Rapport Système de négociation tarifaire

Antoine POSNIC, Antoine LEVAL

January 12, 2018

1 Introduction

Ce projet de TP a pour but de développer une application de négociation tarifaire entre des acteurs, au sujet d'une transaction définie. Le but étant de délivrer un produit fini garantissant la sûreté des propriétés attendue. A terme, les applications de chacun sont mises en ligne afin d'attaquer les programmes des autres binômes.
Comme a chacun nous avons utilisé l'outil Isabelle HOL, un assistant de preuve nous permettant d'avoir accès au outils nécessaires pour tester nos propriétés. Afin de se convaincre à avoir un programme "sûre" avant d'exporter sur le projet Scala produit a cet effet.

2 Notre Code

2.1 Fonctionnement

Le sujet nous ayant été annoncé comme compliqué, nous avons opté pour la défragmentation maximale de nos fonctions. Cela nous permet à la fois de tester plus précisément nos lignes de code (pour desceller une erreur notamment), ainsi que, si nécessaire, prouver de petit bouts de codes simples.
Au final nous n'avons pas prouvé de fonctions, mais cette masse d'appels simples nous a définitivement simplifié la vie.

2.2 Types de variables

Mis a part les types fournis initialement, nous avons crée 4 nouveau types:

- `messageList` qui nous sert a stocker une suite de message à process
- `vals` qui nous sert a stocker un prix ou none. Particulièrement utile par la suite pour gérer les cas ou il n'y a pas encore de valeurs assignées
- `transidvals` un quadruplet, contenant la `transid` (informations identifiant la transaction sujette), les deux valeurs proposées par le client et le marchand respectivement, ainsi qu'un boolean déterminant si cette transaction, malgré les prix des acteurs, est close ou non (Canceled par le marchand).
- `transBdd` un doublet de listes, la première étant une liste de `transidvals` qui représente le stockage des transactions qui ne sont pour le moment pas conclue, ou celles cancelled. Ainsi qu'une liste de `transaction` représentant les transaction validées.

2.3 Division du programme

Travaillant avec une variable du type `transBdd` on peut observer une coupure distincte dans notre code. Une première partie s'occupe de la gestion et mise a jour de la liste de `transidvals` avec ses variables. Tandis qu'une seconde s'occupe de la validation et le maintien de la liste de `transaction`.

3 Propriétés

Neuf propriétés nous ont été imposées. Elle permettent, si bien implémentées, de faire tourner nitpick et quickcheck, des outils cherchant des contre-exemples. Le soucis étant de les transcrire d'un langage naturel à un langage de lemme en isabelle.

3.1 Procédés

Afin de limiter au maximum notre domaine de sûreté, il nous fallait éviter d'ajouter de nouvelles fonctions pour développer nos lemmes. Ainsi, uniquement deux nouvelles fonctions ont été ajoutées pour ces derniers. Il s'agit de `getMinAmmInit` et `getMaxAmcInit` qui respectivement rendent la valeur qui serait prise en compte pour un marchant ou un client. Ignorant les `Cancel`.
Sinon, nous utilisons la librairie d'Isabelle afin d'accéder a des fonctions manipulant des listes, qui sont prouvés. Par exemple `List.member` ou encore `List.distinct`.

3.2 Nos lemmes erronés

Nous avons rencontré des soucis pour cette partie. En effet nos lemme 6 et 9 ne définissent pas suffisamment bien les propriétés respectives. Nous nous sommes rendu compte du soucis lorsque quickcheck trouvait un contre-exemple qui était due au propriétés erronées.

4 Les attaques

Une fois mise en ligne, chacun des binômes sont mis à disposition pour tester et trouver les failles dans les propriétés de chacun.

