Matching Rules and Market Share in an Electronic Trading Platform

by

Yongliang Wang

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In this thesis we study the problem of how to effectively manage and operate a market that attracts trading agents to compete for resources in it. In order to attract more agents to the market, the market needs to have incentive policies. We are particularly interested in the research of the incentive matching policy. We propose a new matching policy with loyalty incentive features. In order to cooperate and improve its performance, we also propose a new accepting policy to work with the matching policy. We use the CAT platform as our test-bed. We describe all the policies and techniques used in the CAT competition in detail. In addition we carry out experiments which further support our proposal.

Acknowledgments

First I would like to thank my supervisor, Kate Larson, for all the effort and help she has dedicated to my research. She has provided me full support, as well as flexibility and freedom during my graduate study. She demonstrated to me a model of top research.

I would also like to thank the people on my thesis committee, Robin Cohen and Chrysanne Di Marco, thank you for your time and valuable feedback on my thesis.

Last but not least, I would like to thank Rongdong Chen, for all the proof-reading.

Dedication

To my parents, for your love and faith in me. To Flora Guan, for your love and support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Continuous Double Auction (CDA) [6] is a mechanism to match buyers and sellers of a particular good, and to determine the prices at which trades are executed. The beauty of CDA is that, at any point of time, any traders (buyers or sellers) can enter into the market and submit offers. A trade would take place instantly if there is a matching offer in the market. CDA has been used in stock markets for trading securities and other financial commodities.

Most of the existing work and research on CDA addresses ways of designing effective strategies to maximize the trader's profit. For instance, in the classic Trading-Agent Competition (TAC Classic), entrants were motivated by the desire to develop automated strategies for buyer and seller software agents to achieve optimum profits in the virtual market place. The trading rules or interaction mechanisms are fixed by the TAC Classic organizers, and competition entrants play against one another by creating agents to trade in accordance with these predefined rules. However, in today's globalized market environment, stocks are often traded simultaneously in different markets around the world. Meanwhile, markets need to differentiate themselves and improve their value propositions to attract more active and profitable traders to become their members. As such, there is growing demand for research in this area to expand its focus to include design strategies from a market's perspective. Our study is a valuable step toward this research direction.

The TAC Market Design Competition (CAT) [15] provides a test-bed for exploring the problem of designing competitive and efficient markets. CAT is the exact reverse of TAC Classic: the software trading agents are created by the organizers, while the game participants compete by defining rules for matching buyers and sellers and setting commission fees for providing relevant services. Entrants compete against each other in attracting buyers and sellers, as well as generating maximum profits from completed trading. This can be achieved by having effective matching rules and setting appropriate fees that strikes a strategic trade-off between making profits and attracting traders.

We are particularly interested in the trading agents' reactions when they face a variety of matching policies[1]. How to attract buyers and sellers by setting up different matching policies is the focus of the research. CAT served as an ideal test-bed for our proposed matching policies, because the agents' reactions can be measured directly by the performance of market share[2]. In other words, constructing a matching policy that can maximize market share is the main objective of the thesis.

[1]The matching policy generates matching pairs between buyers and sellers.

[2]Market share is the proportion of traders registered with one specific market.

Besides the matching policy, we are also interested in how the accepting policy[3] could cooperate with the matching policy to generate better market performance. The motivation of this research direction is that the accepting policy and the matching policy work as a combination in most cases. Variances in the combination of accepting and matching policies represent unique features of each market and thus have direct impact on market performance.

We based our empirical testing for this research in the CAT game. As all our suggested new approaches and algorithms were tested under the CAT game, we will discuss the details of CAT specifications in the ensuing chapter.

1.1 Contributions

The key contributions of this thesis are as follows:

WaterCAT Accepting Policy: We present an accepting policy that can determine which shouts[4] are accepted according to the market condition. Unlike the traditional fixed policy [15], our proposed policy can dynamically adjust the accepting level according to market activities in a real-time fashion. Comparing with the classic equilibrium-beating accepting policy [15], our policy accepts a wider range of agents to enter the market. However, even with the wider range, the policy can still maintain a high Transaction Success Rate[5].

[3] The accepting policy determines which bids or asks are accepted into the market.

[4] Shouts are the general term asks and bids.

[5] TSR is the proportion of bids and asks placed with that specialist which that specialist is able to match.

WaterCAT Matching Policy: We propose a new matching policy based on price ranking and loyalty ranking. We design a new method, Bonus Factor, that aggregates the price ranking and loyalty ranking to compose a final ranking, with which all the matching pairs will be generated. We also show that the new policy outperforms the base model equilibrium matching policy in our empirical setting.

Analysis on Different Charging Fees: We analyze the trading agent's sensitivity and reaction against different charging fees in fixed charging policy. During the test we found a winning setting for charging fees that can maximize the specialist's profits. In addition to the fixed charging policy, we propose several adaptive charging policies that are still being developed as prototypes.

Applications on other CDAs: Our research can also help the design of other CDAs competitions. Our approaches are generic and are applicable to other models. With the demonstrated performance, other researchers can adapt our approaches to their models.

1.2 Guide to the Thesis

In this section we outline the chapters for the rest of the thesis:

Chapter 2 - CAT Test-bed Specifications: In this chapter we discuss the detailed specifications and requirements of the CAT game.We also provide the background information of trading agent strategies as used in this thesis. The four trading strategies (ZI, ZIP, GD and RE) used by CAT are also explained in detail.

Chapter 3 - Policy Designs for CAT Specialists: In this chapter we review all the current policies designed for the CAT game. We then expound our modifications to the existing policies with particular focus on the accepting and the matching policies. Toward the end of the chapter, we also analyze the impacts of charging fees on the trading agents.

Chapter 4 - Post-game Analysis: We begin this chapter by describing the competition setup of the CAT tournament. We then decompose the winning reasons of our trial game and what caused the performance issues in the official game. By reporting the game re-run results with the post-tournament agents, we conclude this chapter by evaluating the performance of updated policies.

Chapter 5 - Related Research: In this chapter we discuss other relevant research in this area. We list two interesting research directions for the CAT competition.

Chapter 6 - Conclusion: In this chapter we conclude our work with a review of our contributions and a discussion of future directions.

Chapter 2

CAT Test-bed Specifications

This chapter provides an overview of the specifications and requirements of the CAT game. The general concept and overview of the CAT game are provided first. We then describe the concept of the trading agent, the requirements of the specialist, and the assessment criteria of the game. Besides the CAT game setup, the bidding strategies used by the trading agents in the game are also explained in this chapter.

2.1 The CAT Game Overview

The CAT game consists of trading agents, i.e., buyers, sellers and specialists. Each specialist operates and sets the rules for a single exchange market, and traders buy and sell goods in one of the available markets. In the CAT competition the trading agents are provided by the CAT game, whereas specialists (and the rules of the markets) are designed by the entrants. Each entrant is limited to operate a single market.

A typical CAT game consists of a CAT server and several CAT clients, which may be traders or specialists. CAT clients do not talk to each other; instead they connect to the CAT server through sockets and the server responds to messages from clients and forwards information if needed.

A CAT game lasts a certain number of days, each day consists of rounds, and each round lasts a certain number of milliseconds. Trading is only permitted during rounds, and hence during a day. After a day closes, information on the profit made by each specialist and the number of traders registered with it are disclosed. This allows specialists to change their market rules, adapting these rules to improve their competitiveness. Between days traders may change the specialist that they trade with, and they can migrate to specialists that allow more profitable trades [16].

2.1.1 Trading Agent

Each trading agent is endowed with a trading strategy and a market selection strategy, and is assigned private values for the goods being traded. The traded goods are assumed to be homogenous and non-divisible. Furthermore, each buyer and seller has a certain demand and supply. Private values and the demand and supply of the markets are allocated by the CAT server, and are reset at the start of each day. Although private values remain constant during a day, these values may change from day to day.

The trading strategy is used to generate bids and asks (both also called shouts), whereas the market selection strategy is used to select a market or specialist. Each trader is furthermore endowed with a limited budget that they can spend within a trading day. This budget prevents a trader from paying excessively high fees. Budget sizes are unknown to the specialist.

2.1.2 Specialist

Specialists facilitate trade by matching bids and asks, and determining the trading price in an exchange market. Each specialist operates its own exchange market. Each entrant in the game is required to design a single specialist or market, which is achieved by implementing the following policies:

- *Charging Policy.* This policy sets the fees which are charged to traders and other specialists who wish to use the services provided by the specialist. Each specialist is free to set the level of the charges. These are the following:
  - *Registration fees.* Fees charged for registering with a specialist.
  - *Information fees.* Fees for receiving market information from a specialist.
  - *Shout fees.* Fees for successfully placing bids and asks.
  - *Transaction fees.* A flat charge for each successful transaction.
  - *Profit fees.* A share of the profit made by traders, where a trader's profit is calculated as the difference between the shout and transaction price.
- *Accepting Policy.* This policy determines which shouts are accepted. A specialist has the option to reject shouts which do not conform to the specialist's

policy. For example, a "beat the quote" policy requires a shout to beat the market quote. If the received shout violates this, it can be immediately rejected and will not be considered for a transaction, allowing the trader to submit a new shout. The policy may reject any shout that is unlikely to trade so as to increase the specialist's transaction success rate, which is one of the assessment criteria for CAT game.

- *Clearing Policy.* This policy determines the way in which bids and asks are matched. For example, the specialist can sort the accepted bids and asks in order and select one from each group by applying pre-designed algorithms
- *Pricing Policy.* This policy determines the transaction price of a matched bid and ask. The most common is to set the price half way between the bid and ask.

2.1.3 Assessment

The entrants are assessed on multiple criteria, which will be evaluated on a number of trading days. In order to avoid effects arising from the fact that the tournament has a start-day and end-day, not all the trading days will be used for assessment purposes.

Each specialist is assessed on three criteria on each assessment day, and these criteria are then combined into a single score for that day. These three criteria are as follows:

- Profits: The profit score of a specialist on a particular day is given by the total profits obtained by that specialist on that day as a proportion of the total profits obtained by all specialists on that same day.

- Market Share: Of those traders who have registered with a specialist on a particular day, the market share score of a specialist on that day is the proportion of traders that have registered with that specialist on that day.
- Transaction Success Rate: The transaction success rate score (TSR) for a specialist on a given day is the proportion of bids and asks placed with that specialist on that day which that specialist is able to match.

Each of these three criteria results in a value for each specialist for each day. The three criteria are then weighted equally and added together to produce a combined score for each specialist for each assessment day. Scores are then summed across all assessment days to produce a final game score for each specialist. The specialist with the highest final score is declared the winner of the tournament.

2.2 The Bidding Strategies of the CAT Game

In the CAT competition each trader is equipped with both a bidding strategy for generating bids and asks, and a specialist selection strategy for selecting specialists each day. The specific details of the learning component of specialist selection, and the occasions on which it is applied, are private information to the traders, and are not provided to specialists. Thus, we could only provide some general introduction on the bidding strategies available to the traders

Each trader of the CAT game uses one of the following four strategies: ZI (Zero Intelligence), ZIP (Zero Intelligence Plus), GD (Gjerstad Dickhaut) and RE (Roth and Erev) [8].

2.2.1 Zero Intelligence

The Zero-Intelligence (ZI) strategy is derived from a concept first put forward by Gode and Sunder [10]. In their description of a ZI agent, Gode and Sunder wrote that "*it has no intelligence, does not seek or maximize profits, and does not observe, remember, or learn. It seems appropriate to label it as a zero-intelligence trader*". In other words, ZI represents a non-history-based and non-reactive strategy.

Gode and Sunder used these computer agents to simulate market transactions in double auctions. There are two type of ZI traders,constrained and unconstrained Zero-Intelligence traders, labeled receptively as ZI-C and ZI-U traders. ZI-C traders are subject to budget constraints and the traders are not allowed to trade at loss. ZI-U traders have no limit on their shouts. They can submit shouts that are higher than their limit prices. In this case, the traders would suffer loss from the transactions.

In the CAT game, we only consider ZI-C traders. The shouts are drawn from a uniform distribution over a given range. The ZI-C buyer forms a bid from a value drawn from a uniform distribution between 0 and p_{min}. The ZI-C seller forms an ask from a value drawn from a uniform distribution between its cost price and p_{max}. p_{min} and p_{max} are the prices below or beyond which we assume no transaction can take place.

2.2.2 Zero-Intelligence Plus

The Zero-Intelligence Plus strategy has been primarily used for agents in auctions with multiple buyers and sellers [2]. The main feature of ZIP agents is that they learn a bidding strategy based only on information provided by the results of the previous auction. The ZIP strategy estimates the agent's profit margin based on the history of market information. The agents can adjust their profit margin to remain competitive in the market. The profit margin is calculated as the difference between the agent's limit price and the shout price.

The ZIP agent works in the following way when adapted to the CAT game. At the beginning of the trading day, the agent (the buyer or the seller) generates a low profit margin arbitrarily. When events in the market indicate that it could acquire a unit at a lower price than its current shout price, the ZIP buyer could increase its profit margin. For a ZIP seller, if the last shout resulted in a transaction and its shout price was less than the transaction price, this indicates that it could transact at a higher price which would necessarily increase its profit margin. Conversely, ZIP buyers and sellers reduce their profit margin when the margin is too high to remain competitive. The bidding behavior of the ZIP trader can be summarized in Figure 2.1, where $p_i^b(t)$ and $p_j^s(t)$ are the most profitable offer to buy or sell for ZIP buyer i and seller j respectively at any time during the trading period. $s(t)$ denotes the price of the most recent shout [8].

The profit margin is modified based on the Widrow-Hoff algorithm [23]. At any given time t, the ZIP trader i calculates the shout price according to the equation $p_i(t) = l_i(1 + u_i(t))$, where l_i is the trader's limit price, and $u_i(t)$ is the trader's profit margin. The ZIP trader can adjust its margin by increasing or decreasing

Adaptive Rules for the ZIP Seller:

if (the last shout was accepted at price $s(t)$)

any seller j for which $p_j^s(t) \leq s(t)$ should raise its profit margin

if(the last shout was a bid)

any active seller j for which $p_j^s(t) \geq s(t)$ should lower its margin

if (the last shout was an offer)

any active seller j for which $p_j^s(t) \geq s(t)$ should lower its margin

Adaptive Rules for the ZIP buyer:

if (the last shout was accepted at price $s(t)$)

any buyer i for which $p_i^b(t) \geq s(t)$ should raise its profit margin

if(the last shout was an offer)

any active seller i for which $p_i^b(t) \leq s(t)$ should lower its margin

if (the last shout was a bid)

any active seller i for which $p_i^b(t) \leq s(t)$ should lower its margin

Figure 2.1: The ZIP Trading Strategy

$u_i(t)$, $u_i(t) \in [0, \infty)$.

