Programmation DM 2 : *Chameaux et λ-termes*

{ jrobert, fbouchez, apardon } @ens-lyon.fr
http://perso.ens-lyon.fr/florent.bouchez/

27 novembre 2007

Formalités : *ce devoir est à faire en binômes, et à rendre avant le* **9 janvier 2008 à 23h59**. *Vous devez impérativement déclarer vos binômes avant le* **1er décembre 2007**. *Vous devrez aussi rendre une préversion minimale avant le* **20 décembre 2007 à 23h59**.

Vous devez nous envoyer par e-mail un rapport (LaTeX compilé en `ps` *ou* `pdf`*) et une archive* `tar.gz` *contenant tous vos fichiers. Si vous n'êtes toujours pas à l'aise avec LaTeX, vous pouvez aussi utiliser un autre logiciel de mise en page, mais soignez la présentation.*[1] *Indiquez bien vos noms dans le rapport et l'archive (nommez-les* `login1-login2.pdf` *et* `login1-login2.tar.gz`*).*

Introduction

Dans la province de chameauland, tout était calme et tranquille. Une petite communauté de chameaux informaticiens — unique en son genre puisque tout le monde sait que les chameaux sont plutôt mathématiciens par nature[2] — blatérait paisiblement tout en jonglant machinalement avec des λ-termes quand le drame survint. Une exception surgit du désert environnant et mélangea tous les petits λ-termes qui étaient auparavant entreposés en petits tas bien rangés. Affolés, les λ-termes formèrent des bandes paranoïaques et commencèrent à agresser les pauvres chamelons qui avaient le malheur de passer par là.

Vous êtes membre de la garde des valeureux chameaux-protecteurs. Votre mission est de remettre de l'ordre dans la communauté en remettant à leur place les λ-termes farouches en l'échange de quoi, en remerciement, vous aurez le droit de promener le Roy toute la semaine prochaine. Mais prenez garde ! D'autres que vous ont également été mandatés pour ce travail et un seul chameau-protecteur aura le privilège de supporter le séant de sa majesté ; certains de vos adversaires n'hésiteront pas à vous pousser dans les Ω^3 mortels qui minent le terrain.

Avertissement : dans un but ludique, nous avons présenté ce devoir comme un petit concours où vous vous mesurez aux programmes des autres binômes. Mais il n'est évidemment *pas* question d'attribuer la meilleure note au chameau le plus fort, puis de diminuer jusqu'au dernier qui a zéro. Il est tout à fait acceptable d'implanter un chameau minimal qui tout seul réussit à remettre tous les λ-termes à leur place, et en présence d'autres chameaux se fait complètement laminer.

1 Les règles

1.1 Le monde

Le monde dans lequel votre chameau évolue est un damier rectangulaire. Chaque case peut être du sable, un palmier, un Ω ou une base (contenant initialement des λ-termes para-

[1] Et dans tous les cas, utilisez un correcteur orthographique.

[2] cf. *Pyramids* de Terry Pratchett.

[3] $(\lambda x.xx)(\lambda x.xx)$ ou autres atrocités.

noïaques). Il n'est pas possible de traverser les palmiers. Si un chameau tombe dans un Ω, il est éliminé. Les bords du monde se comportent comme s'il y avait des palmiers au delà.

Au début de la partie, on ne trouve de λ-termes que dans les bases.

Une case du monde est désignée par ses coordonnées (x, y) comprises entre $(1, 1)$ (le coin Nord-Ouest[4]) et (L, H) (la largeur et la hauteur du monde).

1.2 Les chameaux

Chaque chameau possède un identifiant unique (entier positif), une largeur d'esprit (capacité mémoire donnée par un entier) et commence la partie avec une certaine quantité d'eau (encore un entier) pour survivre dans le désert.

Trois commandes permettent de faire interagir votre chameau avec le monde :

- `Deplace` : pour déplacer le chameau sur une case adjacente dans une des quatre directions Nord, Sud, Est, Ouest ;
- `Prends` : pour prendre un ou plusieurs λ-termes. Il n'est possible de prendre un λ-terme que s'il se trouve sur la case du chameau, et s'il reste au chameau assez de largeur d'esprit ;
- `Pose` : pour relâcher un ou plusieurs λ-termes.

