UFC
UNIVERSITÉ
DE FRANCHE-COMTÉ

# Gestion distribuée d'une partie de poker

Sujet de TP-Projet 2014

Algorithmique distribuée - M2 Informatique

## 1 Présentation

Le but de ce TP-Projet est de réaliser un jeu de poker simple dans lequel nous ne gérerons que la partie jeu, pas la partie des mises. Les règles du jeux sont les suivantes: de 2 à 5 joueurs de réunissent pour jouer, le maître du jeu distribue cinq cartes (une par une, en cinq tours) à chacun des joueurs, après la donne chaque joueur peut changer tout ou partie de ses cartes, les joueurs ont le droit de faire au plus deux changements, une fois les changements terminés les joueurs exposent leurs cartes une par une pour savoir qui a gagné.

Le principe à implémenter est le suivant. Les joueurs ne se connaissent pas avant de jouer, ils se connectent donc en se découvrant mutuellement et en s'identifiant. Au début de la partie un des joueurs doit devenir le maître qui distribuera les cartes. Dans les phases de distribution et de changement le maître distribue les cartes une par une en cinq tours lors de la distribution puis l'ensemble des cartes échangées par le joueur dans la phase de changement. Lorsqu'ils sont d'accord sur la fin de la phase de changement les joueurs exposent leurs cartes.

## 2 Contraintes et indications

Voici les contraintes qui vous sont imposées pour la réalisation de ce jeu:

- Les joueurs sont implantés sous la forme de processus distincts qui ne peuvent communiquer que par échange de messages. Ils doivent pouvoir être exécutés sur des machines différentes. Chaque joueur possède un nom et il n'y a pas deux joueurs avec le même nom.
- Les joueurs communiquent uniquement à travers un processus `Reso`. Ce processus désynchronise les échanges de messages pour mettre en évidence les problèmes de concurrence. On supposera qu'il n'y a pas de perte de message. Le délais maximal de transmission d'un message est de 5 secondes. Attention la transmission des messages n'est pas FIFO.
- Au début les joueurs ne se connaissent pas et ils ne savent pas combien il y a de joueurs. Un joueur qui veut jouer se déclare au processus `Reso` puis il diffuse un message de jeu à travers la fonction `broadcastMessage()`. Après une minute d'attente il peut considérer que tous les joueurs qui souhaitent jouer se sont déclarés. Il joue avec les joueurs dont il a reçu un message de jeu.
- Au début les joueurs n'ont pas de numéro. Ils doivent en choisir un chacun, de manière distribuée (ça ne peut pas être un seul processus qui donne les numéros), qui ne repose pas sur leur nom et de manière à ce qu'il n'y ait pas deux processus avec le même numéro.
- Une fois que tous les joueurs ont un numéro ils élisent un maître du jeu qui distribue les cartes une par une dans l'ordre croissant des numéros.
- Tout au long du jeu le maître ne peut distribuer que les cartes une par une, c'est-à-dire qu'il doit émettre un message pour chacune des cartes qu'il distribue.
- Quand un joueur a reçu toutes ses cartes de la donne, il choisit celles qu'il veut changer. Les cartes doivent être échangées en une seule fois avec le maître, sans interférence avec les autres joueurs, sachant que le maître ne donne toujours les cartes que une par une.
- Les joueurs se mettent ensuite d'accord pour être sûrs que chacun a fini ses échanges. Aucun joueur ne doit afficher ses cartes si un autre joueur est encore en train d'échanger des cartes.
- L'affichage des cartes se fait par round, une par une, joueur par joueur. Un joueur ne doit pasafficher une nouvelle carte alors que le round n'est pas fini.

## 3 Déroulement du projet

Le projet est composé de trois parties :

1. Pour numéroter les processus, vous devez concevoir un algorithme de numérotation qui attribue un numéro à chacun des processus. Attention cet algorithme doit pouvoir marcher quelque soit le nombre de processus qui sont connectés. On supposera que ce nombre est constant pendant le déroulement de l'algorithme et qu'il n'y a pas de panne ni perte de message. Vous devez donner la preuve du bon fonctionnement de votre algorithme.

2. Pour choisir le maître du jeu, vous utiliserez un des algorithmes d'élection vu en cours. Vous justifierez correctement le choix de cet algorithme.

3. Pour déterminer la fin de la phase de changement de carte des joueurs vous mettrez en place un algorithme de terminaison. Vous justifierez correctement le choix de cet algorithme.

Vous écrirez, en respectant les contraintes données précédemment, un programme joueur qui peut jouer au poker avec d'autres programmes joueurs lancés à partir de la même instance. L'ensemble du projet sera écrit en Java et les communications se feront sous la forme d'invocations RMI.

Chaque instance de votre programme devra générer un fichier de traces contenant les informations relatives aux émissions et réceptions de messages (TYPE et données contenues). Chaque événement devra être transcrit sur une ligne séparée et chaque paramètre de l'événement devra être séparé des autres paramètres par le caractère |.

Le projet sera réalisé en binôme. Une démonstration permettra de mettre en évidence le fonctionnement correct des algorithmes.

## 4 Compte-rendu

Un rapport décrivant les algorithmes mis en œuvre et leur déroulement au sein des joueurs sera rédigé. Vous insisterez sur la description des algorithmes, la justification des choix, l'explication des adaptations, l'implantation et la validation du fonctionnement des algorithmes choisis. Vous présenterez une trace d'exécution qui illustre le bon fonctionnement de chacun des algorithmes. Des schémas formels ou informels peuvent illustrer l'architecture de la plate-forme et les différentes interactions. Votre code sera documenté et respectera les conventions de code Java.