The initial value of the profit margin, $u_i(0)$, is drawn from a uniform distribution over the range [0.1, 0.5] at the beginning of the game. The learning mechanism of the profit margin is constructed by the function of:

$$u_i(t+1) = (p_i(t) + \Gamma_i(t+1))/l_i - 1$$

The $\Gamma_i(t+1)$ is defined as $\Gamma_i(t+1) = \gamma_i \Gamma_i(t) + (1-\gamma_i)\Delta_i(t)$, where $\Gamma_i(0) = 0$, $\Delta_i(t) = \beta_i(\tau_i(t) - p_i(t))$ and $\tau_i(t) = R_i(t)s(t) + A_i(t)$

The learning coefficient, $\beta_i \in [0.1, 0.5]$, determines the rate of convergence of the trader's shout price toward the target price $\tau_i(t)$. R_i is a randomly generated coefficient that sets the target price relative to the submitted shout price $q(t)$. $A_i(t)$ is an absolute perturbation. $A_i(t)$ is drawn from a uniform distribution over [0,0.05] for an increase and over [-0.05,0] for an decrease.

2.2.3 Gjerstad-Dickhaut

The Gjerstad-Dickhaut (GD) strategy for continuous double auction (CDA) is a memory based agent architecture [9]. GD traders have a strategy for shout price selection based on maximizing expected profit. The maximization of expected profit relies on the GD trader forming a belief and utility functions. The shout price (bid b^* and ask a^*) is constructed as the product of the belief function $(\hat{q}(b), \hat{p}(a))$ and risk-neutral utility function $(\pi(b), \pi(a))$.

$$b^* = \arg \max_{b \in (o_{ask}, o_{bid})} [\pi(b) \cdot \hat{q}(b)]$$

$$a^* = \arg \max_{a \in (o_{ask}, o_{bid})} [\pi(a) \cdot \hat{p}(a)]$$

Belief Function

The traders form their beliefs on the basis of the history of observed market data and on the frequencies of submitted bids and asks and of accepted bids and asks resulting in a transaction. The GD strategy also considers the notion of recency, limiting the trader's memory length to a few transactions.

The bid and ask frequencies used in the belief function are defined in the following. $\forall d \in D$, where D is the set of all permissible shout prices in the market, B(d) is the total number of bid offers made at price d, TB(d) is the frequency of accepted bids at d, and RB(d) the frequency of rejected bids at d. A(d) is the total number of ask offers made at price d, TA(d) is the frequency of accepted asks at d, and RA(d) the frequency of rejected asks at d.

The Seller's Belief Function for each potential ask price, a, is defined as:

$$\hat{p}(a) = \frac{\sum_{d \geq a} TA(d) + \sum_{d \geq a} B(d)}{\sum_{d \geq a} TA(d) + \sum_{d \geq a} B(d) + \sum_{d \leq a} RA(d)}$$

The Buyer's Belief Function for each potential bid price, b, is defined as:

$$\hat{q}(b) = \frac{\sum_{d \leq b} TB(d) + \sum_{d \leq a} A(d)}{\sum_{d \leq a} TB(d) + \sum_{d \leq a} A(d) + \sum_{d \geq a} RB(d)}$$

The seller's belief function, $\hat{p}(a)$, is based on the following assumptions.

- If an ask $a' < a$ has been rejected, then an ask, a, will also be rejected.
- If an ask $a' > a$ has been accepted, then an ask submitted at a will also be accepted.
- If a bid $b' > a$ is made, then an ask $a' = b'$ would have been taken, since they assume that this ask a' would be acceptable to the buyer who bid b'.

Similar assumptions are made about the buyer's belief function, $\hat{q}(b)$.

The seller's belief function is modified to satisfy the spread reduction rule[1]. Thus, for any ask that is higher than the current outstanding ask, the belief function is set to 0. Similarly, for the buyer, the belief function is set to 0 when any bid submitted is lower than the outstanding bid.

Utility Function

Gjerstad and Dickhaut view traders as risk-neutral, whose utility function is linear. Their profits can be calculated as the difference between the seller's ask price and its cost price, and the difference between the buyer's bid price and its limit price. When the trader's maximum expected surplus is negative, there is no incentive to submit a bid or an ask and the trader abstains from bidding. Thus, the trader's utility function is formulated as follows:

For a buyer i,

$$\pi(b) = \begin{cases} \ell_i - b & \text{if } b < \ell_i \\ 0 & \text{if } b \geq \ell_i \end{cases}$$

For a seller j,

$$\pi(a) = \begin{cases} a - c_j & \text{if } a > c_j \\ 0 & \text{if } a \leq c_j \end{cases}$$

[1]Any ask that is permissible must be lower than the current outstanding ask, so each new ask either results in a trade or it becomes the new outstanding ask. A similar remark applies to bids.

2.2.4 Roth-Erev

The Roth-Erev algorithm was designed to model how humans play in repeated games against multiple strategic players. The strategy is based on probabilistic choices: agents are faced with a given set of actions, each of which produces a particular reward at each time-step. The information about these rewards enables them to associate an action to a propensity. All the propensities are translated into probabilities to choose an action when the agent needs to act. For the detailed information and its adaption to the CAT game please refer to [4, 8].

2.3 Summary

In this chapter, we provided the detailed information about the CAT system. We looked at the overview of the game, trading agents, specialist setup, and the game assessment criteria. In addition, the trading strategies (ZI, ZIP, GD, and RE) used by trading agents are also provided.

Chapter 3

Policy Designs for CAT Specialists

Our objective is to design a specialist that maximizes its overall game score. In this chapter we provide an overview of the different kinds of policies (Accepting Policy, Pricing Policy, Clearing Policy and Charging Policy) that can be adjusted to achieve better results. We then describe the modified policy strategies we propose for our entrant specialist named WaterCAT. Test results from actual games are presented to provide empirical support for our proposed policy modifications.

The policies can be organized into two categories: **Market Policy** and **Charging Policy**. It is possible to further divide the market policy into three interrelated components: the accepting policy, the clearing policy and the pricing policy. In the rest of this chapter we discuss each of them and our policy design strategies for the WaterCAT specialist.

3.1 Market Policy - Accepting Policy

The accepting policy determines which shouts (i.e., bids and asks) should be accepted. A specialist has the option to reject shouts when they do not conform to the specialist's policy, so that the maximization of TSR score can be assured. The WaterCAT accepting policy is designed based on the Equilibrium-beating accepting policy, but the original Equilibrium-beating accepting policy [15] does not have the capability of adjusting itself in response to market feedback. In order to enable the specialist to reset its accepting range dynamically, we added the market monitoring function and took into account the possible skewness of trader offers.

3.1.1 Equilibrium-beating accepting policy

The original equilibrium-beating accepting policy calculates the market price based on previous transactions. First, the policy calculates the equilibrium market quote, which is estimated through learning algorithms such as sliding-window-average learning [15]. The window size is set to α days, and the learner calculates the average transaction price of the past α days. The result is the equilibrium market quote, the expected value of the past α days' transactions.

E: Equilibrium price

P^t: Transaction price of day t

α: Window size

$$E = \frac{\sum_{t=i}^{i+\alpha} P^t}{\alpha}$$

Delta, δ, is another component of the equilibrium-beating accepting policy. Delta defines the accepting range of market quotes. Two bar values are set for bids and asks, labelled as **ExpectedLowestBid** and **ExpectedHighestAsk**. The ExpectedLowestBid is calculated by subtracting δ from the equilibrium market quote and ExpectedHightstAsk is the sum value of that quote and δ. With this range component, bids and asks will receive additional margin to meet the accepting standard. The bids that are lower than but very close to the equilibrium market quote will be accepted into the market. Such an accepting policy releases the single-value restriction in a reasonable scale while still keep its tightness by having a proper value of δ. ExpectedHighestAsk follows the same logic but in the opposite direction. In short, as long as the discrepancies are within the range defined by Delta, the market would accept the bids that are higher than the ExpectedLowestBid, and the asks that are lower than the ExpectedLowestAsk.

- ExpectedLowestBid=Equilibrium Market Quote - δ
- ExpectedHighestAsk=Equilibrium Market Quote + δ

In Figure 3.1, the solid line is the equilibrium market quote in the time series of trading days. We set $\delta = 5$. According to the formula, two extra lines are generated, ExpectedLowestBid (dot line) and ExpectedHighestAsk (dash line). All the bids above the ExcpectedLowestBid line and those asks below the ExpectedHighestAsk line would be allowed into the market. In this way, the trading agents are

Figure 3.1: Quotes chart of equilibrium-beating accepting policy

given extra margin to enter the market and complete the transactions.

The original design of the δ is a fixed value that applies equally to both bid and ask sides throughout the game. The delta can be the percentage of the market quote or a predefined absolute value. However, the absolute value approach needs to identify the appropriate range of the market quote with extreme caution; otherwise, the specialist may specify unrealistic accepting margins that may invite too much unqualified traders.

In the CAT game, therefore, the percentage approach is often preferred [15], especially because the traders' pricing range is confidential to the specialist. The percentage design would work well if the agents (bids and asks) were of the same quality. The same delta value would allow both sides to have the same level of tolerance. However, if one side outperforms the other, assigning the same delta would lead to an inefficient solution. Such quality imbalance occurs more often than not, both in simulation and in real market transactions, because the market needs to deal with a large variety of agents. For instance, when the market receives heavily skewed numbers of overqualified bids and underqualified asks in one round, assigning the same delta value to both sides may lead to the case where the good bids are outnumbered for a matching pair because the market may reject too many asks that could still be matched if the policy recognized the skewness of participating agents. Thus, the next question becomes how to set up an accepting policy that can calculate the delta values dynamically and respectively for asks and bids in line with actual market conditions.

3.1.2 Market Monitoring Function

As discussed above, the equilibrium quote price is the average transaction price of the last α days, and the traditional percentage approach is to set the delta to a certain percentage of the quote price. This setup works well for most conditions, but it would fail to optimize the matching mechanism if the variance of the trading agent offers is irreconcilable.

Let us look at an example demonstrating how the process works.

We assume the current equilibrium quote E is 5, and Delta is set at 1, 20% of the equilibrium quote. According to the functions in the previous section, we have

ExpectedLowestBid=5-1=4

ExpectedHighestAsk=5+1=6

The accepted agents would be the bids with offering price 4 or more and asks with offering price 6 or less. If the true distribution of the agent offers is close to this case, then this model works well. However, if the distribution of the asks is around price 1, the ExpectedLowestBid 4 would not be the true reflection of the market. The market would reject bids with price range from 1 to 4, all of which could still be matched with the asks. In other words, the rigidly preset, single-value percentage delta excludes many potentially matchable pairs, depriving the specialist of some profit-generating transactions.

In order to make our accepting policy adjust itself according to the market condition dynamically, we add two learner functions to the policy, namely **AskQuote** and **BidQuote**. AskQuote learns the average ask offers in a sliding window, while BidQuote learns the average bid offers in the same window. Compared with the equilibrium quote price that calculates the average transaction price in a sliding window, AskQuote and BidQuote compute the expected value of all shouts, not just the transaction prices.

ask^i: Ask price of agent i

bid^i: Bid price of agent i

n: Number of agents

$$AskQuote = \frac{\sum_i^n ask^i}{n}$$

$$BidQuote = \frac{\sum_i^n bid^i}{n}$$

The reason we introduce AskQuote and BidQuote is that we want to have two separate delta values for asks and bids, namely δ_a and δ_b. In this way, our amended policy could adjust the accepting bar value according to actual market conditions. Recall that the original formulas **ExpectedLowestBid=Equilibrium Market Quote -** δ and **ExpectedLowestAsk=Equilibrium Market Quote +** δ. The new formulas with δ_a and δ_b are

ExpectedLowestBid=Equilibrium Market Quote - δ_a

ExpectedHighestAsk=Equilibrium Market Quote + δ_b

Note that b and a are applied on the equations for the other side of the matching game, because the distribution of bids would affect the accepting range of asks and the distribution of asks would impact the accepting level of bids. In other words, the distribution of the shouts would alter the accepting bar of the opposite group.

The δ_a and δ_b are calculated in two steps. In Step 1, we calculate the absolute value of the difference between the AskQuote/BidQuote and the MarketQuote. In Step 2, we multiply the value by an adjustment factor, α.

The definitions of the new deltas are summarized in the following,

δ_a=$|Q - Ask_q| \times \alpha_a$

δ_b=$|Q - Bid_q| \times \alpha_b$

Where Q is the Equilibrium Market Quote; Ask_q is the AskQuote; Bid_q is the BidQuote.

The adjustment factors α_a and α_b are used to adapt the levels of deltas. We can amplify the level by setting $\alpha > 1$ or discount the level by setting $\alpha < 1$ but > 0. The alpha can be adjusted individually for bids and asks. In our experiments, we set the alpha at 80% level.

As illustrated above, the quality of the bids and asks should also be considered. If the market faces a condition where the quality of one side of the traders (either bids or asks) is better than the other side, the accepting bar value of the other side may drop substantially. This would result in an imbalance between the accepted bids and the accepted asks, causing a low TSR score. Thus, a counter N is also defined in our policy to monitor the difference between the two sides. If the counter reached a preset limit, our market would not accept the over-supplied shouts. In this way, the market can generate the matching pairs as many as possible while still maintaining a good balance between the two sides. In our experiments, we set $N = 5$.

3.1.3 Experiments

Before presenting our empirical results in the CAT experiments, let us briefly review the other two kinds of accepting policies included in our empirical tests. In addition to the equilibrium-beating accepting policy, the always accepting policy and the quote-beating accepting policy are also commonly used in the CAT competition.

1. **Always accepting policy**. This is the loosest policy which basically accepts any submitted shout. It gives agents complete freedom to enter the market with no restrictions, so as to maximize the potential capture of market share. But since many inferior shouts would be allowed in the market, the TSR score may suffer significantly.

2. **Quote-beating accepting policy**. This policy only accepts the shouts which are more competitive than the corresponding market quote — the price at which the last transaction is processed. As such, when the market is exposed to a high level of pricing volatility , the accepting price will witness much fluctuation accordingly. This has been commonly used in the real stock markets, and is also called the "NYSE rule" [22].

We describe the CAT experiments in two parts. Part one explains why the Equilibrium-beating accepting policy is selected as the base model of the Water-CAT accepting policy. Part two shows the improvement in results achieved by the suggested policy modifications.

All experiments ran for 100 trading days with 10 rounds per day. The trader population comprised 50 ZIP traders, 50 RE traders, 50 ZI-C traders and 50 GD traders. Buyers and sellers were evenly split in each trader sub-population. We ran the tests ten times and take the average scores.

Base Model Selection

The first experiment uses just two specialists named M1 and M2. M1 is using the always accepting policy and M2 is using the quote-beating policy. For M1 and M2, all the other policies are identical. Table 3.1 shows the summary of the tests. Except the Market Share score, M2 outperformed M1 in other categories by a remarkable margin. The reason that the Market Share score of M1 was better than M2 is because M2 drove out extra-marginal traders[1] and those traders were forced into the M1. Yet the slightly larger market share didn't improve M1's competitive position, as the extra-marginal traders obviously drag down its scores in TSR and Profit.