	Propriété 1	Propriété 2	Propriété 3	Propriété 4	Propriété 5	Propriété 6	Propriété 7	Propriété 8	Propriété 9
LeClech_Iquel					O		O		
<JAMMALKADRI>				H ; J ; K	A ; H ; O	D	O	E ; F ; G	
Le_Haye						C ; D			P
Deschamps_Esnault									
Landure_Cusson									
BOUYAMINEetLALKHA									
Ghilas									
BORDIER_JB_&_LARZILLIERE_Charles	Q								
Daniel Nelle					P		N		
<PIERRE_LESAINT>					A ; B ; O		O	E ; F ; G	
AudreyMartin KillianRousseau									
<NEMEH_RABE>					O		O		
<POSNIC_LEVAL>									
Pierre Le Luron & Alan Turpin									
<MERZOUK_SFOUA>									
NguyenMirabile									
<MIOLA_MOUTARAJJI>					A ; L ; O		O	E ; F ; G	
Emilien Petit, Sophie Sennoun									
<VANSTEENE_MINARD>									
Anne									
Banse	Q								
barrere								E ; F ; G	
Bartoccioni									
Bautista									
Bellec									
Bonneau								E ; F ; G	
bordais					N		N		
Bourel					A ; B ; E ; G ; N ; P		N ; O		
Chaffangeon									
Chambe									
Gougeon									
MARI			I	J					
Parreaux									
Koskas									
Voorwinden					B ; N		N		
<ConseilLion>				H ; K	H				
bonhomme_ropers									
Beyou Nicolas Djombissi Willy									
gautraingodbillot				H ; K	H				

gautraingodbillot				H ; K	H				
heyeJego									
LotoutNabot									
<Connan_Melin>	Q				N		N		
brunoykeroullas				H	G ; H ; N		M ; N	F ; G	
labrueleyhuelic									
TP89BoureauTurmel					A ; B ; E ; F ; G ; L ; N ; O ; P		N ; O		
Amir_Pierre									
Amaury BERTHELOT & Erwan BOUET									
<Bikel_KEUMEKA_Mame_NGUEYE>		F	I	J	A ; O		O	E ; F ; G	
CHIQUET_ROCHAT									
PERRICHOT_REBOUX									
BONHOMME_FICHOUMEUNIER									
CharletPlantet									
ELGHARIB_COURTOIS					N ; P		N		
<oubahroche>					N ; P		N		
RiouTaja					N ; P		N		
Genet									
Ghorbal									
mleduc					N		N		

A :	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)					
B :	ack(1, 1, 1, 1)	pay(1, 1, 1, 1)					
C :	pay(1, 1, 1, 1)	ack(1, 1, 1, 2)					
D :	pay(1, 1, 1, 1)	ack(1, 1, 1, 2)	pay(1, 1, 1, 1)				
E :	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 2)				
F :	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 2)	ack(1, 1, 1, 2)	pay(1, 1, 1, 2)	ack(1, 1, 1, 2)	
G :	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 2)	ack(1, 1, 1, 2)			
H :	cancel(1, 1, 1)	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)		
I :	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 1)	cancel(1, 1, 1)			
J :	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 1)	cancel(1, 1, 1)	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 1)
K :	cancel(1, 1, 1)	cancel(1, 1, 1)	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)	pay(1, 1, 1, 1)		
L :	pay(1, 1, 1, 2)	ack(1, 1, 1, 1)	pay(1, 1, 1, 2)				
M :	pay(1, 1, 1, 2)	ack(1, 1, 1, 3)	pay(1, 1, 1, 1)	ack(1, 1, 1, 1)			
N :	pay(1, 1, 1, 1)	ack(1, 1, 1, 2)	ack(1, 1, 1, 3)	pay(1, 1, 1, 2)			
O :	pay(1, 1, 1, 2)	pay(1, 1, 1, 1)	ack(1, 1, 1, 2)				
P :	pay(1, 1, 1, 5)	ack(1, 1, 1, 10)	pay(1, 1, 1, 15)				
Q :	pay(1, 1, 1, 0)	ack(1, 1, 1, 0)	pay(1, 1, 1, 0)	ack(1, 1, 1, 0)			