèse. Les autres membres sont des élèves au niveau du D.E.A. Chacun a sa propre machine.

Les efforts du laboratoire sont tous dirigés vers la réalisation du projet LET. Pour le professeur Yi, l'environement informatique de demain est un monde d'interconnection globale[5]. Le professeur Yi exprime ainsi sa volonté de créer les outils permettant la compilation efficace de langages d'ordre supérieur et typés (comme ML) pour un tel environement. Les caractèristiques qu'on est en droit d'attendre d'un tel compilateur sont, d'après Yi, la génération d'un code compact (smal**L**), sûr (saf**E**) et enfin intelligent (smar**T**). Pour arriver à ses fins, le professeur Yi souhaite faire une utilisation intensive des techniques d'analyse statique. L'analyse statique de programmes consiste en effet à extraire du texte d'un programme, *avant execution*, des informations relatives à des propriétés du programme *lors de son execution*. Une utilisation intelligente par un compilateur de telles informations peut donc mener à des optimisations non négligeables.

Lors de mon départ, le professeur Yi et son équipe, voulant mettre en pratique leur expérience des deux années précédentes, commençaient l'implémentation d'un compilateur pour un langage type ML.

5. Je cite *global computing environment of the future.*

2.3 La dégustation de bibimpap avec Roberto Giaccobazzi

Il faut reconnaitre que les non-coréens sont encore peu nombreux au K.A.I.S.T[6]. Cette lacune était en partie comblée par de fréquentes visites de chercheurs à renommée internationale. Ainsi, j'ai eu le plaisir d'assister aux présentations d'Olivier Danvy et de Roberto Giacobazzi. Chaque visite était aussi une occasion pour le professeur Yi de faire découvrir à ses invités le plat le moins redoutables, pour des palais occidentaux, de la nourriture coréenne : le bibimpap. Trois mois de coréen n'ont pas suffis à m'enseigner la signification du mot bibim, pourtant je peux affirmer que le bibimpap est un plat à base de riz (pap), de légumes mélangés, et de l'éternel couple piment-chou.

2.4 Liés par le mekchu

Les pages de ce rapport ne suffiraient pas à narrer toutes les surprises, sans aucun doute imputable au fort décalage culturel, que la Corée m'a réservées. Je tiens seulement à donner quelques exemples en relation avec la vie de laboratoire.

6. On pouvait quand même compter une demi-douzaine de français.

e	$::=$	c	constante : $0, 10, \mathbf{true}, \mathbf{nil}, \ldots$
	$\mid$	x	variable
	$\mid$	(e_1, e_2)	contruction de paire
	$\mid$	$(e_1.e_2)$	appel de fonction
	$\mid$	$\mathbf{fun}\ f\ : p_1 \Rightarrow e_1 \mid \cdots \mid p_n \Rightarrow e_n$	définition de fonction avec les patterns p_1 à p_n

TAB. 1: *Syntaxe du langage analysé.*

En Corée, vie privée et vie professionnelle sont étroitement melées. Le membre d'un groupe de recherche se sent presque au sein d'une famille. Il m'a donc fallu prendre régulièrement part aux dîners et beuveries du laboratoire. Les membres du groupe semblent prendre plaisir à avaler des quantités gargantuesques de mekchu [7] en présence de leur professeur.

Cependant, il ne faut en aucun cas se faire d'illusions sur cette apparente familiarité avec le professeur. Le respect de l'enseignant est une valeur très forte dans ce pays confucianiste, et la passivité de certains m'a souvent étonnée. Ainsi, personne n'a bronché lorsque le professeur Yi nous a fait un jour l'annonce par mail, par l'intermédiaire de sa secrétaire, du report d'un mois du prochain traitement...

3 Le sujet de recherche

3.1 Le choix d'un sujet

Pendant ce stage, je pensai travailler avec le professeur Yi ou un de ses élèves. Je pensai avoir à lire des articles, des résultats du laboratoire et implémenter quelque chose. La réalité fut tout autre. J'ai été d'abord surpris, dérouté, puis plutôt heureux d'avoir à résoudre complètement librement un problème. La seule exigeance du professeur Yi était l'utilisation de la méthodologie de l'Interprétation Abstraite[CC77, Cou81].

Le professeur Yi a donc exigé de moi la conception et l'implémentation d'une analyse statique par Interprétation Abstraite. Il voulait que je donne une estimation [8] de la mémoire consommée par un programme d'ordre supérieur.