Specialist	**Market Share**	**Profit**	**TSR**	**Total Score**
M1	53.98%	19.20%	71.30%	48.16
M2	46.02%	80.80%	96.42%	74.41

Table 3.1: Always Accepting Policy (M1) versus Quote Beating Accepting Policy (M2)

The quote-beating accepting policy in M2 performed well on the Transaction Success Rate because it is a tighter policy — only the shouts that can beat the current market quote could enter the market. However, because of its tightness, the market loses some relatively good shouts (bids slightly lower or asks slightly higher than the current market quote) that could be cleared in the later rounds. Since each trading day includes multiple rounds, the policy results in, unnecessarily, fewer trading agents and fewer matched bid-ask pairs in the market. Thus, the quote-beating policy limits the Market Share and Profit scores while keeping a high

[1] An intra-marginal buyer (resp. seller) is expected to trade in the market because of its limit price is higher (resp. lower) than the market price. The remaining traders are considered as extra-marginal. For detailed information, please refer to the Section Matching Policy

Transaction Success Rate.

The next experiment demonstrates why we select the equilibrium-beating accepting policy as our base model. The specialists M1, M2 and M3 are all generic double auctioneers with the same policy setups except for the accepting policy. The equilibrium-beating policy with δ=20% was applied to M1. M2 used the quote-beating policy and M3 was equipped with the always accepting policy. We ran the game ten times and the average scores were compared. Table 3.2 shows that M1 and M2 both surpassed M3 by a significant margin. Among the two leading policies, however, M1 beat M2 in all three scoring criteria, albeit by smaller margins. As can be seen from the chart, even though M1 has a relatively looser accepting policy than that of M2, it achieved larger market share, generated more matching pairs, and achieved a better TSR.

Specialist	**Market Share**	**Profit**	**TSR**	**Total Score**
M1	34.01%	39.656%	97.613%	57.093
M2	32.175%	34.68%	92.772%	53.209
M3	33.815%	25.665%	76.938%	45.473

Table 3.2: Equilibrium-beating (M1) versus Quote-beating (M2) versus Always Accepting (M3)

These results supported our choice of the equilibrium-beating policy as our base model. We were particularly encouraged to find the TSR score was even improved by the introduction of an extra accepting margin.

The Improvement of WaterCAT Accepting Policy

We ran a series of tests using specialists M1(WaterCAT Accepting), M2(Equilibrium-beating), M3(Quote-beating) and M4(Always Accepting). All the other policies of the specialists were specified identically. Table 3.3 summarizes the results. As predicted, M1 (WaterCAT Accepting) reported the best performance. Although it was not dominant in all three criteria, its leading position was very consistent through all the test games. The reason that M4 (Always Accepting) had the best score on the Market Share was that all the extra-marginal traders restricted by other markets turned to M4. As we discussed earlier, the increase in market share still failed to generate performance benefits for M4 in the other two more important criteria.

Specialist	Market Share	Profit	TSR	Total Score
M1	25.345%	28.065%	96.576%	49.995
M2	25.005%	23.857%	95.85%	48.237
M3	21.875%	28.333%	92.099%	47.436
M4	27.775%	19.742%	75.015%	40.844

Table 3.3: WaterCAT Accepting (M1) versus Equilibrium-beating (M2) versus Quote-beating (M3) versus Always Accepting (M4)

3.2 Market Policy - Clearing Policy

This section discusses another component of our market policy design — the clearing policy, which specifies when and how to clear the market. When to clear defines the time to clear the matched pairs. The specialist has the option to set the clearing time as per round, per day or by random. How to clear focuses on the method of finding matched pairs. The specialist sorts the accepted bids and asks in order and selects one from each group by applying pre-designed algorithms.

There are several ways to condition the **when to clear** policy. One approach is to collect all the bidding and asking offers from each round and clear the market at the end of the trading day. The specialist can maximize profits by matching the highest bid with the lowest ask first. Then the specialist clears the matched pairs from the market and repeats the process on the remaining offers. However, because traders bid for a single unit at a time, this approach would imply that traders have the opportunity to strike only one deal per day and become unable to trade the rest of their multi-unit endowments.

An alternative approach is to maximize the number of transactions (instead of profits) by a continuous clearing rule — if the newly accepted trader could find a matched trader, the market would clear them immediately. The cleared trader can get the opportunity of submitting another offer to the market and seek multiple transactions for each trading day. The disadvantage, however, is that this policy may limit the profits by settling the offers on a timing basis instead of the best offer basis.

The strategy we propose for our specialist WaterCAT adopts a rule positioned somewhere between the two above-mentioned approaches - we suggest clearing the market at the end of each round. For a typical CAT game setup, there are ten rounds per day. The market collects all the shouts from the traders till the end of the round and then triggers the matching function. All the matched pairs will be completed off the list. Then the market starts accepting new shouts for the next round. The cleared traders can submit their shouts in the new round and the remaining/unmatched traders can have new opportunities to shop for a matching pair. In this way, the specialist can be almost as efficient as clearing at the end of the day, while allowing the traders to take the advantage of multiple submissions. By so doing, we will obtain most of the benefits from both approaches without the drawbacks.

The second part of the Clearing Policy is **how to match**. This is one of the most important policies in the thesis. We would discuss the related knowledge and our design in the next section - the matching policy.

3.3 Market Policy - Matching Policy

The third component in our market policy design is the matching policy. The matching policy defines how a market matches shouts submitted by traders. The market receives shouts (bids and asks) from the traders and needs to generate the matching pairs. When the bids and asks satisfy the criteria set by the matching policy, the market would clear the pairs to finish the transactions.

Before we start discussing the details of the policy, we first take a look at the classic demand and supply curve [7]. Figure 3.2 demonstrates that the supply curve and the demand curve cross each other at the **Equilibrium Point**. The left side of the Equilibrium Point is called the **Intra-Marginal** area and the right side the **Extra-Marginal** area. Traders make profits when the transaction is executed in the Intra-Marginal area, where the supply price is lower than the demand price. Conversely, they carry losses when the transactions fall in the Extra-Marginal area as the supply price exceeds the demand price limit. The supply-and-demand curve can be adopted to the bids-and-asks model simply by applying bids to the demand curve and asks to the supply curve. The agents in the left side are called Intra-Marginal since they can make profits and the ones in the right side are called Extra-Marginal. Sometimes we also refer to Intra-Marginal traders as good traders and Extra-Marginal traders as bad traders.

There are basically two types of matching policies, the equilibrium matching and the Max-volume matching. The equilibrium matching policy tends to generate the maximized profits for the trading agents, while the Max-volume matching policy attempts to catalyze the maximum amount of matching pairs.

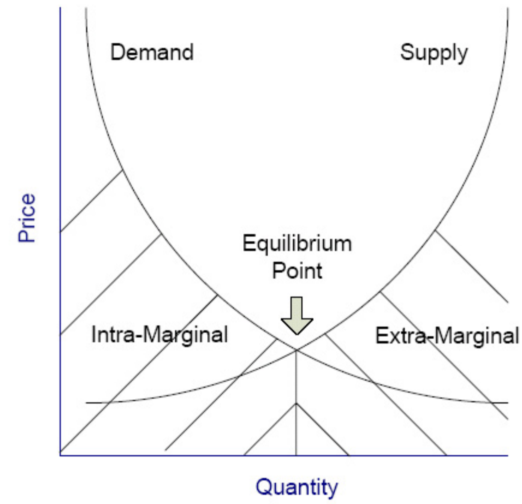

Figure 3.2: Supply and Demand Curve with Equilibrium Point, Intra and Extra Marginal area

- **Equilibrium matching**

  This policy clears the market at the reported equilibrium price and matches the intra-marginal asks with the intra-marginal bids. It can also be called the Good-Good matching policy, because both bids and asks are from the Intra-Marginal area. The policy has a profit-maximizing feature by matching the highest bid with the lowest ask.

- **Max-volume matching**

  This policy aims to increase transaction volume based on the observation that a high intra-marginal bid can match with a lower extra-marginal ask.

In this case, the highest bid offer and the highest but reasonable[2] ask offer would be the matched pair. This policy generates the most matched pairs and transactions for a given set of offers. It is also called the Good-Bad matching policy, because one of the matched pair is from the intra-marginal area and the other is from the extra-marginal side.

In the following examples we demonstrate how the two matching policies work. Two sorted lists are provided, one for the bids in a descending order and the other for the asks in an ascending order.

- **Bids**: $5, $4, $3, $2, $1
- **Asks**: $1, $2, $3, $4, $5

Under the equilibrium matching policy, the intra-marginal bids are matched with the intra-marginal asks. In Table 3.4, the best bid is $5 and the best ask is $1. The $5 bid is matched with the $1 ask, yielding a profit of $4 ($2 to each trader if the profit is evenly split). Following the same rule, the $4 bid is matched with the $2 ask and the $3 bid is matched with the $3 ask. The policy leaves the rest unmatched, because it can not generate a matching pair if the bid price is less than the ask.

Under the max-volume matching policy, the highest bid is matched with the highest (less or equal than the bid price) ask. In Table 3.5, the $5 bid is matched

[2]Value is less or equal than the bid price

Bid	Ask
\$5	**\$1**
\$4	**\$2**
\$3	**\$3**
\$2	\$4
\$1	\$5

Table 3.4: Equilibrium Matching

with the \$5 ask. The same rule applies to the other traders.

Bid	Ask
\$5	**\$5**
\$4	**\$4**
\$3	**\$3**
\$2	**\$2**
\$1	**\$1**

Table 3.5: Max-volume Matching

As seen from the two tables, with the equilibrium matching policy, two agents make profits but the Transaction Success Rate is 60% (3 out of 5). While with the max-volume matching policy, the market clears all the shouts and achieves 100% Transaction Success Rate, but no agent makes profit.

If the specialists offer a free market (all the charges are zero) to the agents, they may hardly earn any profit from such volume-maximizing transactions. Let us recall the scoring function [16], which is the weighted average of Market Share, Profit and Transaction Success Rate (TSR). Thus, under the max-volume scenario, the overall score would only receive contributions from Market Share and TSR.

Let us assume a simple market competition scenario, in which two markets with almost identical setup compete with each other and the only difference lies in the matching policy — Equilibrium matching versus Max-volume matching. From the simple examples described above, it still seems possible that the max-volume matching policy will have competitive advantage over the equilibrium matching policy by scoring a significantly better Transaction Success Rate (100% over 60%). An open question is whether this would actually be verified in practice.

To answer this question, we conducted a series of tests. All the basic game setups resemble those specified in Section 3.1. The aim of the tests is to learn how the traders react to the matching policies when other policy parameters are the same. Interestingly, the result ran against to what we expected - the equilibrium matching policy totally outperformed its max-volume counterpart. The test shows that agents are sensitive to the profits they could acquire from the transactions. If the agents earn no profit or less than what they expected, they tend to switch market. Between making profits and sealing transactions, the traders are willing to work with the market that provides them with more profits but fewer transactions. As discussed in the previous work by *Niu et al.* [15], dominating the market share remains as the key to success, and therefore we will use the equilibrium matching policy as our base model.

Equilibrium Matching can generate good profit and overall performance, but the market only scores 60% on Transaction Success Rate. How can we improve the TSR without sacrificing the traders' profit? The solution is to impose a tighter accepting policy, which stops those low quality extra-marginal traders from entering the market. As we discussed in the accepting policy section, Accepting and Matching policies can complement each other in achieving better specialist performance.

In the next section, we propose a design that combines the benefits of the WaterCAT accepting policy and the equilibrium matching policy.

3.3.1 WaterCAT Matching Policy - a Modified Equilibrium Matching

To summarize the discussions we have presented so far, there are a number of properties desirable in our performance-enhancing matching policy. Some of them , however, are unavailable in the standard equilibrium matching policy.

Attracting more agents to the market while maintaining the existing agents: We propose a policy that could attract new traders to enter the market while sustaining the loyalty of the existing agents. In this way, we can increase our market share.

Be fair to all the agents: Fairness to agents is accomplished by ranking bidding priority according to shout prices. In other words, the higher price the agent is bidding, the higher priority the agent has. Higher priority means better chance for a high-bidding buyer to be match with a good ask. Although our matching policy does not rely only on ranking the agents by their shout prices, the main idea is still to follow the shout price ranking.

Be profitable for the agents: The agents are seeking profits through transactions. Our policy would generate fair amount of profits to agents

Maintaining a high Transaction Success Rate: Transaction success rate is a key factor of the scoring function. Our policy should keep a satisfactory TSR while allowing traders to make profits. As we have already illustrated in the previous examples, there exists a trade-off between securing profits and maximizing TSR.

Computational feasibility: Computational requirements are also important because the running time of each round is limited.

To satisfy those criteria, we start with preparing some ordered lists. The primary job of the matching policy is to generate ordered lists of shouts. The simplest solution is obviously to rank by shout prices, assigning the top position to the highest bid or the lowest ask. Table 3.6 illustrates a descending rank list for bidders.

ID	Price
5	\$5
2	\$4
3	\$3
4	\$2
1	\$1

Table 3.6: Bid Price Ranking

In reality, however, many shops adopt special programs to motivate their customers (i.e., Shoppers Points of Shoppers Drug Mart shop). The shops try to keep their customers by giving bonus or other benefits to their loyal customers or first time buyers. We adapt the same idea to our policy. In the CAT game specification, it is not allowed to attract agents by giving them money from our account. Our alternative solution is to set another ranking by loyalty (Table 3.7) and assign better matching pairs to the loyal agents so that they could profit more from the transactions. The loyalty ranking is based on the association between the agent ID and the number of times that agent has registered with our market specialist.

ID	Loyalty
1	9
4	7
3	5
2	3
5	1

Table 3.7: Loyalty Ranking

The next problem is how to actually aggregate the two very different rankings.

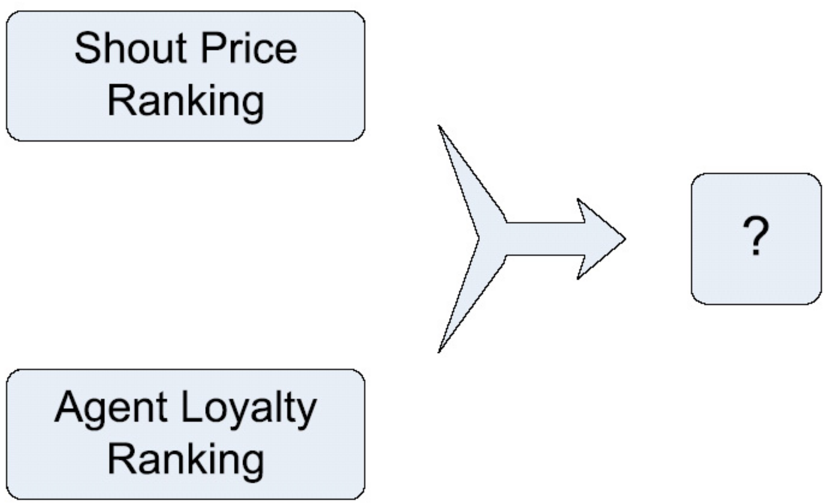

Figure 3.3: How can we aggregate the two rankings together?