A chaque tour de jeu, vous devez envoyer une unique commande à votre chameau. Vous devez aussi miser une partie de votre eau pour cette commande ; en solo, il suffit de miser le minimum : 1 litre à chaque tour. Si un chameau envoie une commande illégale ou impossible (se déplacer sur un palmier, prendre un λ-terme qui n'est pas sur sa case, déposer un λ-terme qu'il n'a pas en mémoire, prendre des λ-termes plus gros que sa mémoire), l'action illégale n'est pas effectuée. Dans tous les cas, il perd la quantité d'eau misée. S'il n'a plus d'eau, il meurt déshydraté et finit en un petit tas d'ossements dans un coin du désert.

1.3 Les λ-termes

Chaque λ-terme possède un identifiant unique (entier strictement positif), un besoin en espace vital (la place qu'il occupe en mémoire), et une destination (coordonnées d'une case de la carte). Si un chameau meurt, les λ-termes qu'il avait en mémoire meurent dans d'atroces souffrances et disparaissent du monde. Les λ-termes déposés à destination s'endorment paisiblement et ne peuvent plus être ramassés (on considère qu'ils disparaissent du monde). Un λ-terme déposé à un autre endroit que sa destination ré-entre en transe paranoïaque et reste là jusqu'à ce qu'un chameau le récupère.

Si un chameau est sur une case contenant des λ-termes, il peut choisir d'en ramasser une partie (zéro, un, deux, ..., tous) s'il lui reste assez de capacité mémoire. De même, à tout moment, un chameau peut choisir de déposer une partie des λ-termes qu'il a en mémoire.

1.4 Multijoueur

Cette partie peut être sautée en première lecture ou si ça ne vous intéresse pas.

[4] En haut à gauche, quoi.

1.4.1 Miser

Quand un chameau veut faire une commande (déplace, prends ou pose), il doit également miser une certaine quantité d'eau. Les commandes des chameaux seront alors effectuées dans l'*ordre décroissant* des mises. Cela est important si, par exemple, deux chameaux veulent se déplacer sur une même case (voir section suivante). En cas d'égalité, l'ordre d'exécution sera (pseudo)aléatoire.

Une mise est un entier *relatif* non nul : pour que la commande de votre chameau soit exécutée en premier, il faut miser plus que les autres ; à l'inverse, un entier négatif augmentera les chances que votre action soit résolue en dernier. Dans les deux cas, c'est la valeur absolue de la mise qui sera retranchée de la réserve d'eau du chameau : si vous avez misé -10, votre réserve d'eau sera diminuée de 10 litres.

Il est interdit de miser 0 (c'est une mise illégale qui élimine le chameau). Un chameau avec une réserve d'eau vide meurt. Comme la mise minimum est de 1 (ou -1), la durée de vie d'un chameau est limitée par sa réserve d'eau initiale.

1.4.2 Pousser

S'il y a plusieurs chameaux qui évoluent dans le monde en même temps, ça peut devenir compliqué car les chameaux peuvent se pousser ! En effet, deux chameaux ne peuvent se trouver simultanément sur la même case. Un chameau 1 pousse un chameau 2 si le chameau 1 se déplace[5] sur la case du chameau 2. Le chameau 2 est alors poussé sur la case adjacente dans le sens de la poussée. Si cette case est mortelle (Ω), il est éliminé. Si la case est un palmier, aucun des deux chameaux ne bouge mais le chameau 2 est tout de même considéré comme étant poussé. Le poussage est transitif : si la case où est poussé le chameau 2 est occupée par un chameau 3, ce dernier est également poussé. Dans ce cas les chameaux 2 et 3 sont considérés comme poussés, et de même les chameaux ne bougent que s'ils peuvent tous bouger.

Quand un chameau est poussé, il est momentanément désorienté. Un chameau désorienté ne peut effectuer sa commande, et reste désorienté jusqu'à la fin du tour. En conséquence, si un chameau est poussé avant d'effectuer sa commande, il ne fait aucune action ce tour-ci.

