Programmation 1, Projet 2

L'Inféatrice

David Baelde, Alexandre Drewery & Pablo España Gutierrez

28 novembre 2202

> "All right," said [the Inféatrice]. "The Answer to the Great Question..."
> "Yes...!"
> "Of Life, the Universe and Everything..." said [the Inféatrice].
> "Yes...!"
> "Is..."
> "Yes...!!!...?"
> "Forty-two," said [the Inféatrice], with infinite majesty and calm.
>
> Douglas Adams, *The Hitchhiker's Guide to the Galaxy*

De récentes fouilles archéo-électroniques semblent confirmer une légende ancienne, affirmant qu'une civilisation informatisée du XX$^{\text{ème}}$ siècle aurait développé une intelligence artificielle symbolique capable de trouver par elle-même des dérivations au moyen de systèmes de règles d'inférences arbitraires.

Des fragments de ce système mythique, dénommé Inféatrice, ont pu être extraits des rémanences de rayonnements électromagnétiques anciens. Il s'agit de notes de travail, de fichiers d'entrée de l'Inféatrice, et de fragments de code Haut-Caml. Ces informations parcellaires donnent une idée vague des capacités que pouvait avoir ce système, ainsi que de son mode de fonctionnement. Pour tenter de mieux comprendre ce système et la civilisation qui l'a engendré, vous avez été missionnés, en tant qu'archéo-informaticiens, pour tenter de reconstruire l'Inféatrice à partir des données historiques.

Votre tâche sera facilitée par le fait que l'antique langage Haut-Caml est déjà bien connu, et a été reproduit aujourd'hui à l'identique sous le nom d'OCaml. Bien entendu, vous respecterez la ligne de conduite de tout bon archéo-informaticien : aucun des fichiers historiques ne devra être modifié et, quand le mode de fonctionnement historique est connu, vous le reproduirez scrupuleusement.

1 Fonctionnement général

Les fichiers d'entrée de l'Inféatrice décrivent des systèmes de règles d'inférence. On a retrouvé des notes des concepteurs qui donnent une idée du mode de fonctionnement de l'Inféatrice :

```
$ ./inferatrice 2_arith.inf

Requête? plus(z,s(z),s(z)).
  # La requête se réduit à plus(z,z,z) puis
  # une dérivation est trouvée.

Requête? plus(z,X,Y).
  # Requête résolue: il suffit que X = Y
  # pour avoir une dérivation.

Requête? plus(X,z,X).
  # L'inféatrice ne trouve pas seulement une solution
  # mais une infinité de possibilités!
  # Première solution:
  #   X = z    (choix de la première règle d'inférence)
  # Deuxième solution:
  #   X = s(z) (choix de la seconde puis de la première règle)
  # Etc. vers l'infini... et au-delà?
```

Une autre note, sibylline, semble donner des indications sur le principe général de fonctionnement de l'Inféatrice, au delà d'un exemple particulier :

```
L'inféatrice tire sa force de très peu d'ingrédients:
 - l'unification,
 - le retour sur trace,
 - une recherche en profondeur,
 - toujours de haut en bas et de gauche à droite!
```

Un échange entre programmeurs du système donne des informations sur le style de l'implémentation :

```
Simon:
 - Mais quelle idée, ces états mutables de partout!
Xavier:
 - Bof, c'est juste les reines en un peu plus compliqué...
   et c'est le prix à payer si on veut profiter du GC gratis.
```

2 Tests

Les fichiers exhumés correspondent à deux exécutables. Avant l'Inféatrice elle-même, un exécutable de test permet de vérifier que tout fonctionne bien.

Le journal de bord d'un programmeur atteste que les tests fonctionnaient sur le système historique.

```
26 novembre 1972, 23h00:
- Les tests ne marchent plus, c'est dingue!
26 novembre 1972, 23h32:
- C'est bon, ça remarche. J'ai oublié ce qui n'allait pas.
```

Ne cherchez pas à faire passer tous les tests immédiatement : faites compiler, puis validez les tests un par un. Commencez par mettre en place une implémentation bidon des modules requis pour compiler, puis tâchez de faire passer les tests pour les modules `Term`, `Unify` et enfin `Query`.

3 L'Inféatrice

L'exécutable de l'inféatrice utilise des analyseurs lexical et syntaxique qui ont été retrouvés, fort heureusement car ils reposent sur une technologie aujourd'hui fort mystérieuse. Visiblement, le système s'appuie sur un module `Convert` dont l'implémentation n'a pas été retrouvée. Des notes indiquent que ce module peut être développé en plusieurs étapes successives :

```
24 novembre 1972, 13h37:
- J'ai encore amélioré Convert, ça poutre!
```

Vous pourrez vous appuyer sur les quelques fichiers d'entrée, et leurs commentaires, pour comparer votre Inféatrice avec son modèle historique. Il est probable que vous ne saurez pas développer en une semaine, et sur la base d'informations aussi fragmentaires, une version aussi élaborée que l'originale.

4 Au delà du travail archéo-informatique

Maintenant que vous avez reconstruit une réplique fidèle de l'Inféatrice, utilisez-la pour automatiser le cours de Programmation 1 : spécifiez les règles de typage et la sémantique à grands pas d'un lambda-calcul plus ou moins enrichi, et utilisez l'Inféatrice pour typer et exécuter des programmes !

Vous pouvez aussi envisager plusieurs extensions possibles de l'Inféatrice, par exemple produire des arbres de dérivation lisibles !