The rank aggregation problem occurs when we need to combine many different rank orderings on the same set of candidates for the purpose of designing an ordering with better features. Most ranking aggregation methods are generated under the same or similar criteria [5, 19, 3]. But the orderings we try to combine here are from two totally different groups (one by price and the other by loyalty). If we use those well-known methods such as Kemeny-Young from the voting system [13, 25], the result will be too biased to make sense. One extreme case could be the "most loyal but worst agent", who may appear to be loyal by bidding constantly at $1 throughout the whole game.

After scouring the available aggregation methods, we designed a very simple solution — the **Bonus Factor**. Bonus Factor is generated from the loyalty ranking but is combined with the price ranking to construct a new adjusted ranking. Our matching policy would be based on the new ranking, **Price with Bonus Factor Ranking** (short as PBF Ranking).

The definition of Bonus Factor is described as:

- n_i: The number of times that agent i has registered in the market.
- N: $\max(n_1, \cdots, n_i)$ Maximum value in the loyalty ranking.

$$V_i = \frac{n_i}{N}$$

where V_i is the value that is used for the lookup table in the next step.

$$BonusFactor_i = \begin{cases} \alpha & V_i \in (0, x] \\ \beta & V_i \in (x, y] \\ \gamma & V_i \in (y, 1] \end{cases}$$

where $\alpha, \beta, \gamma \geqslant 1$ and $0 \leqslant x \leqslant y \leqslant 1$.

α, β, and γ can be assigned to any value. Since we want to have the bonus effect for the loyal agents, we assign them with values greater than or equal to one. However, cautions should be taken for the upper range of α, β, and γ because we also do not want the bonus effect to be exaggerated.

The Bonus Factor can be then applied to shout prices to construct the new PBF ranking. For bids, it is achieved by multiplying the Bonus Factor with the shout price. For asks, the value is arrived by dividing ask shout price by bonus factor.

- Bids: $price_{BF} = price_{shout} \times BonusFactor$
- Asks: $price_{BF} = price_{shout} \div BonusFactor$

After applying the Bonus Factor to the Price Ranking, the policy has some new properties in PBF Ranking:

- The loyal agents get better rankings. They also make more profits from transactions by matching with better pairs. A loyal buyer from PBF Ranking would match a seller with lower price comparing with the seller from the original Price Ranking. In Table 3.8, we have a loyal buyer with a shout price \$4 and a Bonus Factor 1.5. By being listed in the PBF Ranking, the buyer enjoys a better ranking (1st instead of 2nd) in comparison with its position in the original Price Ranking. The new matching policy can benefit the loyal buyers by enabling them to make more profits through transactions (matching pair \$4 and \$1 instead of \$4 and \$2).

Price Ranking		PBF Ranking	
Bid	**Ask**	**Bid**	**Ask**
$5	**$1**	**$4**$\times$ 1.5	**$1**
$4	**$2**	**$5**	**$2**
$3	**$3**	**$3**	**$3**
$2	$4	$2	$4
$1	$5	$1	$5

Table 3.8: Comparison between Price Ranking and PBF Ranking

- This method also solves the ranking problem when agents submit shouts with the same price. In Table 3.9, there are 2 bids with the same shout price $4. In the original Price Ranking, it would be ambiguous to determine the proper ranking for the two agents. In PBF Ranking, however, thanks to the additional variance introduced by the Bonus Factor, the market would have a lower probability of ranking ambiguity.

Price Ranking		PBF Ranking	
Bid	**Ask**	**Bid**	**Ask**
$5	**$1**	**$5**	**$1**
$4	**$2**	**$4**$\times$ 1.1	**$2**
$4	**$3**	**$4**	**$3**
$2	$4	$2	$4
$1	$5	$1	$5

Table 3.9: Solving the same price issue

- This method could also attract new trading agents. Bonus Factor can be set to a value greater than one when $V_i = 0$. The new entrants thus reap better profits from transactions comparing with what they can in the other markets. Consequently, our market can move upward in the trading agent's market preference list.

Although our WaterCAT matching policy seems promising by using PBF Ranking, there are a few issues we need to be aware of.

The first is how to set up the proper value of Bonus Factor (value of α, β, γ). We rank the shouts by $price_{BF}$ using PBF Ranking, but the transaction price still uses the original shout price,$price_{shout}$. If the Bonus Factor is set to a value that would raise certain traders' ranking dramatically, the matching policy may face the case where the trader in the high ranking could not be cleared, while the lower ranking trader could make the deal. In Table 3.10, there is a buyer with shout price \$2 and Bonus Factor 2. In PBF Ranking, the ranking of the buyer increases from the fourth to the third. The bid \$2 will match with ask \$3 according to the PBF Ranking. This is impossible to clear, because the market can not clear a transaction with the bid price less than the ask price. However, the fourth buyer \$3 could clear the ask \$3 which would increase TSR from 40% to 60% under the same setup. We solved this issue by checking the next N traders with lower priority. N is configurable by the users.

Price Ranking		**PBF Ranking**	
Bid	**Ask**	**Bid**	**Ask**
\$5	**\$1**	**\$5**	**\$1**
\$4	**\$2**	**\$4**	**\$2**
\$3	**\$3**	**\$2**×2	**\$3**
\$2	\$4	\$3	\$4
\$1	\$5	\$1	\$5

Table 3.10: Proper Bonus Factor value for PBF Ranking

The second issue is the **Canceling Effect**. If we apply the Bonus Factor to both side of asks and bids, the matched pair for either side may not be the optimum response. In Table 3.11, buyer \$4 and seller \$2 both have Bonus Factor 2, and they are our loyal traders. According to the design of the PBF Ranking, they should match with the shouts that could give them better profits. In this case, the best response for buyer \$4 is ask \$1 and the best response for seller \$2 is bid \$5. However, after applying Bonus Factor to both sides, bid \$4 and ask \$2 become the matching pair, which clearly is not the best response for both sides. The solution is to apply the Bonus Factor only to either the bids or the asks side, depending on the market condition. If there are fewer buyers in the market, the market should apply Bonus Factor to the bids side to attract more buyers to our market.

Price Ranking		**PBF Ranking**	
Bid	**Ask**	**Bid**	**Ask**
\$5	**\$1**	**\$4**×2	**\$2**÷2
\$4	**\$2**	**\$5**	**\$1**
\$3	**\$3**	**\$3**	**\$3**
\$2	\$4	\$2	\$4
\$1	\$5	\$1	\$5

Table 3.11: Canceling Effect of PBF Ranking

3.3.2 Implementation

The most straightforward implementation of the matching policy would be maintaining a sorted list of all shouts. However, this approach is not efficient enough for the CAT game. In the game, the specialist is dealing with a large number of agents, who may submit multiple offers in each round. The specialist needs to organize all the offers and generate matching pairs with the imposed time constraints. We can improve the efficiency by employing two sorted lists, one for the bids and one for the asks. Once again, the efficiency of Insert, Delete and Generate matching pairs is not good enough, especially when the market is dealing with a large number of agents.

A better solution is to develop four heap structures to organize the shouts [1, 18]. We adapted the four heap algorithm developed by Wurman and his colleagues [24]. The shouts are categorized into matched shouts (asks and bids) and unmatched shouts (asks and bids). The four heaps are:

B_{in}: Contains all of the bids that are in the current match set. The heap priority is the minimal price with Bonus Factor, so that the lowest priced bid with the Bonus Factor is on the top.

B_{out}: Contains all of the bids that are not in the current match set. The heap priority is the maximal price with Bonus Factor.

A_{in}: Contains all of the asks in the match set. The heap priority is the maximal price with Bonus Factor.

A_{out}: Contains all of the asks not in the match set. The heap priority is the minimal price with Bonus Factor.

b_{in}, b_{out}, a_{in} and a_{out} are the top nodes of B_{in}, B_{out}, A_{in} and A_{out}, respectively. Value(n) is the value of node n. The following constraints are enforced:

- Value(b_{in}) $\geqslant$ Value(b_{out})
- Value(a_{out}) $\geqslant$ Value(a_{in})

- Value(a_{out}) $\geqslant$ Value(b_{out})

- Value(b_{in}) $\geqslant$ Value(a_{in})

The 4-heap algorithm was originally designed for the single bid auction, but the CAT game is a competition involving multiple units. Therefore a split function is added to make the matching with the same quantities. The split function would choose the lower quantity of the matched pair as the standard quantity, and then split the high quantity shout according to the standard quantity. Finally, the split function makes the matching with the same quantities and put the remaining shouts back to the heap.

Update agent loyalty history
Calculate Bonus Factor and apply to the shout
if ((Value(a_{new}) $\leqslant$ Value(b_{out})) **and** (Value(a_{in}) $\leqslant$ Value(b_{out})) **then**
  Put(a_{new}, A_{in})
  b ←Get(B_{out})
  Put(b, B_{in})
else if (Value(a_{new})<Value(a_{in})) **then**
  a ←Get(A_{out})
  Put(a, A_{out})
  Put(a_{new}, A_{in})
else
  Put(a_{new}, A_{out})
end if

Figure 3.4: Pseudo code for receiving a new ask

After the policy place all the shouts in the four heaps, the matching procedure would easily improve its efficiency. The program can just pop the shouts from B_{in} and A_{in}, and then match the bids and asks with the same quantities by using the split function.

3.3.3 Experiments

To further examine the validity of our proposed WaterCAT matching policies, we conduct a series of games with the same setup as those defined in Section 3.1.

Experimental setup

Out aim is to examine the performance of our modified WaterCAT matching policy against that of the original equilibrium matching policy. The same scoring criteria were used as those in the tournament but we assess all the game days. We ran the game 20 times and evaluate the average scores of the games.

Experiment Result

As expected, the WaterCAT matching policy outperformed the Equilibrium matching policy.

Matching Policy	**Average Market Share**	**Maximum Market Share**	**Average Transaction Success rate**	**Maximum Transaction Success Rate**	**Average Overall Score**	**Maximum Daily Score**
WaterCAT Matching	76.54%	98%	92.466%	98.3%	56.33	0.647
Equilibrium Matching	72.09%	88%	93.35%	97.3%	55.144	0.607

Table 3.12: WaterCAT Matching Policy V.S. Equilibrium Matching Policy

From Table 3.12, the WaterCAT matching policy produced better results than the equilibrium matching policy in almost all scoring criteria.

- Market Share: WaterCAT leads in both average and maximum scores. The 4.45% advantage in average score represents a significant improvement, considering that the other specialists use the same policies except for our suggested additions for the WaterCAT specialist.

- Transaction Success Rate:The equilibrium matching policy outperforms the WaterCAT in the average category, but the difference is less than one percent. In the maximum value category, WaterCAT leads the equilibrium matching policy by one percent. It is reasonable to say that the two policies are on a par when TSR is the measurement.

- Profit: We did not apply any charging policy to the test, because we want to test the performance of the matching policy under a free market condition. But suppose we apply charges to agents, assuming the same charging policy to all the specialists, WaterCAT would probably outperform the equilibrium matching rival, because WaterCAT leads in market share and larger market share would generate more transactions to increase profits.

- Overall Score: The WaterCAT leads in Market Share and Profit and ties in Transaction Success Rate. Overall the WaterCAT policy outperforms the equilibrium matching one.

3.4 Pricing Policy

We now turn the fourth part of our policy design discussion. The pricing policy is responsible for determining the transaction price for matched ask-bid pairs. The decision may involve only the prices of the matched ask and bid, but other information may also be relevant, such as the market quotes.

The Pricing Policy we chose for our WaterCAT specialist is discriminatory k-pricing policy with k=0.5 [15]. This policy sets the transaction price of a matched ask-bid pair at somewhere in the interval between the two prices. The parameter k in [0,1] controls which points is used and usually takes the value of 0.5 to avoid the bias in favor of either buyers or sellers. Beside the discriminatory k-pricing policy, there are other possible alternatives.

- Side-biased pricing. This is basically discriminatory k-pricing policy with k set to split the profit in favor of the side where fewer shouts exist.
- Uniform k-pricing. This policy sets the transaction price for all matched ask-bid pairs at the same point between the ask quote and the bid quote.

We choose the discriminatory k-pricing policy with k=0.5 over the side-biased policy because we failed to observe any performance improvement from the side-biased pricing in our previous tests. Moreover, the discriminatory k-pricing policy is an independent policy that counts only on the matched pair, which makes it fast to be calculated and superior in terms of algorithm efficiency. The difference between the discriminatory and the uniform k-pricing policies is that the former sets the transaction price for each matched pair, so that the specialist can generate a list of different transaction prices for each cleaning round or day[3]. The uniform policy, in comparison, sets only one transaction price for all the matched pairs. The major disadvantage of such a policy is that it cannot be used together with the max-volume matching policy, because the price intervals of some matched ask-bid pairs do not cover the spread between the ask quote and the bid quote. Thus, we decided to use the discriminatory 0.5-pricing policy.

[3]Clearing time depends on the clearing policy setup

3.5 Charging Policy

The final ingredient of policy recipe comes from the charging policy. The charging policy determines the specific charges levied upon the traders in the system. A registration fee is paid by the traders to register with the market specialist of their choices at the beginning of the day, irrespective of whether they transact or not. An information fee is required if transaction history information is obtained. The shout fee and the transaction fee are the amount paid respectively when a shout is placed and when a transaction occurs. The profit fee is the percentage of the difference between the accepted shout and the transaction price that is paid by the traders to the market.

3.5.1 Fixed Charging Policy Approach

We first study the agent's sensitivity towards the charging fees. We set up two generic double auction specialists adopting the same policies (equilibrium-beating accepting policy, discriminatory 0.5-pricing policy, round clearing policy, equilibrium matching policy and fixed charging policy). We ran a series of tests with one specialist charging different kinds of agent fees and the other offering free access to the market. We monitor the Market Share to measure the sensitivity. If the Market Share drops dramatically once the specialist starts demanding a specific fee, we can conclude that the agents are sensitive to the fee. We categorized the results in the following chart.

According to the experiments, the agents seem very sensitive to the fees, especially the registration fee, the shout fee and the transaction fee. Probably because

Fee Name	Market Share Drop Percentage	Reaction
Registration Fee	36.46%	Very Sensitive
Shout Fee	23.47%	Sensitive
Transaction Fee	10.08%	Sensitive
Profit Fee	5.23%	Not Sentitive

Table 3.13: Fees Study

the information fee charges the other data-requesting market specialists rather than the trading agents, we observed that it has limited impact on criteria such as the transaction success rate, so we focussed our study on the registration, shout, transaction and profit fees.