7. bière

8. Il a d'abord exigé une réponse sous forme de contraintes polynomiales ! Aucun outil

3.2 Méthodologie

Le début du stage fut très difficile. Je pensai traiter le problème de façon élégante en utilisant une sémantique concrète dénotationnelle. C'est en effet une sémantique très courante et peut-être la plus concise pour des langages fonctionels de haut niveau. Je ne suis pas arrivé à mes fins, tout simplement à cause de lacunes en théorie des domaines.

J'ai alors pris la décision de ramener le problème dans un terrain mathématiquement beaucoup plus simple. Toute mon analyse s'est déroulée sans complication à partir d'une sémantique de petits-pas pour machine à pile.

Le langage analysé est un langage fonctionel d'ordre supérieur minimaliste. C'est un langage avec pattern-matching à la ML. Son seul constructeur de données est le constructeur de paires. Le tableau 1 montre sa syntaxe. On donne au langage une sémantique de petits-pas exécutés sur une machine à pile. On utilise des fermetures pour représenter les valeurs fonctionnelles. Il suffit ensuite de formaliser la notion de taille (d'une variable, puis de la mémoire) par le nombre de constructeurs paires, le nombre de fermetures et le nombre de valeurs atomiques. La propriété que l'on veut extraire du programme devient une fonction entre la taille des arguments passés au programme et la taille de la mémoire consommée lors de l'exécution du programme. Pour inférer une telle propriété, il est nécessaire de partir d'une sémantique concrète de trace. À partir de là il ne reste plus qu'une inconnue : le langage étant d'ordre supérieur, on ne connait pas *à priori* son flot de contrôle. Qu'à cela ne tienne, on se contente d'appliquer une analyse du graphe d'appel[9], puis on déroule de façon presque automatique la méthodologie générale de l'Interprétation Abstraite. On peut signaler que l'analyse est complètement paramétrée par le treillis numérique utilisé.

3.3 Résultats

En utilisant le produit des deux treillis numériques de Karr [Kar74] et des intervalles [CC77] on obtient déjà des résultats satisfaisants. Le programme de la figure 2 prend en argument un arbre formé de paires et retourne une copie de cet arbre. Ainsi, l'implémentation actuelle de l'analyse peut prédire que l'execution de ce programme va mener à la création en mémoire d'une structure de la taille de l'arbre passé en paramètre. Mais de plus, il est amu-

–treillis numérique– de la sorte n'existe à ma connaissance pour le moment.

9. 0-CFA suffit amplement.

```
((fun f = (a, b) -> ((f.a), (f.b))
        | 1      -> 1)            .x)
```

```
Karr analysis results :
  env.x.val - env.x.pair = 1
  env.x.pair - mem.pair = 0
  env.x.ferm = 0
  mem.val - mem.pair = 1
  mem.ferm = 1
  env.f.val = 0
  env.f.pair = 0
  env.f.ferm = 1
```

FIG. 2: *Fonction de copie et résultat de l'analyse.*

sant de noter que la machine infère aussi la relation : `mem.val - mem.pair = 1`. C'est à dire que la structure crée en mémoire a une valeur atomique de plus que de constructeurs de paires. Ceci correspond simplement au fait que tout arbre binaire a une feuille de plus que de noeuds internes.

4 Conclusion

En conclusion, je me contenterai de reprendre la remarque judicieuse de mon tuteur Granboulan : "J'estime qu'on ne peut se permettre d'envoyer, dans des pays comme la Corée, que des personnes qui ont déjà un intérêt pour l'Asie. Ceux qui recherchent uniquement un dépaysement, on peut, par exemple, les envoyer en Inde". Il est indéniable que ce séjour ne fut pas, tant sur le plan humain que celui de la recherche, facile. Pourtant, il fut, peut-être par là même, formateur et extrèmement enrichissant. J'espère ainsi que la chance de partir dans un pays un peu plus *exotique* que les États Unis sera reproposée aux normaliens en soif de nouveauté.

Anion Ikajuseyo. [10]

Références

[CC77] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977. ACM Press, New York, NY.

[Cou81] P. Cousot. Semantic foundations of program analysis. In S.S. Muchnick and N.D. Jones, editors, *Program Flow Analysis: Theory and Applications*, chapter 10, pages 303–342. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1981.

[Kar74] Michael Karr. Affine relationships among variables of a program. *Acta Informatica*, 6:133–151, May 1974.

10. *Au revoir* dans le plus poli des trois niveaux de langue du coréen.