De plus, si un chameau est poussé, il perd un λ-terme choisi au hasard parmi les λ-termes qu'il transporte. Ce λ-terme tombe sur la case avant que le chameau ne soit poussé (si possible) sur une autre case. Un chameau contenant des λ-termes et considéré comme poussé perd nécessairement un λ-terme, même s'il ne bouge pas (cas du palmier).

Exemple : Voir la figure 1 pour plusieurs scénarios possibles (`'ϒ'` est un palmier, `'.'` est du sable).

1.5 Mort

Un chameau qui meurt disparaît du monde.

1.6 Fin de la partie

La partie se termine quand tous les λ-termes ont disparu ou que tous les chameaux (honte à vous !) sont morts.

[5] Pousse toi d'la que j'm'y mette !

FIG. 1 – 1 veut aller à l'*Est*, 2 au *Nord* et 3 ne bouge pas.

Initialement	1 bouge en premier	2 bouge en premier
... 1.. .2.	.1. .2. ...	... .12 ...
... .1. .2.	... .21 ...	.1. .2. ...
.♈. .1. .2.	.♈. .21 ...	.♈. .1. .2.
... .3. .1. .2.	... .3. .21 ...	.3. .1. .2. ...

1.7 Score

Un chameau apportant un λ-terme à destination gagne un nombre de points égal à la taille de ce λ-terme. Le score final n'est pas très important mais permettra de comparer vos différentes stratégies, ou d'affronter d'autres chameaux. Dans une partie standard, la taille des cartes variera entre 1 case et 1000×1000 cases ; le nombre de λ-termes entre 1 et 10,000. La quantité d'eau initiale donnée sera au plus de 1,000,000,000 (litres), la taille d'un λ-terme sera au maximum de 1,000,000,000 et la capacité de mémoire d'un chameau sera d'au plus 1,000,000,000.

2 Mise en place

Vous devrez dans un premier temps fabriquer un plateau de jeu en bois et confectionner des petits chameaux en cuir afin de pouvoir disputer des matchs amicaux...

Non, sérieusement, comme le jeu peut être joué en multijoueur, une interface client-serveur (tout en OCaml) a été choisie :

- Le serveur : pas d'inquiétude, on s'en est occupé.
- Les clients : c'est votre boulot. Mais là encore pas d'inquiétude, une interface minimale vous sera donnée.

Vous trouverez sur le site ouéb[6] des TPs plusieurs choses utiles :

- une liste d'hôtes auxquels se connecter pour tester votre client ;
- le serveur (fichier exécutable) si vous voulez travailler à l'extérieur de l'ENS ;
- une archive, contenant l'interface minimale pour communiquer avec le serveur ;
- éventuellement une FAQ si vous avez plein de questions ;
- d'autres suprises...

[6] `http://perso.ens-lyon.fr/florent.bouchez/dmchameaux`

L'archive contient plusieurs fichiers :
- `types.mli` cette signature contient tout les types nécessaires à l'interface avec le serveur. A lire absolument pour savoir comment jouer.
- `protocole.mli, protocole.ml` l'interface et l'implantation pour les communications avec le serveur. Il n'est pas utile de les lire.
- `jouer.mli` l'ensemble des fonctions vous permettant de faire des actions et de connaître les actions ayant été effectuées par les autres joueurs.
- `jouer.ml` l'implantation de l'interface `jouer.mli`. Vous n'êtes pas obligés de lire ce fichier.
- `main.ml` comme exemple d'utilisation de l'interface `jouer.mli`. C'est ce fichier qui commandera votre chameau. Vous devrez modifier ce fichier pour y ajouter votre touche personelle.
- `Makefile` pour compiler le tout.

Dans l'état actuel, vous pouvez compiler avec la commande `make` qui vous crée un exécutable `client`. Lancez le avec l'option `-help` pour avoir la liste des options. Le programme se connecte à un serveur et vous pouvez déplacer votre chameau via les touches 'h','j','k' et 'l'.[7] Bien évidemment, votre but est que le programme joue ensuite sans intervention humaine.