As seen from the test results, our market should not charge the registration fee. Once the market start charging registration, its market share drops remarkably. The reason is simple. Nobody likes paying fees without getting any promise of making profits. In the CAT game, this means few agents would pay the registration fee without being assured that they could be matched and generate profit from the market.

The shout fee has a similar effect as the registration fee. The difference is that the market charges the shout fee only after the agent has become an member of the market. In other words, the agents are like “sitting ducks”. They have to pay for each shout without much promise of completing the transaction. Depending on the agent’s market selection strategy, it will leave the market and select another specialist in the next round if the fee is deemed unfair. In the long run, these two fees would inevitably decrease the market share.

The transaction fee and the profit fee are of the same kind — they charge the agents when the market completes a transaction. During a transaction, as both sides obtain some profit, the specialist is allowed to take a portion of that profit. The difference is that the transaction fee is of a fixed value and the profit fee is by percentage. The agents would not leave the market if the transaction fee is required by a fair amount, although, admittedly, it is difficult for the specialist to determine a proper value. The decision factors may involve the shout value of the matched pair and the profit generated from the transaction. In contrast, the profit fee is much easier and will not overcharge agents because of its percentage property.

Based on the exhaustive study on each possible charging scheme, our conclusion is that we should charge Profit Fee if we decide include the charging policy.

Winning Number of the Profit Fee

The profit fee is a portion of the profit made by the traders, calculated as the difference between the shout and the transaction price. As explained above, we chose the profit fee as the only charging scheme in our policy design. But how much should we charge? We will provide the answer by running some explorative tests to see if there is a winning number.

All the basic game setups remain the same as those described in the previous sections. As shown in Table 3.14, the profit fee started from 0% and went up to 90% by an increment of 10% each time. Surprisingly, we did find a magic number — 20%.

Profit Fee 1 (Score)	Profit Fee 2 (Score)	Result
20% (79.98)	0% (51.37)	20% win by 55.7%
20% (68.58)	10% (62.84)	20% win by 9.1%
20% (74.77)	30% (54.6)	20% win by 32.1%
20% (72.46)	40% (59.04)	20% win by 22.7%
20% (73.95)	50% (57.29)	20% win by 29.1%
20% (73.58)	60% (57.64)	20% win by 27.6%
20% (70.68)	70% (60.59)	20% win by 16.6%
20% (73.79)	80% (56.18)	20% win by 31.3%
20% (71.71)	90% (58.99)	20% win by 21.6%

Table 3.14: 20% VS other Profit Fees

From the table we can see that the 20% level beats all the other numbers. To be sure of this result, we also ran a series of experiments with fee levels around 20%. We did not observe any convincing alternative as either the winning margin is too small or there is no consistent wins. As a result, we will use 20% Profit Fee as our percentage charging policy.

3.6 Summary

In this chapter, we provided an overview of all the policy candidates for our design effort, including the Accepting Policy, the Pricing Policy, the Clearing Policy and the Charging Policy. Particularly, we presented detailed suggestions for the WaterCAT accepting and matching policies built on top of the baseline models. We explained the rationales behind each proposed modification, and then reported test results based on the actual CAT game experiments. In most cases, our suggestions were supported by significant improvement in specialist performance.

Chapter 4

Post-game Analysis

As described in the previous chapter, the majority of our policy suggestions received positive support in individual tests. However, the effectiveness of our overall strategy remains to be substantiated. To do so, WaterCAT, our designated specialist entered CAT Tournament 2009. In this chapter we will describe the game strategies used by our WaterCAT specialist and examine the reasons behind both successes and failures. Because of the technical issue during the official games, we did not fully test the performance of our specialist. In order to demonstrate WaterCAT's performance, we tested our specialist in the tournament re-run simulations using post-tournament specialists. Our specialist WaterCAT completed the games as one of the top three in both short and long period simulations.

4.1 Trial Game Analysis

Each CAT simulation consists of a single tournament that runs for a number of trading days. All games were carried out with the JCAT server and all clients were connected remotely. The trial game ran for 250 trading days with 10 rounds per day. The trader population was 400, and buyers and sellers were split evenly. Although the game organizer did not reveal the distribution of the trading agent strategies, we have reasons to believe that the distribution in the trial game was similar to the setup in the official games, which are presented in the next section. Each team was permitted to have two specialists in the game. The specialists of the trial game include: BazarganZebel, CrocodileAgent, IAMwildCAT, IAMwildCAT2, Mertacor, Mertacor2, MetroCat, PSUCAT, PSUCAT2, PersianCAT, TWBB, TWBB1, TWBB2, Tianuani, UMTac09, UMTac091, WaterCAT1, WaterCAT2, cestlavie, cestlavie2, jackaroo, jackaroo2, rucat0 and rucat1.

Out team was a knockout success in the trial game. We had two agents, WaterCAT1 and WaterCAT2, and both of them ended as one of the top two. The strategy setup of the two specialists were the same except for the charging policy.

Trial Game Strategy Setup

Accepting Policy: WaterCAT Accepting Policy

Clearing Policy: Round Clearing + WaterCAT Matching Policy

Pricing Policy: Discriminatory 0.5 Pricing Policy

Charging Policy: Fixed Charging Policy (WaterCAT1: Free Market; WaterCAT2: 20% Profit Charging)

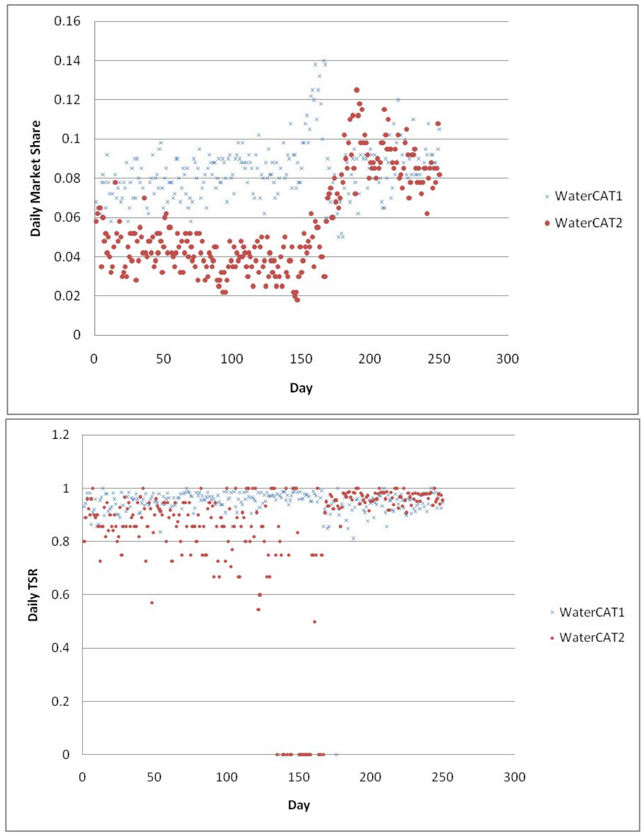

Figure 4.1: Market Share(upper) and TSR(lower) Distributions of WaterCAT1 and WaterCAT2

The reason we introduced some variance in the charging policy is that we wanted to further test the performance of the different charging policies when the specialist was facing a more sophisticated and competitive environment. We would pick the winning one as our official competition specialist. Figure 4.1 shows that WaterCAT1 outperformed WaterCAT2 in Market Share and TSR. Because of its free market strategy, WaterCAT1 attracted more agents to participate its bidding process. Thus, WaterCAT1 created a better chance of matching pairs and clearing them. As the game proceeded, WaterCAT2's Market Share caught up with WaterCAT1's. The main reason was that the loyal intra-marginal traders would get better matching pairs and thus brought about more profits for WaterCAT2, a result of the WaterCAT matching policy. After trying out the other markets, the intra-marginal traders decided to stay in our market, producing a stable stream of ongoing profit. Our Market Share also increased as the game unfolded. As discussed in the system design chapter, the intra-marginal agents can be easily matched and cleared. Because the market secured a steady group of intra-marginal traders, WaterCAT2's TSR also increased with desirable consistence. In Figure 4.1, we can see that after Day 150, as the Market Share of WaterCAT2 started climbing up, its TSR maintained at a high and steady level (over 90%) with less volatility.

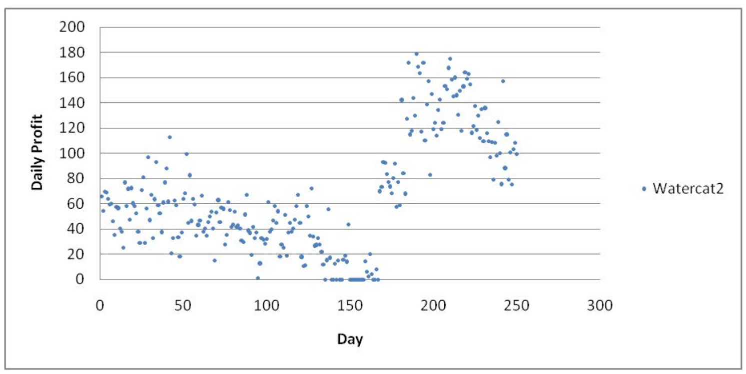

Figure 4.2: Profit Distribution of WaterCAT2

WaterCAT1 performed better than WaterCAT2 in terms of Market Share and TSR, but the final winner was WaterCAT2. That is because WaterCAT2's lead in the Profit score seemed to offset its lags in the other two criteria. We can observe such nuanced dynamics even more clearly if we examine Figure 4.1 and Figure 4.2 together. As the Market Share started soaring around Day 150, the time when WaterCAT2 obtained a steady group of intra-marginal traders, the Profit also started to rise. Since the intra-marginal traders can generate more profits. WaterCAT2 managed to preserve a relatively high level of profit gains comparing with its scores in the early stage. Thus, the total score of WaterCAT2 win over that of WaterCAT1.

Specialist	Overall Score	Market Share Score	Profit Score	TSR Score
WaterCAT2	86.819	14.188	38.269	208.006
WaterCAT1	85.422	20.95	0	235.345
PersianCAT	82.372	21.462	0	225.665
cestlavie	81.585	15.604	25.968	203.168
jackaroo2	76.651	13.166	50.717	166.063
cestlavie2	72.822	17.299	3.92	197.261
UMTac09	68.025	14.397	19.981	169.704
CrocodileAgent	59.771	14.441	9.028	155.886
Mertacor	58.073	11.688	26.958	135.568
jackaroo	57.561	14.59	6.927	151.19
IAMwildCAT	56.827	11.442	1.821	157.203
UMTac091	55.138	11.536	10.108	143.778
TWBB1	51.415	7.977	7.377	138.909
PSUCAT	45.255	10.496	0	125.263
Mertacor2	40.751	9.044	8.445	104.781
IAMwildCAT2	36.918	9.865	0	100.889
MetroCat	36.85	4.164	30.91	75.484
TWBB	30.865	7.388	7.679	77.536
rucat1	23.47	6.823	0.043	63.536
TWBB2	20.574	5.054	0.991	55.658
rucat0	18.923	5.362	0.178	51.233
Tianuani	8.051	2.069	0.693	21.375
BazarganZebel	2.89	0.981	0.012	7.681

Table 4.1: Trial Game Score Summary

Based on the result of the trial game, we chose to use WaterCAT2's strategies for our competition specialist.

4.2 Game Analysis

We entered the 2009 TAC Market Design Tournament Final Games confident with the success in the trial game. However, the end result is not what we expected, which can be largely explained by the platform changes exerted by the game organizer. We will discuss the causes in details in the following section.

4.2.1 CAT Game Setup

The entrants of the final contest were: BazarganZebel, CUNY, Cheshire, CorcodileAgent, IAMwildCAT, Jackaroo, Lancashire, Mertacor, MertroCat, PSUCAT, PersianCAT, TWBB, Tiannuani, UMTac09, WaterCAT, cestlavie and rucat0. The contest included three games, each ran for 500 trading days with 10 rounds per day. The trader population was 400. Buyers and sellers were evenly split with 200 each group. However, several aspects of the setup differ from game to game, and the details are listed as follows.

GAME G1:

- Assessment (scoring) days

  Start day = 12; End day = 480

- Numbers of traders of each type (out of 200)

  GD = 50; ZIP = 60; RE = 50; ZIC = 40

- Range of private values

  Buyer.minvalue = 30; Buyer.maxvalue = 130

  Seller.minvalue = 30; Seller.maxvalue = 130

GAME G2:

- Assessment (scoring) days

  Start day = 23; End day = 491

- Numbers of traders of each type (out of 200)

  GD = 60; ZIP = 40; RE = 70; ZIC = 30

- Range of private values

  Buyer.minvalue = 80; Buyer.maxvalue = 180

  Seller.minvalue = 80; Seller.maxvalue = 180

GAME G3:

- Assessment (scoring) days

  Start day = 17; End day = 477

- Numbers of traders of each type (out of 200)

  GD = 60; ZIP = 50; RE = 60; ZIC = 30

- Range of private values

  Buyer.minvalue = 60; Buyer.maxvalue = 160

  Seller.minvalue = 60; Seller.maxvalue = 160

4.2.2 Failure Analysis

The CAT competition organizer released a new version of the game platform right before the official game's commencement. The update was deemed necessary because some competition entrants experienced time-out issues during the trial game.

According to the release note, the organizer revised the related connection timeout functions and the new platform was compatible with the old version. However, from what we witnessed in the game and analyzed afterwards, we found that the organizer also changed some of the shout handling functions. To be specific, they altered the type of the shout from a generic Object class to a specific Shout class. Such a modification may not affect other entrants, but it had a deadly impact on us, leading our matching policy to malfunction.

