Panic on Wall Street

Alexandre da Silva Lima and Daniel Fernandes Gomes

Agents and Distributed Artificial Intelligence
Faculty of Engineering, University of Porto
Rua Dr Roberto Frias S/N, 4200-465 Porto, Portugal
{up200503841, up201306839}@fe.up.pt

1 Introduction

With the constant communication networks growth, the motivation to solve spatially distributed problems through multi-agent system architectures increases. Such agents must be capable of perceive the environment in which they are inserted and cooperate with others in order to accomplish the desired tasks. Because of that, it is important to study both their modeling as well the communication protocols that enable their communication.

One example of such scenario is the stock market, where abroad individuals invest on companies through brokers. Here often, investors compete among themselves in order to get the best prices while brokers want to sell the most. Such dynamics are captured by the board game Panic on Wall Street where players embrace the role of managers (brokers) and investors attempting.

In this project this board game is considered for the study of a multi-agent system. Its implementation was made using the Jadex framework and two levels of intelligence were developed: stochastic and expected value based – specified in 2.3. Simulations were run to compare their performances and the results are exposed in 4. The final conclusions are presented in 5.

2 Specification

2.1 Game rules

There are two types of players: managers that own companies and investors that invest on them. Each company is associated to one of the four market fluctuations (yellow, red, blue and green). These fluctuations consist on a list of ordered values from negative to positive ones. At each point in time, each fluctuation assumes one of its values. The companies different range of values make a company more ore less risky to invest on. The game begins with each player receiving 120K$ and Managers receiving also 2 companies. Each fluctuation starts at its middle position: 30K. Then a month (game round) happens:

1. Investors try to invest, openly, on many companies as they desire by negotiation with each respective manager the investment value. A company has only, at most, one investor. Each manager or investor is free to drop the deal when desired, unless previously both accept to close it.

2. Ended the negotiation phase, the market fluctuations are updated. For this purpose, each respective dice is rolled. These have both positive and negative values, indicating the number of positions the fluctuation moves relatively to the previous state.
3. Managers and investors collect their profits and pay their expenses according in the following order:
   (a) For each company that an investor invested on, receives its market value.
   (b) Then, pays to the correspondent manager the agreed investment.
   (c) Managers pay for each of their companies the management costs to the market: 10K$.
   (d) If any manager reaches a negative balance it may sell many of its companies as necessary back to the market for 5k$ each.
4. At the end of each round, $2 * |managers| - 1$ new companies are auctioned between the managers.

Having been fulfilled 5 rounds, the manager and investor with most money are the two winners of the game.

2.2 Ambiguous scenarios

The rules of the board game do not specify all the details on two scenarios: what to do with managers or investors that reach a negative balance and how the new companies auction is conducted. For the negative balance case, the game rules mention the possibility of the affected reach some friendly deal that allow the player in debt to keep in game. Yet, given the low probability of this scenario and for simplicity reasons, in our implementation the agent simply leaves the game.

Regards the new companies auction, it was adopted the simultaneous ascending style: each player submits one bid, no isolated highest bid appears a new round is performed with the highest bidders. When only one bidder remains, that's the winner.

2.3 The Agents

When modeling this game into a multi-agent system the two types of players were mapped into two agents: Investor and Manager. Also, in order to make the information synchronization simpler and more efficient a third agent was introduced: WallStreet. For each of the two player agents two levels of strategy were implemented.

1. **Wall Street**: is a simple agent, mediates the game phases, the bank and money transactions, stock fluctuations and company auctions. It does not have self goals and acts as an object.
2. **Investor**: invests in companies with the goal of maximizing its balance competing against the other investors. If it reaches a negative balance it leaves the game. It knows about all the negotiations between Managers and other

Investors and decides on which companies it should invest, what value to invest and if and when should close a deal. It may also decide to drop a deal. The two strategies are:

(a) **Dummy**: attempts to make random investments with random values between 5K$ and 35K$ range on random companies, requesting to close it half of the times. It never drops a deal.

(b) **Intelligent**: makes investments based on the current state of the market and the expected revenue **(E)** at the end of the round:

$$E(c) = \sum_{i=1}^{N} p(i) * val(CC, cp(CC) + dice(CC, i)) \tag{1}$$

where **N** is 6, the number of faces of the dice, **p(i)** the probability of that face being drawn (1/6), **CC** the company color, **CP** the current position of the fluctuation, and **val** the fluctuation value.

