SIM202 : Simulation d'une place boursière

sujet proposé par Nicolas KIELBASIEWICZ : nicolas.kielbasiewicz@ensta-paris.fr

9 janvier 2023

Le but de ce projet est de simuler le comportement de la Bourse de Paris en réalisant, d'une part, un serveur simulant le comportement du marché boursier et d'autre part, des clients simulant des porteurs de portefeuilles d'actions. Dans ce qui suit ne sont énoncés que les principes généraux et quelques éléments sur les classes à utiliser. Bien évidemment, il vous appartient de préciser et de compléter ces classes, voire d'en ajouter de nouvelles si le besoin s'en fait sentir.

1 Principe de fonctionnement de la bourse

L'ensemble des hypothèses et des modélisations qui sont faites repose sur la description de la place de Paris (voir [1]). On distingue deux types d'intermédiaires pouvant intervenir sur la place : les établissements de crédit et les entreprises d'investissement ; ils collectent les ordres des clients, ils exécutent les ordres et la gestion des portefeuilles. Il existe par ailleurs des établissements assurant les opérations de règlements et de livraison dans les délais règlementaires (les compensateurs). Le marché est segmenté en trois compartiments : le premier marché composé des plus grandes sociétés, le second marché ayant vocation à accueillir des entreprises de taille moyenne avant leur admission au premier marché et le nouveau marché composé d'entreprises ayant un fort potentiel de croissance (start-up). Sur tous ces marchés, le règlement se fait au comptant. Un ordre de bourse est soit un ordre de vente ou d'achat et doit comporter des indications sur la durée de validité et sur le prix d'exécution. Les agents peuvent fixer une date limite de validité de l'ordre. En l'absence de date limite, les ordres sont dits « à révocation ». Les ordres peuvent être donnés « à cours limité » : le client fixe un prix maximum dans le cas d'un achat et un prix minimum dans le cas d'une vente. Ils peuvent également être donnés « à tout prix » ; dans ce cas, ils sont prioritaires sur tous les ordres. Ils existent d'autres types d'ordre sur la place de Paris qui ne seront pas modélisés ici.

L'établissement des cours se fait de manière automatique et de la façon suivante :

— pour chaque valeur (action), les ordres sont classés par limite de prix et chronologiquement par ordre d'arrivée pour des ordres de même prix

— l'exécution de l'ordre s'effectue selon deux règles de priorité :

par le prix : sont servis prioritairement les ordres d'achat ayant une limite maximale et les ordres de vente ayant une limite minimale

par le temps : deux ordres de même sens et à même limite sont exécutés dans leur ordre d'arrivée

La séance de Bourse se déroule en plusieurs étapes :

1. La préouverture de 8 h 30 à 10 h 00 : les ordres s'accumulent sans aucune transaction
2. L'ouverture à 10 h 00 : le système calcule, en fonction des ordres présents, un prix d'équilibre (le fixing) qui permet l'échange du plus grand nombre de titres
3. La séance de 10 h 00 à 17 h 00 : le marché fonctionne en continu et l'introduction d'un ordre provoque immédiatement une ou plusieurs transactions dès lors qu'il existe un ou plusieurs ordres en sens contraire. Le cours d'exécution est celui de la limite de l'ordre en contrepartie.
4. La pré-clôture de 17 h 00 à 17 h 05 : comme en préouverture, les ordres s'accumulent.
5. La clôture à 17 h 05 : un prix d'équilibre est déterminé comme à l'ouverture fixant le dernier cours.

1.1 Calcul du cours de la bourse

On ne décrit ici que la méthode de calcul des cours pour le premier marché.

1.1.1 Cours de préouverture et de pré-fermeture (fixing)

Pour une action donnée on établit une liste des offres d'achat par ordre de prix décroissant et chronologique (un achat à tout prix se trouve en haut de la liste) ainsi qu'une liste des offres de ventes par ordre de prix croissant (une vente *à tout prix* se trouve en haut de la liste). On met alors en corrélation les ordres d'achats et de vente et on exécute tous les ordres possibles en partant du début de la liste. Le dernier prix de réalisation fixe le cours de l'action. Illustrons ce calcul à l'aide d'un exemple.