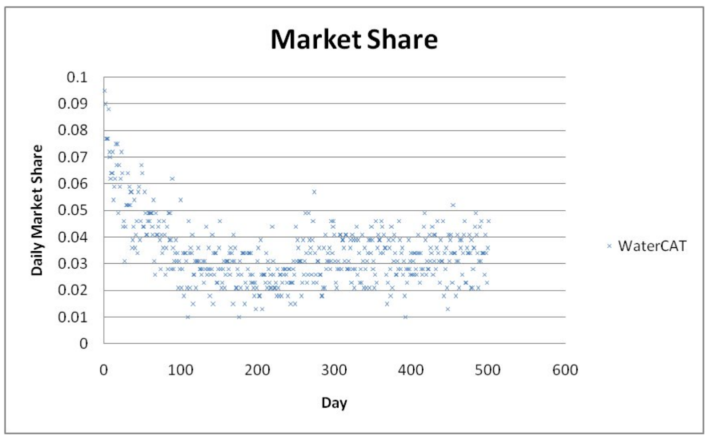

Figure 4.3: Market Share Distributions of WaterCAT

In order to implement the loyalty ranking of our matching policy, we created a new class called WaterCATShout. When an agent enters into the market, the system converts the Shout to the WaterCATShout object first. Then the market calculates the Bonus Factor of the agent according to the loyalty ranking and stores the value into the WaterCATShout object. In the old system, since the type of the shout was Object, we could keep the critical value of our matching policy during the interaction [1] with the server. After the platform change, however, we could

[1]There are several interaction processes between the server and the specialist before making the matched pairs.

not complete this process due to the casting of the Shout from an Object class to a specific Shout class. The end result was that we did not offer any promotive policies to the intra-marginal traders. Figure 4.3 is the Market Share distribution of WaterCAT in Game 1. The Market Share started dropping from the beginning of the game. Most of the agents that left us were intra-marginal traders. Because of the combination of our accepting policy and matching policy, WaterCAT can only match intra-marginal traders. As the number of intra-marginal traders kept decreasing, matching pairs became increasingly difficult for WaterCAT. Because of the discouraging policies we appeared to be offering, WaterCAT had a low position in the trading agent's preference ranking. On Day 38(Figure 4.4), we experienced the first zero TSR day. The TSR dropped from an average 89.47% of the previous days to zero. This caused our overall score dropped significantly. Since then, because of the small number of participating agents, WaterCAT kept getting zero TSR for most of the days.

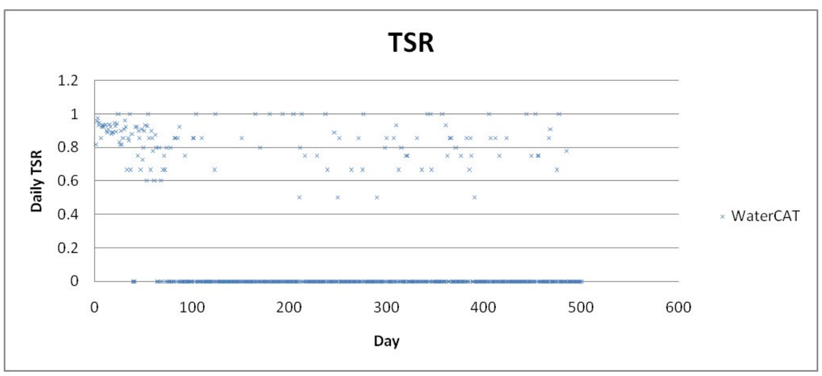

Figure 4.4: TSR Distributions of WaterCAT

Recall that strategy was to use the profit score to offset the decrease from the market share. This approach proved to be effective as we tested in the trial game.

But in the official game, since the market share dropped below the critical level[2], our market could not generate profits from the transactions. Thus, WaterCAT did not obtain the profit benefit as it did in the trial game. From Figure 4.5 we can see that the profit distribution stuck at a low level because of the zero TSR. Zero TSR means zero profit. In this situation, the profit obviously can not offset the decrease from the market share any more.

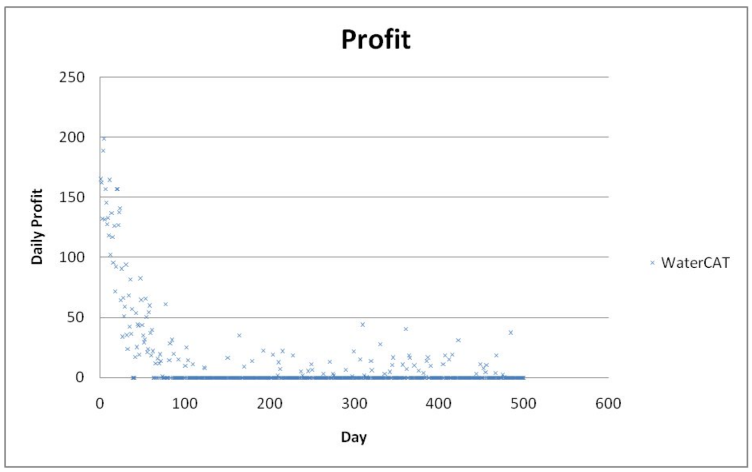

Figure 4.5: Profit Distributions of WaterCAT

We had a similar situation in the Game 2. Given that we were not aware of the disabling issue of shout type change, we did not change policies because we wanted to check whether what happened in Game 1 was a rare case or not. Figure 4.6 is the result distributions of all three scoring criteria of Game 2. They are similar with the ones of Game 1.

In Game 3 we changed the charging policy to free market at the beginning to attract traders and start charging profit fee after the early stage. During the

[2] The specialists can not match trading pairs when they do not have enough number of agents.

Specialist	Overall Score	Market Share Score	Profit Score	TSR Score
CUNY	241.574	47.147	213.503	464.04
IAMwildCAT	187.33	57.651	29.322	475.042
BazarganZebel	183.833	45.781	21.947	483.758
CrocodileAgent	170.69	33.264	28.4	450.391
Tianuani	165.887	73.477	0.075	424.123
rucat0	163.889	24.027	48.997	418.675
Jackaroo	161.43	32.825	33.041	418.464
cestlavie	144.526	49.977	45.602	338.026
PersianCAT	115.594	22.844	16.419	307.544
UMTac09	108.902	36.263	25.088	265.335
PSUCAT	101.621	20.657	9.317	274.926
TWBB	53.786	23.838	6.149	131.422
WaterCAT	49.296	17.194	14.021	116.744
Mertacor	42.057	15.122	8.135	102.939
MetroCAT	0	0	0	0
Cheshire	0	0	0	0
Lancashire	0	0	0	0

Table 4.2: Game 1 Score Summary

Specialist	Overall Score	Market Share Score	Profit Score	TSR Score
PSUCAT	219.624	63.937	126.494	468.414
Jackaroo	217.372	54.564	115.024	482.527
UMTac09	201.752	52.085	84.21	468.93
IAMwildCAT	181.15	51.146	19.914	472.43
Mertacor	168.68	75.604	36.138	394.295
TWBB	167.567	39.781	20.173	442.757
rucat0	153.72	18.116	54.789	388.243
CUNY	149.001	26.73	25.142	395.161
PersianCAT	68.522	21.507	1.536	182.559
cestlavie	56.975	27.945	7.984	135.018
WaterCAT	50.567	21.873	5.422	124.447
Tianuani	44.56	29.947	1.177	102.588
BazarganZebel	34.533	5.398	1.424	96.697
CrocodileAgent	31.083	11.344	0.597	81.185
MetroCAT	0	0	0	0
Cheshire	0	0	0	0
Lancashire	0	0	0	0

Table 4.3: Game 2 Score Summary

game, we also changed the matching policy from the WaterCAT matching to max-volume matching in order to generate more matching pairs. As we demonstrated in the accepting policy section, max-volume matching would not help increase TSR because of the drop of the Market Share. However, since our market share was already at a miserably low level, switching to the max-volume matching did help

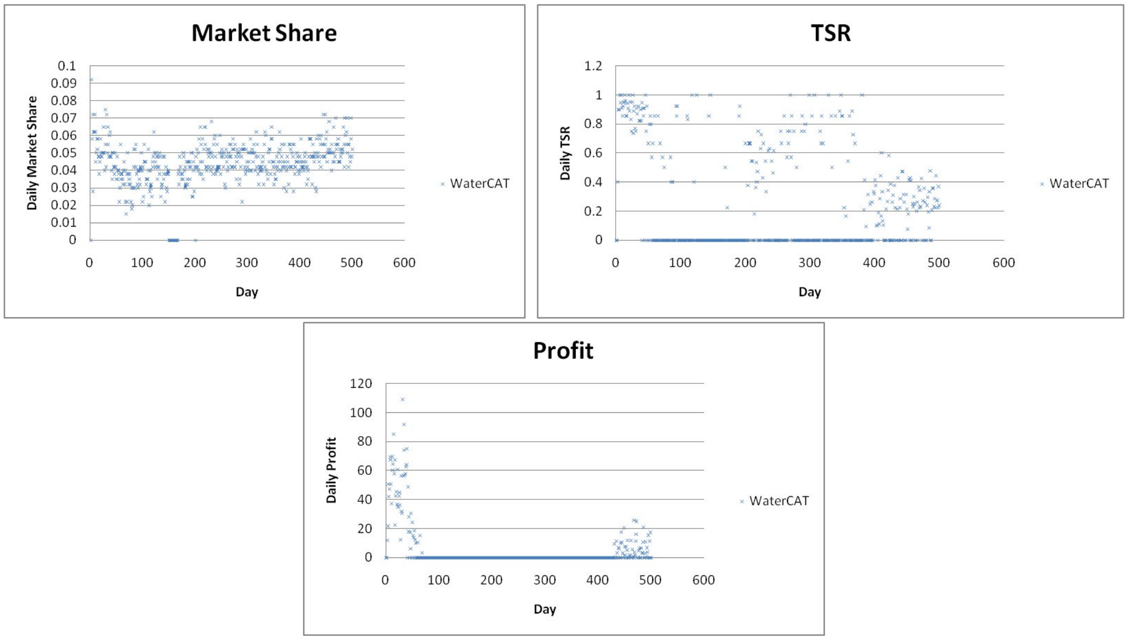

Figure 4.6: Market Share, TSR and Profit Distributions of WaterCAT in Game 2

us generate more matching pairs. Figure 4.7 shows that the TSR was better than the previous two days. However, market share is still the definitive factor to win the contest. Better market share generates better profit and TSR. Although we rearranged the policies to improve the market share during the game, because of the collapse of our matching policy, the final result was disappointing.

4.3 Competition Re-run Analysis

We experienced technical failure during the official games. In order to evaluate our specialist's performance in recognition of the shout class change, we conducted a number of experiments using the post-tournament version of specialists, found in the TAC agent repository. The specialists we used for the re-run analysis were:

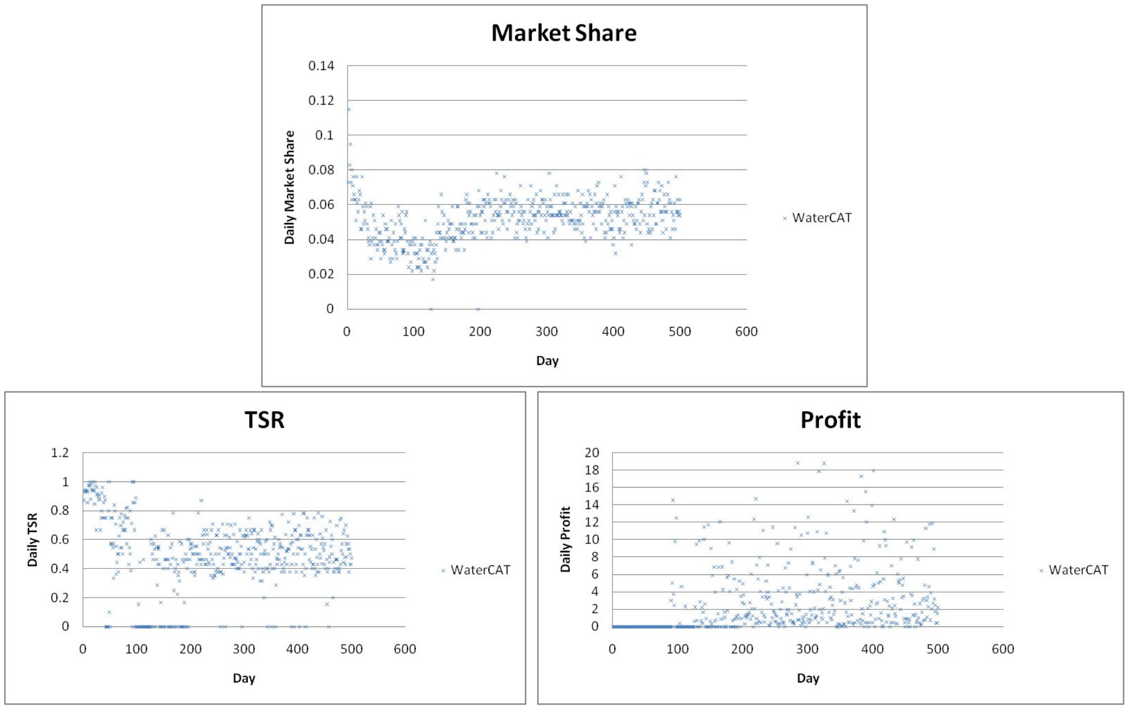

Figure 4.7: Market Share, TSR and Profit Distributions of WaterCAT in Game 3

Specialist	Overall Score	Market Share Score	Profit Score	TSR Score
PSUCAT	209.532	50.214	115.665	462.699
Mertacor	208.664	93.176	82.538	450.296
Jackaroo	200.413	41.013	81.184	479.062
cestlavie	191.539	50.626	51.952	472.049
IAMwildCAT	180.588	46.389	24.877	470.468
CUNY	175.707	28.805	62.478	435.834
PersianCAT	162.771	35.75	18.499	434.067
BazarganZebel	117.6	22.438	2.345	328.004
rucat0	109.072	17.272	27.188	282.748
UMTac09	97.619	25.574	10.702	256.567
TWBB	95.462	25.899	16.247	244.231
WaterCAT	91.387	25.747	3.259	245.205
Tianuani	69.666	25.628	1.796	181.559
CrocodileAgent	30.858	11.421	1.222	79.848
MetroCAT	0	0	0	0
Cheshire	0	0	0	0
Lancashire	0	0	0	0

Table 4.4: Game 3 Score Summary

IAMWildCAT, Jackaroo, CUNY, Mertacor and TWBB. We did not manage to include other specialists because of their unavailability at the time. We analyze the performance of our specialist against other competitors in two different durations of the game. We adopt a similar experimental setup as in the competition, with games running over 100 and 500 trading days and with 10 trading rounds each day.

4.3.1 Short Period Re-run Test

The top three players of the 100-day re-run were Jackaroo (51.427), WaterCAT (44.084), IAMwildCAT (37.684). Comparing with the official competition result that top-ranked Jackaroo, CUNY, IAMwildCAT, our specialist WaterCAT replaced CUNY and took the second place. Interestingly, CUNY had performance issues in the competition re-run and finished in last place.