Lisez le fichier `main.ml` en vous référant à `jouer.mli` ; ce fichier utilise les deux fonctions principales de `jouer.mli` qui correspondent au protocole suivant :

Initialisation : on signale qu'on est un joueur via la fonction `Jouer.init` qui renvoie la carte du jeu, l'identifiant de votre chameau, sa largeur d'esprit et sa quantité d'eau initiale, ainsi que la liste des positions de tous les chameaux.

Le jeu : La partie dure un certain nombre de *tours*. A chaque tour, vous devez envoyer la mise et la commande qu'effectuera votre chameau via la fonction `Jouer.joue_un_tour` qui vous renverra alors la liste des actions ayant été effectuées par tous les chameaux ainsi que la liste des λ-termes se trouvant aux pieds de votre chameau.

Travail préliminaire à faire avant le **20 décembre 2007 à 23h59** *:*
- Définir vos structures de données (stockage du monde, des chameaux et λ-termes).
- Les mettre à jour après avoir reçu la liste des actions et des termes aux pieds de votre chameau.

Afin de vérifier que votre code est correct, vous devrez implanter une stratégie *plus_naif_tu_meurs* pour votre chameau. Celui-ci doit aller tout le temps à l'ouest, en contournant éventuellement les obstacles par le nord ou le sud, et prendre et poser des λ-termes sur son chemin. Plus précisément, si `dir` est une variable initialement à `Nord`, votre chameau doit :

1. s'il est sur la destination d'un de ses λ-termes, le poser ;
2. sinon, s'il y a au moins un λ-terme, et qu'il reste de la place, le(les) prendre (sans surcharge) ;
3. sinon s'il est possible d'aller à l'ouest (pas de palmier ou Ω), y aller ;
4. sinon s'il est possible d'aller vers `dir`, y aller ;
5. sinon changer `dir` (échanger `Nord` et `Sud`) et réessayer 4

[7] À la manière d'un très bon éditeur de texte, et d'un excellent jeu en mode console.

6. sinon quitter la partie (plus possible d'aller ni à l'ouest, ni au nord, ni au sud).

Un des serveurs en ligne proposé contiendra une carte spécifique prévue à cet effet.

3 Stratégies pour un chameau

Dans cette partie, vous allez devoir utiliser les données récoltées dans les sections précédentes pour élaborer une (des) stratégie(s) un peu meilleure(s) que plus_naif_tu_meurs.

Pour chaque sous-problème *au moins* deux stratégies sont demandées (une bête et une moins bête). On vous en propose quelques unes, mais vous pouvez aussi en trouver des meilleures (n'hésitez pas à vous documenter, et à être créatifs). L'important est que votre programme puisse être modifié de façon à supporter d'autres stratégies que celles auxquelles on a pu penser.

3.1 Recherche de chemin

Pour aller d'un point à un autre, votre chameau pourra opter pour différentes stratégies :
- marcher au hasard jusqu'à arriver à destination (mais sans finir dans les palmiers, dans les Ω ou carrément dans les choux).
- y aller en ligne droite si le chemin est libre, sinon suivre les palmiers jusqu'à ce que la ligne droite soit possible.
- choisir d'emprunter le plus court chemin à l'aide de l'algorithme de Dijkstra ou A^* (« A star »).

3.2 Trouver des λ-termes

Au début du jeu, votre chameau est sur une case où il n'y a aucun λ-terme. Il va donc falloir aller au moins sur une base pour voir s'il y a du monde à sauver. Pour choisir une base parmi toutes celles qui sont disponibles, on vous propose ces stratégies :
- choisir une base au hasard ;
- choisir la base la plus centrale (la plus éloignée des bords) ;
- choisir la base la plus proche de votre chameau ;
- visiter toutes les bases pour connaître tous les λ-termes.

Vous aurez aussi besoin de rechercher une base lorsque vous n'avez connaissance d'aucun λ-terme à sauver. Pensez donc au fait qu'il sera inutile de visiter des bases que vous savez vides.