Offre d'achat		Offre de vente	
Quantité	Prix	Prix	Quantité
400	*à tout prix*	*à tout prix*	400
200	156	150	250
250	155	151	400
500	154	152	500
850	153	153	600
1000	152	154	1250
3000	151	155	1700

Voici le déroulement des transactions :

1. Les 400 actions vendues *à tout prix* seront achetées par l'acheteur proposant *à tout prix*. Le prix est en l'occurrence le cours précédent de l'action, à savoir 150.
2. Les 250 actions vendues 150 seront achetées par l'acheteur à 156 pour 200 actions et l'acheteur à 155 pour 50 actions
3. Les 400 actions à 151 seront achetées par l'acheteur à 155 pour 200 actions et l'acheteur à 154 pour 200 actions
4. Les 500 actions à 152 seront achetées par l'acheteur à 154 pour 300 actions et l'acheteur à 153 pour 200 actions
5. Les 600 actions à 153 seront achetées par l'acheteur à 153 pour 600 actions

L'acheteur est privilégié par il paie ses actions au prix de vente, moins cher que le prix d'achat qu'il propose.

Dans cet exemple, donc, 2150 titres sont achetés : 400 à tout prix, 200 à 156, 250 à 155, 500 à 154 et 800 à 153 et 2150 titres sont vendus : 400 à tout prix, 250 à 150, 400 à 151, 500 à 152 et 600 à 153. Aucun autre échange n'est possible. Le cours de l'action est fixé à 153 (dernière valeur de transaction). À l'issue du fixing, le marché de l'action sera alors le suivant :

Offre d'achat		Offre de vente	
Quantité	Prix	Prix	Quantité
50	153	*à tout prix*	400
1000	152	154	1250
3000	151	155	1700

1.1.2 Cours en continu

Lors de l'arrivée d'un nouvel ordre, on examine si cet ordre peut être réalisé, à savoir s'il existe une contrepartie réalisable sur le marché. Illustrons ce mécanisme sur un exemple. Supposons que le marché soit dans l'état suivant pour une action donnée :

Offre d'achat		Offre de vente	
Quantité	Prix	Prix	Quantité
500	149	150	1000
2000	148	151	1500
1500	147	152	2500
1000	146	153	1000

Dans cet état, aucune transaction n'est possible et le cours est de 149 (dernière valeur de transaction). Supposons qu'arrive une offre d'achat de 4000 titres à 152. L'ordre est confronté aux ordres en attente et est intégralement exécuté à 150 pour 1000 titres, à 151 pour 1500 titres et 152 pour 1500 titres. À l'issue de la transaction, le marché de l'action est le suivant :

Offre d'achat		Offre de vente	
Quantité	Prix	Prix	Quantité
500	149	152	2500
2000	148	153	1000
1500	147		
1000	146		

et le nouveau cours est 152.

On notera que dans tous les cas, le cours est la valeur de la dernière transaction effectuée.

2 Système client-serveur

Afin de simuler la place boursière, on réalisera un système client-serveur. Le serveur doit assurer le traitement des ordres passés par les clients sous forme de messages : réceptionner les messages, renvoyer un accusé de réception, établir le fixing et les cours, réaliser les transactions, émettre un message au client lorsqu'une de ses transactions a été réalisée, gérer un historique des transactions, publier les cours en temps réels …En outre, le serveur aura la charge de la gestion du temps et de comptes clients afin que ces derniers ne lancent pas des ordres illimités. Le client devra proposer deux modes de gestion de son portefeuille : un mode manuel (intervention manuelle sur le marché) et un mode automatique générant de façon régulière des ordres. Il devra gérer son portefeuille en mémorisant l'historique des ordres, des transactions, afficher les cours et ses positions actuelles.

Le client et le serveur communiquent par message. On propose le modèle suivant, dans lequel un message client est une chaine de caractères devant contenir :

— un identifiant client unique
— un numéro de message unique du client
— une liste des ordres codés par exemple #A/V,action,quantité,prix,date

On utilise un message particulier lors de la première connexion au serveur demandant de renvoyer un identifiant client, le montant de son compte ainsi que la date et l'heure du serveur. Le serveur doit gérer un fichier contenant le nom des clients, leur identifiant ainsi que la position de leur compte.

Un accusé de réception du serveur est une chaine de caractère contenant l'identifiant client, le numéro de message et la réponse (accepté ou refusé). Le serveur émet également des messages prévenant un client qu'une transaction a été réalisée pour son compte. Un tel message contiendra :

— l'identifiant client
— le numéro de message unique
— la liste des ordres codés par exemple #A/V,action,quantité,prix,date

En outre, lorsque le cours d'une action est modifiée, le serveur remettra à jour un fichier des cours exploitable en mode lecture par tous les clients.