Specialist	Overall Score	Market Share Score	Profit Score	TSR Score
Jackaroo	51.427	16.32	42.858	95.092
WaterCAT	44.084	17.275	24.591	90.395
IAMwildCAT	37.684	16.31	2.614	94.142
Mertacor	35.682	29.015	0	78.023
TWBB	33.687	9.685	15.084	76.301
CUNY	23.516	11.395	3.738	55.433

Table 4.5: Short Period Re-run Score Summary

Figure 4.8 shows the Market Share distribution of the re-run. WaterCAT's market share kept at a stable level and seemed competitive with other specialists. The winner of the market share was Mertacor because of the free market they offered. However, as discussed in the previous chapter, the profit gain will offset the loss from the market share and generate the winning margin for the fee-charging specialists. The fourth overall finish of Mertacor again demonstrates that winning the

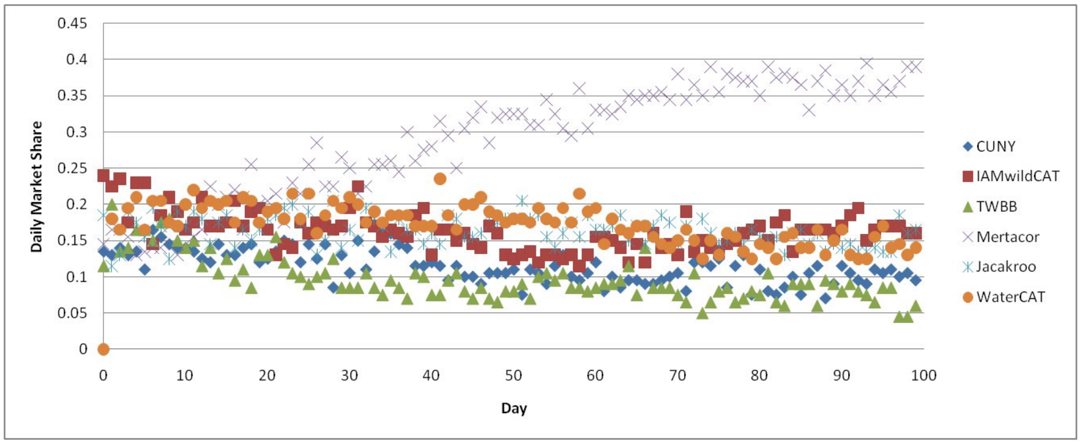

Figure 4.8: Market Share Distribution of Short Period Re-run

market share might not win the competition. The implication is that the specialist needs to construct the best combination between the market share and the profit.

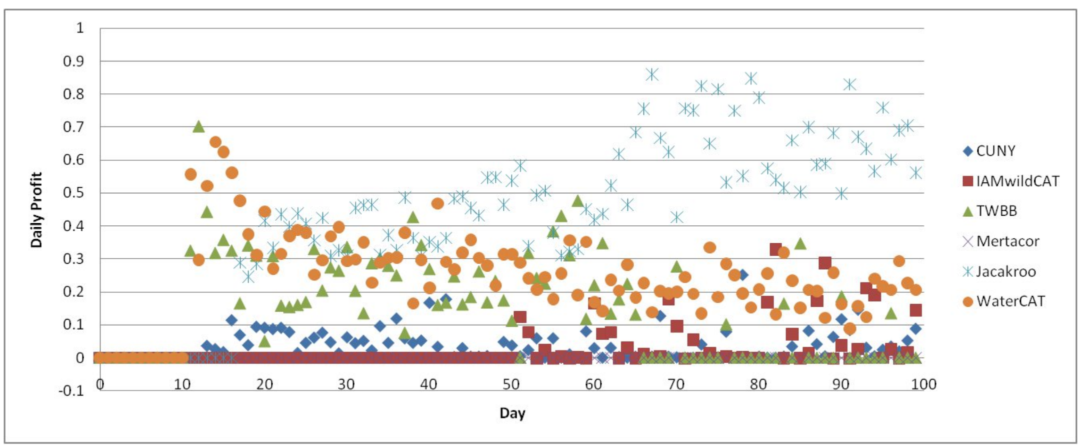

Figure 4.9: Profit Distribution of Short Period Re-run

Figure 4.9 is the Profit Distribution of the game. At the beginning of the game, WaterCAT offered free market in order to attract trading agents. After the initialization period, WaterCAT launched the 20% profit charging policy. Jackaroo turned out to be the winner of the Profit section. Because of their high value agents and better charging policy management, Jackaroo dominated the Profit score at the second half of the game. The Profit score of the second half provided the winning edge for the Jackaroo.

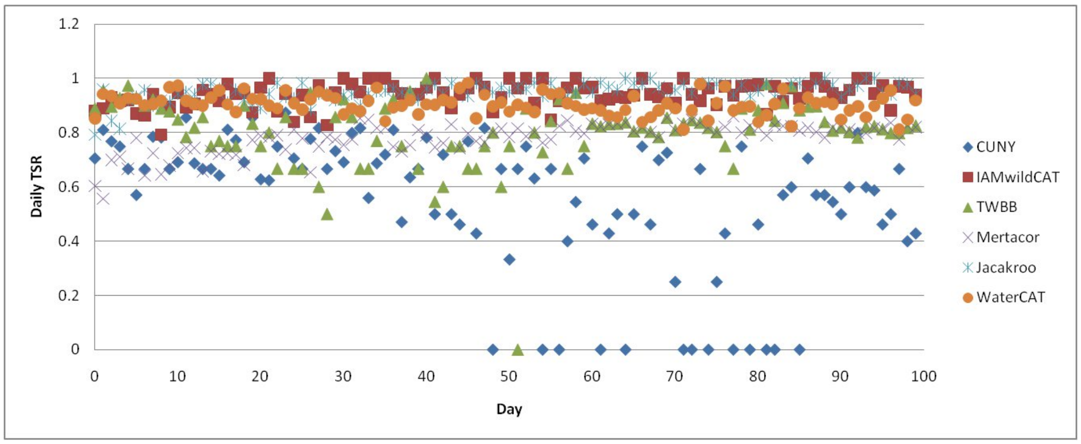

Figure 4.10: TSR Distribution of Short Period Re-run

The Transaction Success Rate score of WaterCAT stayed reasonably good throughout the whole game. It managed to accept a proper number of agents and clear them through the transactions. Figure 4.10 illustrates the TSR scores. Most of the specialists had good performance except CUNY. CUNY had similar result of what WaterCAT experienced in the official games, hitting a number of zero TSR at the second half of the game. This is the main reason why they were ranked last in the game.

4.3.2 Long Period Re-run Test

Surprisingly the top three finalists of the 500 trading days re-run were Mertacor (237.109), Jackaroo (220.406), and WaterCAT (201.876). Mertacor won the simulation thanks to its aggressive charging policy at the later stage of the game. It started charging fees from Day 122 and the extra profit offset the losses occurred in the early stage. It also managed to keep a good score on the market share even after installing a charging policy, which helped the specialist to achieve the best combination of market share and profit during the long period simulation. Thus, it emerged as the winner of the game.

Specialist	Overall Score	Market Share Score	Profit Score	TSR Score
Mertacor	237.109	148.623	152.266	410.472
Jackaroo	220.406	88.188	121.546	459.469
WaterCAT	201.876	65.53	105.619	434.794
IAMwildCAT	196.853	107.417	25.578	457.479
TWBB	157.546	44.308	22.25	406.079
CUNY	122.83	39.515	35.661	282.292

Table 4.6: Long Period Re-run Summary

Figure 4.11 shows the market share, profit and TSR distribution of the 500 day simulation. WaterCAT's overall performance appeared consistent in the long period test. As can be seen, its market share kept at a stable level all through the game. At the early stage, because of our comparatively aggressive charging policy, the market share started dropping on a minor scale. However, the decrease did not affect our overall performance. The profit distribution chart shows that WaterCAT's profit during that stage outperformed most of the other specialists. Our market share started increasing after day 250 when the other specialists exercised more aggressive charging policies. WaterCAT was able to maintain the performance in both categories at a constant level afterwards. Our TSR performance also remained satisfactory, thanks to the tight and dynamically adjusted accepting policy.

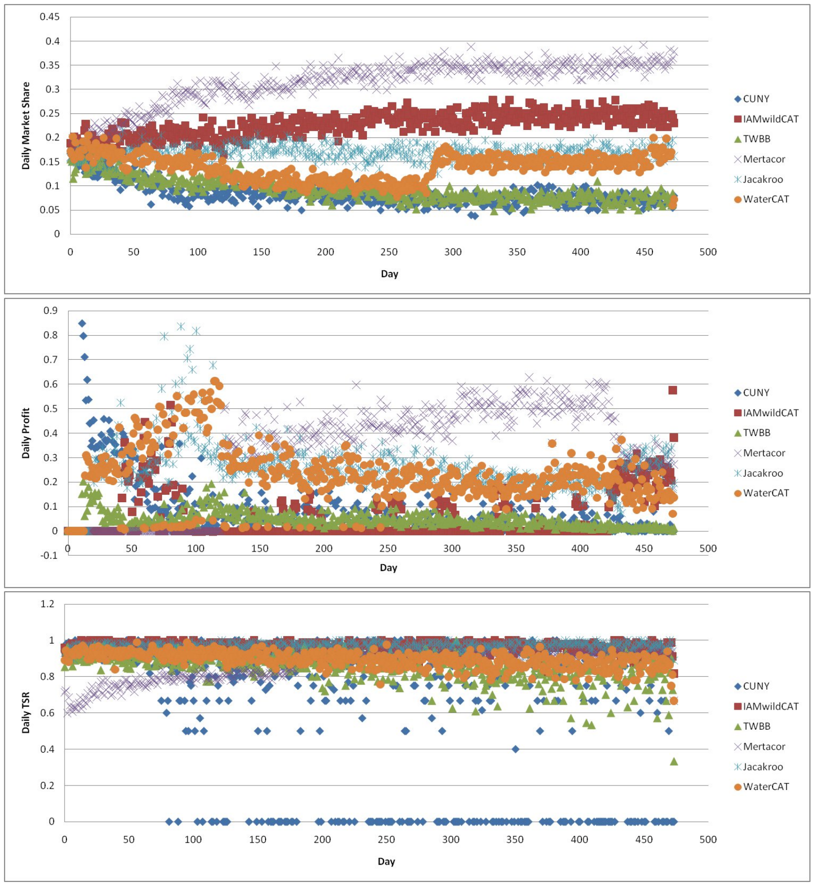

Figure 4.11: Market Share, Profit and TSR Distribution of the 500 Day Simulation

Overall, we are satisfied with the performance of WaterCAT in the re-run games. It ended up among the top three finishers in both short and long period re-run simulations, and maintained stable and competitive scores on Market Share and TSR. This can be attribute to the solid performance of WaterCAT Accepting and Matching policies. Nevertheless, Jackaroo had a better overall performance. WaterCAT was very competitive with it in terms of Market Share and TSR, but the winning margin of Jackaroo came from its profit-generating charging policy, while WaterCAT was equipped with a simple fixed charging policy. We tried different charging fee combinations, but we did not manage to beat Jackaroo using fixed charging policy in the simulations. However, we believe that with a better charging policy setup[3], WaterCAT has the potential to be the winning specialist.

4.4 Summary

In this chapter we analyzed the performance of WaterCAT in the trial and official games. We explained the winning reasons of the trial games and explored the technical failure in the official games. Due to a technical issue of classification, we did not have a successful performance in the official games. In order to evaluate our policy after fixing the technical malfunction, we conducted game re-run using post-tournament version agents. The result shows the solid performance of the policies we designed. WaterCAT achieved as one of the top three finishers in the re-run, indicating its potential to compete against the top players in the official game.

[3]Some dynamic charging policies are proposed in the future research section.

Chapter 5

Related Work

In this chapter we introduce literature related to our research and discuss our contributions to the field of trading market design. We first review the related literature on how to attract intra-marginal traders across the markets by using registration fees. Then we study the related work in generalization properties of CAT entries.

5.1 Attracting Intra-marginal Traders across Multiple Markets

The cornerstone strategy of winning the CAT game is to capture market share as much as possible. In particular, in order to win the game the specialist needs to attract those high-value and intra-marginal traders that can give the specialist a higher transaction success rate and generate more profit both for the trader and for the specialist. To become appealing for the trader, we intentionally excluded registration fees. Our assumption was that including registration fees would put off potentially profit-generating traders. However, this assumption may not hold in light of another body of research on trader behaviors. For instance, *Niu et al.* [15] reported that using registration fees has been shown to be a simple and effective way to attract intra-marginal traders. *Sohn et al.* [20] studied this approach more formally by using a game theory model. Their model is built on the trader market selection behavior to study why registration fees attract intra-marginal traders and drive extra-marginal traders away.

Two Intra-marginal Traders Case

Sohn and his colleagues' simulation starts with two free markets, M1 and M2. There are two trading agents, an intra-marginal buyer with a private value of 125 and an intra-marginal seller with a private value of 75. If they are matched, the transaction price is set to p=100, giving both the buyer and the seller a profit of 25. The case can be analyzed by a normal-form game model as shown in Table 5.1.

	Seller selects M1	Seller selects M2
Buyer selects M1	**(25,25)**	(0,0)
Buyer selects M2	(0,0)	**(25,25)**

Table 5.1: Normal-form market selection model, as reported by *Sohn et al.* (2009). Bold typeface denotes Nash Equilibrium

Apparently this is a battle-of-the-sexes game. The Nash Equilibria of the market selection strategy is (Buyer, Seller)=(M1,M1) or (M2,M2). As simple as this basic case is. *Sohn et al.* intended to verify that this battle-of-the-sexes framework can be extended into n-trader cases in general. They started by considering four trader cases with two buyers and two sellers.

Four Intra-marginal Traders Case

Consider the case that there two intra-marginal buyers B1(private value of 140) and buyer B2(private value of 120), and two intra-marginal sellers S1(private value of 60) and seller S2(private value of 80). The payoff matrix of the normal game form can be constructed in Table 5.2.

	S1 selects M1	S1 selects M2
B1 selects M1	**(40,40,20,20)**	(30,0,0,30)
B1 selects M2	(0,30,30,0)	**(40,40,20,20)**
B2 selects M1, S2 selects M1		
	S1 selects M1	S1 selects M2
B1 selects M1	(40,40,0,0)	(0,0,0,0)
B1 selects M2	(30,30,30,30)	(40,40,0,0)
B2 selects M1, S2 selects M2		
	S1 selects M1	S1 selects M2
B1 selects M1	(40,40,0,0)	(30,30,30,30)
B1 selects M2	(0,0,0,0)	(40,40,0,0)
B2 selects M2, S2 selects M1		
	S1 selects M1	S1 selects M2
B1 selects M1	**(40,40,20,20)**	(0,30,30,0)
B1 selects M2	(30,0,0,30)	**(40,40,20,20)**
B2 selects M2, S2 selects M2		

Table 5.2: Normal-form market selection model for four intra-marginal traders, as reported by *Sohn et al.* (2009). Bold typeface denotes Nash equilibrium

The Nash equilibria under four intra-marginal traders (B1,S1,B2,S2) are (M1,M1,M1,M1), (M1,M1,M2,M2), (M2,M2,M1,M1) and (M2,M2,M2,M2). In these NEs, the higher-valued buyer B1 and seller S1 tend to be together and so do the lower-valued B2 and S2.

Two Intra-marginal and Two Extra-marginal Traders Case

Let us consider another case when there are two intra-marginal traders and two extra-marginal traders. The private values of intra-marginal traders B1 and S1 are

140 and 60. The private values of the extra-marginal traders B2 and S2 are 90 and 110. The payoff matrix of the normal game form can be seen in Table 5.3.