The agent sorts the companies by the decreasing opportunity **(O)** value, defined as:

$$O(c) = E(c) - currentOffer(c) \tag{2}$$

and with a 20% skip probability picks the first with opportunity greater than zero. It then purposes an offer 5K$ higher than the current. Half of the times requests closing the deal.

Because at the end of each round the investments are kept, it is possible that, given the market changes, one deal stop being profitable. The agent solves this by dropping the ones in j which the opportunity is lower than zero.

3. **Manager**: manages companies with the goal of maximizing its profits. It participates on company actions to increase its assets and competes against the other managers. It has a role on the company investments negotiation by deciding when accept an offer and its closing. Then participates on the new companies auctioning by deciding when to bid and what value. The two strategies implemented are:

(a) **Dummy**: accepts any investment greater than the current one and if closing is requested accepts it 30% of the times. On the new companies auction phase, it increases the current bid 60% of the times between 1K$ and 25K$.

(b) **Intelligent**: Rejects offer with a probability depending on the difference between the offered value and the

$$rejectP(c, offer) = \frac{E(c) - offer}{E(c)} \tag{3}$$

If closing is requested then it rejects it with 50% chance. When bidding on new companies, it always increases the current bid between 1K$ and 30K$ until the total remaining expected value.

$$TotalRemainingE(c) = E(c) * nRemainingRounds \tag{4}$$

on auctions based on the company perceived value(current and possible future states)

2.4 Interaction protocols

In order to implement the communication the Jadex services mechanism was used. At wake up, all agents contact WallStreet that before the game begins, introduce players to each other.

1. **Negotiation**
   (a) WallStreet informs all players that the negotiation phase opened and companies current value.
   (b) Investor sends bid on company to manager and if it desires to close it.
   (c) Manager decides if it wants to accept it, and if so contacts WallStreet in order to validate the offer else, rejects it.
   (d) WallStreet confirms/rejects offer.
   (e) Manager confirms/reject offer.
   (f) Manager informs all the remaining players of the new company status
   (g) WallStreet informs all players that the negotiation phase closed

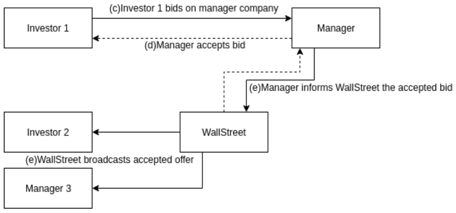

(c)Investor 1 bids on manager company
Investor 1
Manager
(d)Manager accepts bid
(e)Manager informs WallStreet the accepted bid
Investor 2
WallStreet
(e)WallStreet broadcasts accepted offer
Manager 3

Fig. 1. Illustration of an offer communication chain.

2. **Exchange of incomes:** WallStreet updates the market, calculates the company new values, investors gains/losses, transfers payments from investors to managers and collects management costs.
   (a) WallStreet informs all players balance, its company values and market status.
   (b) WallStreet queries each manager for what companies it desires to sell back.
   (c) Managers answer WallStreet.
   (d) WallStreet informs all players balance, its company values and market status.
3. **Company auction**: (2 x |Managers| - 1) new companies are auctioned

(a) WallStreet informs each (in auction) Manager the new company current value
(b) Managers respond with bid to WallStreet
(c) WallStreet informs all players balance, its company values and market status.

3 Development

3.1 Tools

For this assignment Jadex, a library for the implementation of rational agents in Java, was used. Its latest engine (BDIv3) allows the creation of agents following the BDI – beliefs, desires and intentions – architecture renamed in Jadex as beliefs, goals and plans.

Beliefs are agent's attributes that, on its change, activate certain goals or plans. When a goal is activated, its plans are executed until it is considered achieved or there are no more possible plans to try. Plans are simply programmed recipes that contribute to the accomplishment of the associated goal.

Besides the reasoning engine, Jadex provides communication functionalists that allow for the implementation of the messages exchange between agents in an asynchronous service oriented manner.

To plot players balance and companies value evolution along the game JFreeChart library which offers a large variety of charts compatible with Java Swing — the framework used for the user interface implementation.

For the development the Eclipse Java IDE; and for the dependencies management, the Gradle automation system were used.

3.2 Architecture

In order to implement the above agent communication the Jadex services capabilities were used, referred in 7.2. To implement agents internal 3 goals that are activated when the belief game state changes to negotiation:

1. **InvestOnCompanies:** (on investors) this goal is kept until the end of the phase and and executes its only plan "invest" which follows the specification explained in 2.3
2. **ProcessOffers:** (on managers) because manager needs to validate offers, it must hold all the offers to the same company until it receives the confirmation from the wall street.
3. **DropBadOffers:** (on intelligent investors) used to accomplish what mentioned in 2.3 2. (b).