3.3 Sélection des λ-termes

Arrivé sur une base, il faut choisir quels λ-termes prendre, de façon à remplir au mieux votre chameau. Quelques propositions de stratégies assez naïves :
- essayer de prendre les λ-termes un à un, dans n'importe quel ordre, jusqu'à ne plus avoir de place pour personne ;
- la même chose mais en choisissant les plus gros d'abord ;
- la même chose en prenant aussi en compte la destination (les plus proches en premiers).

3.4 Ordre de la livraison des λ-termes

Une fois les λ-termes en poche, vous devez les amener à destination, et il va donc falloir décider de l'ordre de livraison. Une fois encore, quelques idées :

- choisir un λ-terme au hasard et l'amener à destination par la même méthode qu'en 3.1 ;
- à chaque étape, choisir le λ-terme dont la destination est la plus proche (et donc dépendant d'une notion de *distance*) ;
- choisir des destinations proches d'autres bases de λ-termes pour refaire le plein facilement ;
- résoudre le TSP (*Travelling Salesman Problem*) avec un algorithme linéaire (à vous la fortune) ;
- déposer des λ-termes en cours de route pour les reprendre plus tard et les amener à destination.[8]

Travail à rendre au plus tard le **9 janvier 2008 à 23h59**.

Pour ce DM, vous devez rendre un code qui compile, et dont l'exécutable produit est capable de se connecter à un serveur et ranger tous les λ-termes tout seul comme un grand. Le rapport devra être soigné : vous y indiquerez le mode d'emploi pour activer telle ou telle stratégie mais aussi et surtout vos choix, les difficultés rencontrées, les stratégies inventées ou vos méthodes de programmation, bref, le parcours de votre réflexion sur ce DM.

Un travail correct à ce niveau — code correct, modulaire, implantant plusieurs stratégies, associé à un rapport correct — vous garantira une note correcte. La suite (multijoueur) est faite pour vous amuser,[9] *mais quelques points lui sont réservés car quand même c'est chouette d'avoir une carotte au bout.*[10]

---

4 Stratégies pour régner sur le monde des chameaux

Là où ça devient drôle, c'est quand les autres s'en mêlent, et toute votre petite stratégie mijotée aux petits oignons tombe à l'Ω si un autre chameau vous pique tous les λ-termes sous votre nez ou vous pousse sournoisement dans l'Ω.

En mode multi-chameaux, les stratégies doivent en général être adaptées. C'est possible de ne pas le faire, mais votre score peut être grandement amélioré en le faisant. Toutes les stratégies mises en place ont alors un seul but : pourrir les autres. Distinguons deux types de méthodes :

- méthode passive : se débrouiller pour éviter les autres et être balèze (donc ramener plus de λ-termes ce qui pourrit les autres) ;
- méthode active ou « aggressive » : intervenir violemment avec les autres chameaux pour leur faire comprendre qui est le plus fort (p.ex. les pousser dans les Ω pour les tuer ou simplement pour leur piquer leurs λ-termes).

Voici maintenant quelques pistes pour développer vos idées qui assiéront votre domination sur le monde. Nous vous laissons le soin de penser les stratégies vous-même, puisque vous donner des stratégies toutes faites supprimerait la saveur pimentée des combats.

[8] Si, ça peut être utile.

[9] On espère que c'était déjà le cas.

[10] Et oui, les chameaux aiment aussi les carottes.

- Est-ce bien prudent de s'approcher des Ω ?
- Privilégier les couloirs de quelle largeur ? (une, deux, plus de cases ?)
- En position dangereuse, jouer le premier ou le dernier ?
- En position de mettre l'autre en danger, même question.
- Et s'il existait un mode « *deathmatch* »(le dernier survivant a gagné) ?
- Et s'il existait un mode coopération (p.ex. un match en équipes 2v2) ?

Pour finir, n'hésitez pas à poser des question, à râler si les serveurs plantent, à proposer des cartes rigolotes, des stratégies rigolotes, et nous prévenir si vous faites des duels.