	S1 selects M1	S1 selects M2
B1 selects M1	**(40,40,0,0)**	(15,0,0,15)
B1 selects M2	(0,15,15,0)	**(40,40,0,0)**
B2 selects M1, S2 selects M1		

	S1 selects M1	S1 selects M2
B1 selects M1	**(40,40,0,0)**	(0,0,0,0)
B1 selects M2	(15,15,15,15)	**(40,40,0,0)**
B2 selects M1, S2 selects M2		

	S1 selects M1	S1 selects M2
B1 selects M1	**(40,40,0,0)**	(15,15,15,15)
B1 selects M2	(0,0,0,0)	**(40,40,0,0)**
B2 selects M2, S2 selects M1		

	S1 selects M1	S1 selects M2
B1 selects M1	**(40,40,0,0)**	(0,15,15,0)
B1 selects M2	(15,0,0,15)	**(40,40,0,0)**
B2 selects M2, S2 selects M2		

Table 5.3: Normal-form market selection model for two intra-marginal traders and two extra-marginal traders, as reported by *Sohn et al.* (2009). Bold typeface denotes Nash equilibrium

The intra-marginal trader pair (B1,S1) selects either (M1,M1) or (M2,M2) and the extra-marginal traders B2 and S2 do not have preference to which market they are in. Thus, from the result of the NE it is possible to draw the conclusion that intra-marginal traders tend to stay in the same market in order to maximize their transaction profit.

Extending the Model into n-Trader Case

Sohn et al. extend the model into n-trader case by using brute-force NE search program. The program calculates the trader payoffs and search Nash equilibrium. The program results also confirm the NE results in Table 5.2 and 5.3 which are generated by hand calculation.

Incorporation of Registration Fee

Based on the arrangements of previous free markets, *Sohn et al.* demonstrated the traders' market selection behavior under several different registration values and their effect on the resultant NE. The authors showed that as the registration fee increases, the intra-marginal traders and extra-marginal traders are separated into different markets, and the extra-marginal traders tend to move to the free markets. When the registration fee reaches higher level, low-value intra-marginal traders leave the registration fee market. At the end, when the registration fee is too high for the traders to make profits, the remaining intra-marginal traders leave the market. The authors also indicated that the corresponding number of NE decreases as the value of registration fee increases.

Sohn and his colleagues worked in the direction of formally modeling the effects of market policy on trader market selection behavior. In the future, they plan to extend the model to more complex market environments. It is necessary to study the intertwined effects of different market policies and trader populations on Nash equilibrium. However, because of the dynamics of trader's market-selection behavior, *Sohn et al.* also point out that Nash equilibrium is not necessarily the best construct for the model. Concepts such as coalition-proof Nash equilibrium appear to be another more appropriate model for market-selection decisions.

Sohn et al's paper not only provides an example about how to explore the policy issue from a game theory perspective, it also sheds light on how to incorporate registration fees in our future study on policy design. While our empirical results indicated that imposing registration fees will lead to a decrease in market share, *Sohn et al.* suggested that registration fee could, at least theoretically, drive out the extra-marginal traders to improve overall specialist performance. An interesting topic for future research is how to balance the benefits and costs of including registration fees.

5.2 Empirical Evaluation of the Generalization Ability of the CAT Entries

While Sohn et al's work may help us to understand how trader behavior might exert impacts on the specialist performance, there are other factors we have left out of our current study. Specifically, we neglected the external factors across all markets, which are subject to the specifications of the organizer. These factors, however, may also influence the specialist's performance significantly.

Some researchers have made empirical attempts in this direction. For instance, *Robinson et al.* [17] presented a very interesting research about the generalization property of specialists in CAT competition. The objective of their research is to validate whether and how a specialist might favor certain trading strategies. In order to explore the generalization properties, *Robinson et al.* used the 2008 CAT tournament specialists to test the specialists' performance under different testing environments. The results suggested that the specialist can be affected by a number of factors, including trading agent population, presence of other specialists and the scoring period setup.

Robinson et al. first showed that some specialists' performances could be affected by the different mixes of trader type. The authors conducted a number of simulations using different distributions of trading strategies and the result revealed that some specialists' rankings changed under different environments. Robinson and his colleagues then carried out similar simulations with different selections of the specialists. Certain specialists' rankings changed when their competing specialists were changed. In addition, *Robinson et al.* showed even under the same environment, selecting different period as scoring period could also change specialists' rankings.

The findings of *Robinson et al.* are of particular interest to our study because we adopted a simpler view of the external environment faced by the specialist. In our study, all the factors discussed in Robinson et al were hold constant or left out of our design scope. But if these factors are indeed present and influential, we need to examine how our proposed policy modifications can hold true in a more complex market environment. Research in this direction seems to be fruitful as the market conditions in real practice are often more complicated than we assumed.

5.3 Summary

In this chapter we reviewed two interesting streams of research that are closely related to this thesis. The first is about how to attract intra-marginal traders across the markets by using the registration fee. The reviewed research represented the extant effort in the direction of formally modeling the effects of market policy on trader market selection behavior. The second line of research studies the generalization properties of CAT entries. The authors demonstrated that the specialists can be sensitive to trading strategies, other specialist's setup and the scoring period.

Chapter 6

Conclusion

In this thesis we describe the WaterCAT specialist, designed for the TAC Market Design game which is part of the International Trading Agent Competition. The objective of an agent in this competition is to effectively manage and operate a market that attracts traders to compete for resources in it. This market, in turn, competes against markets operated by other competition entrants and the aim is to maximize the market and profit share of the agent, as well as its transaction success rate. To do this, the agent needs to continually monitor and adapt, in response to the competing marketplaces, the rules it uses to accept offers, clear the market, price the transactions and charge the traders. Given this context, this thesis details WaterCAT strategic behavior and describes the techniques we developed.

In this chapter, we summarize the contributions of this thesis. We also describe some directions for future work.

6.1 Contributions

The main contributions of this thesis were:

- WaterCAT Accepting Policy - A new model of accepting policy with market monitor feature
- WaterCAT Matching Policy - A new model of matching policy with the loyalty ranking feature
- Analysis on different charging fees
- Applications on other CDAs

6.1.1 WaterCAT Accepting Policy

Based on the logic of equilibrium-beating approach, we present an accepting policy that can determine which shouts are accepted. We modified the static policy in use so that it can dynamically change the accepting level according to the market condition in real-time. This allows a wider range of agents entering the market with the consideration of maintaining a high Transaction Success Rate.

6.1.2 WaterCAT Matching Policy

We propose a new equilibrium matching policy that has a loyalty ranking feature. We also propose a new method, Bonus Factor, that aggregates the price ranking and loyalty ranking together. We show that our proposed new policy outperforms the original equilibrium matching policy in the simulation tests.

6.1.3 Analysis on Different Charging Fees

We analyze the trading agent's sensitivity towards the different charging fees in fixed charging policy. We find a winning number of profit charging fees. We also propose several adaptive charging policies.

6.1.4 Applications on other CDAs

Besides the specific purpose for the CAT competition, our research can also help the design of the competing market, especially in the area of the exchange market and other CDAs competitions. Our approaches are generic and are applicable to other models. With the demonstrated performance, other researchers can adapt our approaches to their models. Our research can also help the design of the marketing strategies. We can simulate the market activities by proper setup of the specialists and agents in the CAT platform. People can use the model to test the performance of new policies against other competitors and adjust the policies to achieve the best market reaction.

6.2 Directions for Future Work

In this section we outline some directions for future research based on the results of our study.

6.2.1 Adaptive Charging Policy Approach

We have discussed the fixed charging policy in detail. But the game we face is very competitive. The game is assessed by the overall performance of three criteria. We demonstrated that the specialist could get a better overall score by giving up a certain level of market share. The profit score not only offsets the decrease in market share, but also generates additional margin. This leads to new questions: what percentage of market share is the specialist willing to lose and how much profit the market is expecting to get? To answer this question, we need to build a market that can make decisions automatically according to the real-time market conditions. Although we need future research to simplify the complexity of the scenarios, we postulate some prototypes for the adaptive charging policy here.

The Adaptive Charging Policy consists of two stages. The first α rounds would be Stage 1 - free market. From the experiments we learned that at the beginning of the game, the agents keep switching markets due to their market selection strategy. During this initialization period the agents learn different policies from all assessable markets and they will make decisions based on their market selection functions. The specialist need to provide the most attractive policy to occupy a better position in the agents' market selection rankings. At this stage, the main objective is to increase market share.

After the initialization period, optimally the market should already have a good market share, or at least be at a very competitive level. In the next stage a sophisticated engine needs to generate charging policies according to the market conditions. The engine would make the decisions among increasing charges, decreasing charges and keeping current charges. We designed several engines to facilitate the decisions and possible approaches are listed in the following.

- Expectation-beating Approach: The engine compares the current market share with a target value that is either predefined or generated from the market. As long as the current market share surpasses the target level, the predefined charging policy will keep on posing charging fees. When the market share is below the target, however, the engine will disable the predefined charging policy. This approach is the easiest to implement but it is often impractical because it is very difficult to determine the target value. There are too many undetermined elements to generate a proper target value.

- Relative Progress Approach: We define α days as one session. After the initialization period, the engine activates the predefined charging policy and monitors the average market share of each session. If the average market share is more/less than the previous session, the engine increases/decreases the charging amount. We tested this approach against the fixed charging policy defined in the previous section and the result is not convincing. When the engine changes the charging amount by a small percentage, the result is very close to that of the fixed charging policy. If the engine changes the charging amount by a large scale, the fixed charging policy beats the relative progress approach, because adjusting charging policy in a large scale generates an unstable market environment and an unstable market generates a lower overall score. The overall score is affected mainly by the drop in the market share section.

- Random Sampling Approach: The engine generates some random charging samples with limited range and tests each sample for a certain period. The engine selects the best performance sample as the standard charging policy. The advantage of this approach is that it could generate the best performing policy dynamically according to the game entrants and setup. The disadvantage is that it is hard to normalize the sample results because of the dynamic environment, thus the sample results are biased.

Research on developing efficient charging policies is still at a stage of infancy. Although we have explained the approaches adopted in this thesis to comply with the CAT game specifications, future research should explore how other policy design strategies may produce better results in other settings.

6.3 Summary

In this thesis we studied the relationship between matching rules and market share in continuous double auctions. We provided a new matching policy based on the price ranking and the loyalty ranking criteria. We also provided a new accepting policy that can dynamically adjust the accepting level in response to real-time market situations. We use CAT as the test-bed to test the performance of our proposed policy modifications, and the results were mostly supportive. As an entrant of CAT tournament 2009, we analyzed the success and failure reasons and reported how the WaterCAT performance changed in the competition re-run. In the end we provided some adaptive charging policy approaches as future research directions.

Bibliography

[1] A. Andersson and F. Ygge. Managing large scale computational markets. In *31st Hawaiian International Conference on System Sciences*, pages 4–13, 1998.

[2] Dave Cliff. *Minimal-Intelligence Agents for Bargaining Behaviours in Market-Based Environments. Technical report*, June 1997.

[3] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation revisited. In *the Proc. of the Tenth International Conference on the World Wide Web (WWW10)*, pages 613–622, Hong Kong, 2001.

[4] I. Erev and A. E. Roth. Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. *American Economic Review*, 88(4):848–881, 1998.

[5] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *the Proc. of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 301–312, San Diego, 2003.

[6] D. Friedman. The double auction institution: A survey. *The Double Auction Market: Institutions, Theories and Evidence*, pages 3–25, 1993.

[7] D. Friedman and J. Rust, editors. *The Double Auction Market: Institutions, Theories and Evidence.* Addison-Wesley, New York, 1992.

[8] E. Gerding, P. McBurner, J. Niu, S. Parsons, and S. Phelps. *Overview of CAT: A market design competition. Technical report.* Dept. of Computer Science, Universit of Liverpool, Liverpool, UK, 2007.

[9] Steven Gjerstad and John Dickhaut. Price formation in double auctions. *Games and Economic Behaviour*, 22:1–29, 1998.

[10] D.K. Gode and S. Sunder. Allocative efficiency of markets with zero-intelligence traders: Market as a partial sustitute for individual rationality. *Journal of Political Economy*, 101(1):119–137, 1993.

[11] J. Gomoluch and M. Schroeder. Market-based resource allocation for grid computing: A model and simulation. In *First International Workshop on Middleware for Grid Computing*, pages 211–218, 2003.

[12] J. H. Kagel and W. Vogt. Buyers bid double auctions: Preliminary experimental results. *The Double Auction Market: Institutions, Theories and Evidence*, pages 285–305, 1993.

[13] John Kemeny. Mathematics without numbers. *Daedalus*, 88:577–591, 1959.

[14] N. Nisan, S. London, O. Regev, and N. Camiel. Globally distributed computation over the internet - the popcorn project. In *18th International Conference on Distributed Computing Systems*, 1998.

[15] J. Niu, K. Cai, E.Gerding, P. McBurney, and S. Parsons. Characterizing effective auction mechanisms: Insights from the 2007 tac market design competition. In *the Proc. of the 7th Int. Conf. on Autonomous Agents and Multiagent Systems*, 2008.

[16] J. Niu, K. Cai, S. Parsons, E. Gerding, P. McBurney, T.Moyaux, S. Phelps, and D. Shield. Jcat: A platform for the tac market design competition. In *the Proc. of the Seventh International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2008)*, Estoril, Portugal, 2008.

[17] E. Robinson, P. McBurney, and X. Yao. How specialised are specialists? generalisation properties of entries from the 2008 tac market design competition. In *WorkShop on Trading Agent Design and Analysis (TADA 2009)*, 2009.

[18] M.H. Rothkopf, A. Pekec, and R.M. Harstad. *Computationally manageable combinational auctions.* Management Science.

[19] D. Sculley. *Rank Aggregation for Similar Items.* Tufts University, Medford, MA, USA, 2007.

[20] J. Sohn, S. Lee, and T. Mullen. Attracting intra-marginal traders across multiple markets. In *WorkShop on Trading Agent Design and Analysis (TADA 2009)*, 2009.

[21] R. Sutton and A. Barto, editors. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 1998.

[22] P. Vytelingum. *The structure and behaviour of the Continuous Double Auction*. PhD thesis, School of ECS, Univ. of Southampton, 2008.

[23] B. Widrow and M. E. Hoff. Adaptive switching circuits. *IRE WESCON Conventions Record*, 4:96–104, 1960.

[24] P. R. Wurman, W. E. Walsh, and M. P. Wellman. Flexible double auctions for electronic commerce: theory and implementation. *Decision Support Syst.*, 24:17–27, 1998.

[25] H. P. Young and A. Levenglick. A consistent extension of condorcet's election principle. *SIAM Journal on Applied Mathematics*, 35(2):285–300, 1978.