4 Experiments

4.1 Intelligent Investor vs the dummies

Objective The objective of this experiment was to test whether or not intelligent investor was really more intelligent than the dummy one. We ran 15 3vs4 games with one intelligent investor and the rest of the agents are dummies.

Expectations We expected the intelligent investor to win most games. Due to the random nature of the game its almost impossible to win 100% of the time. Because intelligent investors invest on companies considering the opportunity (including the current value) and the maximum expected value, it expected that these agents achieve lower deals and have on average higher profits at the end of a round.

Results An intelligent manager won 13 out of 15 games where won by the intelligent investor(86.7%).

4.2 Intelligent Manager vs the dummies

Objective The objective of this experiment was to test whether or not the the previous obtained results would maintain when intelligent managers and investors play together.

Results the intelligent manager won only 6 out of 15 games (40%).

4.3 Intelligent Agents vs dummies

Objective The objective of this experiment was to test whether or not the intelligent manager was really more intelligent than the dummy one. We also wanted to see if playing with intelligent investors benefited the intelligent managers. We ran 15 games with a different number of intelligent and dummy managers and always with intelligent investors.

Results Out of 10 games, 7 where (70%) won by the intelligent investor while only 3 (30%) by the intelligent manager.

5 Conclusions

With the experiments we ran, it is possible to conclude that the intelligent agents use are somewhat better than the stochastic ones, specially the investors. Analyzing the managers performance, it is noticeable that often it exaggeratedly bids on the first rounds, not being able to achieve enough profit from those investments on the latter game. Although the success of the investors it is arguable that a different calibration of its parameters would favor the stochastic ones.

5.1 Improvements

There are some ways we could improve our program.

1. Implementing agents that react to the others actions. For example, if a winning manager bids on a company the losing one should try to prevent the winning one from getting that company.
2. In order to improve managers performance, instead of consider the expected value the market achieved value could be used.
3. Implementing agreements between investors and managers that benefit both.
4. Making better use of Jadex beliefs, plans and goals.

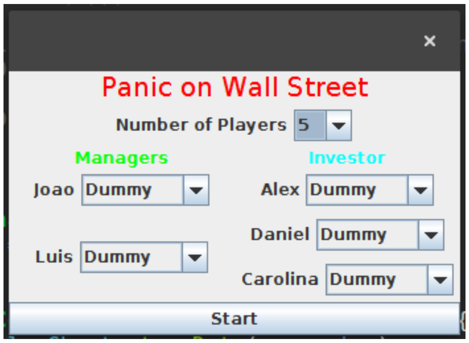
6 Resources

1. Used Resources:
   (a) Jadex https://www.activecomponents.org/
   (b) Jadek FEUP wiki https://paginas.fe.up.pt/~eol/AIAD/jadex/
   (c) JFreeChart www.jfree.org/jfreechart/
2. Effective work Alexandre: 50%
   Daniel: 50%

7 Appendix

7.1 User guide

The developed application allows for the simulation of the Panic on Wall Street game using the agent's implementation described before in this report.



Panic on Wall Street

Number of Players 5

Joao Dummy

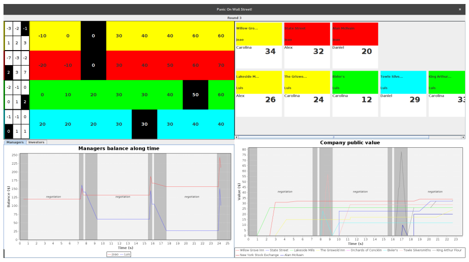
Luis Dummy

Alex Dummy

Daniel Dummy

Carolina Dummy

The first window allows the configure the game to be disputed: number of players, according with game rules, and level that each runs on (dummy or intelligent).



Panic On Wall Street!