2.1 Transfert de messages

Il existe plusieurs systèmes pour passer des messages entre applications (socket, sémaphore, …). Si vous avez déjà vu la programmation par socket, vous pouvez l'utiliser. Sinon on peut utiliser des transferts de message par fichier verrouillé créé dans un dossier commun appelé par exemple *messages* ouvert en lecture et écriture. Décrivons plus précisément un tel mécanisme.

Supposons qu'un client veuille émettre un message :

1. Il crée tout d'abord un fichier de nom CLI_nomclient_nummessage.mes.lock
2. Il crée un fichier de nom CLI_nomclient_nummessage.mes contenant son message.
3. Une fois le fichier *.mes* fermé, il détruit le fichier *.mes.lock*. Ainsi, tant que le fichier *.mes.lock* existe, le message n'est pas exploitable.
4. De son côté, le serveur explore en permanence le dossier *messages* en détectant dans l'ordre de priorité les fichiers *.mes*. De deux choses l'une, soit il existe un fichier *.mes.lock* auquel cas on passe au suivant, soit il n'existe pas et le serveur lit l'information du fichier puis le détruit ou l'archive ailleurs. Les fichiers d'accusé de réception envoyés par le serveur à un message client s'appelleront SER_nomclient_nummessage.acr.
5. Après l'émission d'un message, le client doit donc attendre que le fichier *.acr* soit accessible pour valider son offre. Les fichiers contenant les transactions effectuées par le serveur pour un client donné se nommeront SER_nomclient_nummessage.mes
6. Le client devra régulièrement explorer le dossier messages pour mettre à jour les transactions qui le concernent. Dans tous les cas, l'écriture d'un fichier passe par la création d'un fichier *.mes.lock* associé.

Le fichier des cours de bourses mis à jour par le serveur, placé également dans le dossier *messages*, est géré de la même façon. Noter que dans ce dernier cas il y a une difficulté supplémentaire, car le serveur ne peut pas savoir si le fichier des cours de bourse n'est pas exploité par un client au moment où il souhaite le mettre à jour. Proposer une solution ?

Au vu des opérations que l'on doit effectuer sur les fichiers, on a tout intérêt à écrire des routines générales pouvant servir aussi bien au serveur qu'au client.

3 Travail à effectuer

On réalisera un serveur et un client basés sur les principes précédemment décrits. La programmation devra être réalisée en C++ en introduisant des classes d'objet de message, de marché d'une action (liste des offres et des demandes), de portefeuille pour le client, …(passer un peu de temps à réfléchir à vos classes). Comme on l'aura noté, afin de réaliser les opérations, il est nécessaire de disposer de structures possédant des capacités de tri. Dans cette optique, il est conseillé d'utiliser les classes map ou multimap de la STL qui possèdent de telles aptitudes. Dans un second temps, on essaiera de construire des modèles simples simulant l'activité soutenue d'un client afin de disposer de clients automatiques.

3.1 Organisation du travail

Afin de mener le développement de façon efficace, je propose l'organisation du travail suivante :

— Travail collaboratif pour définir une classe **Message** encapsulant tous les types de message envisagés (éventuellement dans un cadre d'héritage). Il est impératif que cette classe soit définie rapidement, car elle constitue le point commun du client et du serveur.

— Une fois ce travail réalisé, un élève se charge du développement de la classe **Message** qui doit être rapidement fonctionnelle à quelques détails près. Un autre s'attaque à la conception du serveur et plus précisément au système qui permet d'opérer les achats et ventes basés sur des listes d'offres d'achat et de ventes. Le dernier s'attèle au développement d'un client manuel et automatique dans un deuxième temps.

— Une fois la classe **Message** opérationnelle, celui qui l'a développée pourra rejoindre celui qui développe le serveur, afin de réaliser un système d'horloge pour séquencer la journée (1 s = 1 mn par exemple), les procédures d'initialisation (liste des clients, compte des clients) et toutes les fonctionnalités utiles (historique des ventes, des achats, des cours, …).

Lors de la conception, vous avez intérêt à fabriquer un schéma fonctionnel de votre système : classes + fonctionnalités et interactions entre elles.

3.2 Extensions possibles

Développer une interface client graphique à l'aide de bibliothèques graphiques permettant à un client de visualiser les cours, son panier d'action et de passer des ordres.

Références

[1] Bourse de Paris. *La Bourse de Paris*. SBF.