Round 3

-3	-2	-1	1	2	3	-10	0	0	30	40	40	60	60
-7	-3	-2	2	3	7	-20	-10	0	30	40	50	60	70
-2	-1	0	0	1	2	0	10	20	30	30	40	50	60
-1	-1	0	0	1	1	20	20	20	30	30	30	40	40

Willow Gro...
Joao
Carolina 34

State Street
Joao
Alex 32

Alan McIlvain
Joao
Daniel 20

Lakeside M...
Luis
Alex 26

The Griswo...
Luis
Carolina 24

Bixler's
Luis
Carolina 12

Towle Silve...
Luis
Daniel 29

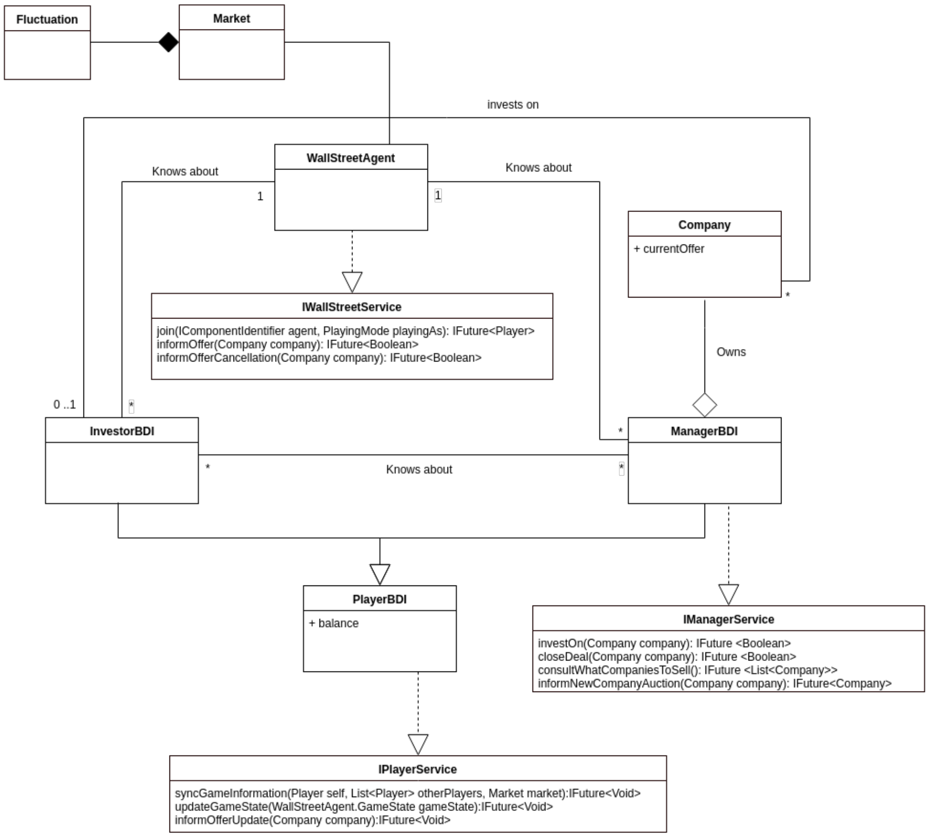
King Arthur...
Luis
Carolina 33

The second, the game window, exhibits all game relevant information:

1. At the top left, the market. Each row represents one fluctuation. On the left, in white, the associated dice and in black the last drawn face. The rest are the possible values of the fluctuation and in black its current value.
2. At the top right, the company cards grouped, in rows, by its owner. On the top half of the card it is possible to see its associated fluctuation color, name and owner name. On the bottom half, the current investor, the deal value and if it is closed.
3. At the bottom left, under the two tabs, it is possible to follow the evolution of the managers and investors balance. The time axis is segmented using vertical stripes into the four main moments of each round: negotiation, exchange of incomes, solving of managers debts and finally new companies auction.
4. At the bottom right, companies value is shown. On the negotiation phase it represents the value offered by the investors while on the new companies auction phase it represents the current bid.

7.2 UML Class diagram

The UML diagram below expresses the overall architecture. PlayerBDI is an abstract agent that groups all the code and services that both investors and agents must implement.


Fluctuation
Market
invests on
WallStreetAgent
Knows about
Knows about
1
1
Company
+ currentOffer
*
IWallStreetService
join(IComponentIdentifier agent, PlayingMode playingAs): IFuture<Player>
informOffer(Company company): IFuture<Boolean>
informOfferCancellation(Company company): IFuture<Boolean>
Owns
0 ..1
*
InvestorBDI
*
ManagerBDI
*
Knows about
*
PlayerBDI
+ balance
IManagerService
investOn(Company company): IFuture <Boolean>
closeDeal(Company company): IFuture <Boolean>
consultWhatCompaniesToSell(): IFuture <List<Company>>
informNewCompanyAuction(Company company): IFuture<Company>
IPlayerService
syncGameInformation(Player self, List<Player> otherPlayers, Market market):IFuture<Void>
updateGameState(WallStreetAgent.GameState gameState):IFuture<Void>
informOfferUpdate(Company company):IFuture<Void>