Intelligent Trading Agents for Massively Multi-player Game Economies *

J. Reeder
School of EECS
University of Central Florida
Orlando, FL
jreeder@mail.ucf.edu

G. Sukthankar
School of EECS
University of Central Florida
Orlando, FL
gitars@eecs.ucf.edu

M. Georgiopoulos
School of EECS
University of Central Florida
Orlando, FL
michaelg@mail.ucf.edu

G. Anagnostopoulos
Department of ECE
Florida Institute of Technology
Melbourne, FL
georgio@fit.edu

Abstract

As massively multi-player gaming environments become more detailed, developing agents to populate these virtual worlds as capable non-player characters poses an increasingly complex problem. Human players in many games must achieve their objectives through financial skills such as trading and supply chain management as well as through combat and diplomacy. In this paper, we examine the problem of creating intelligent trading agents for virtual markets. Using historical data from EVE Online, a science-fiction based MMORPG, we evaluate several strategies for buying, selling, and supply chain management. We demonstrate that using reinforcement learning to determine policies based on the market microstructure gives trading agents a competitive advantage in amassing wealth. Imbuing agents with the ability to adapt their trading policies can make them more resistant to exploitation by other traders and capable of participating in virtual economies on an equal footing with humans.

Introduction

Managing virtual markets is a relatively new but important aspect of maintaining an immersive multi-player gaming environment. It is no longer enough to create a world with interesting strategic and tactical gameplay opportunities; many players want the additional option of achieving their goals through financial means such as buying, selling, investing, and supply chain management. Although the game designers can exert control over item scarcity, fees, and wealth generation mechanisms, much of the economy is dictated by the collective will of the players in the virtual markets. In some virtual worlds, the participants use the virtual world as an alternate means for advertising and selling real-world goods. For instance, Second Life, by Linden Labs, is inhabited by over 1.2 million regular visitors, performing $600,000 of business transactions per day (Kirkpatrick 2007). In other virtual worlds, such as World of Warcraft, the electronic marketplace is used as a mechanism for players to satisfy in-game needs such as equipment, food, or trade goods.

In this paper, we investigate the potential of introducing autonomous trading and supply chain management agents into these virtual financial ecosystems. The goal of the agent is to buy components, manufacture goods, and sell finished products while maximizing accrual of wealth in a fluctuating market. However, unlike the autonomous agents developed for AI competitions in trading agent and supply chain management (Greenwald & Stone 2001), we want these agents to cope with a marketplace created by a vast number of transactions by human players rather than markets dictated by financial models or transactions with other autonomous agents. Previous work on developing autonomous agents for game environments has focused on issues such as game replayability (van Lent *et al.* 2005) and agent variability (Wray & Laird 2003).

We selected the multi-player online game EVE Online, as a testbed for our trading agent. EVE Online, developed by CCP, has a very sophisticated player-controlled economy that is actively regulated by a professional economist who monitors inflation, deflation, commodity indicies, and production levels within the virtual world. At the end of 2007, there were 220,000 active subscribers and 460,000 player characters, trading billions of units every month (EVE Online 2007). Since strategic battle planning is heavily influenced by logistics and access to resources, supply chain management is an important operational problem for EVE players.

In this paper, we focus on the problem of creating agents with good "financial tactics" in buying, selling, and supply chain management. We evaluate several strategies and show that a reinforcement learning approach based on the market microstructure can give a trading agent a competitive advantage in amassing wealth over standard fixed policies. An additional consideration for developers of trading agents is protecting their agents from player exploitation; the introduction of easily duped trading agents in the virtual market would create an easy avenue for smart players to cheaply acquire rare items. We believe that imbuing agents with the ability to learn trading policies from recent historical data will make them potentially more resistant to predatory trading practices.

Related Work

Previous work on developing agents for simulated economies has centered around the Trading Agent Competition, which provides a standardized competitive benchmark

*This work was partially supported by NSF grants: 0341601, 0647018, 0717674, 0717680, 0647120, 0525429, 0203446

for the agents community. There are two different competition environments of interest:

TAC Classic: agents procure items in multiple simultaneous auctions for hotel reservations, flights, and event tickets while trying to maximize utility functions for a group of simulated clients. Agents compete against other autonomous agents bidding for items offered by simulated vendors in a variety of auction styles; certain items are directly traded between autonomous agents in a continuous double-side auctions. The agent that obtains the travel itineraries with the highest client utility wins the competition. The agents developed for this competition must be able to rapidly calculate the marginal utility of acquiring the items in the various auctions to determine good bidding strategies (Greenwald & Stone 2001). There has also been some work on the problem of learning pricelines predicting the future cost of market resources, based on bidding in previous rounds (Greenwald & Boyan 2004).

TAC Supply Chain Management: agents act as computer manufacturers in a market with simulated customers and part suppliers. Agents outbid other agents for customer orders, buy components from simulated supplier with fluctuating prices, and manage their factories production schedule (Pardoe & Stone 2006b). The agent that makes the most profit over the course of the competition wins. TacTex-06, the winner of TAC SCM 2006, used various prediction techniques for anticipating future supplier prices and the likelihood of client bid acceptance to improve the performance of its supply chain (Pardoe & Stone 2006a).

The supply chain management scenario faced by agents trading in EVE Online markets differs in several key ways from these competition scenarios. First the volume of units and number of transactions in the EVE Online markets is significantly larger than the number of trades executed in the competition settings. Moreover, there is a constant flow of traders entering and leaving the EVE markets, whereas each competition round occurs with a fixed group of agents and simulated vendors. Because of the large and open nature of the EVE markets, it is basically always possible to obtain commodities by paying a higher price, whereas the TAC agents cannot always obtain items even by paying a higher price.

EVE Online

EVE Online is a space-based MMORPG (massively multiplayer online role playing game) in which the players play Pod Pilots, immortal demi-gods capable of flying starships. The players affect the EVE universe through market trades, player vs. player (PVP) combat, and political maneuvering. Although some commodities are supplied by non-player character (NPC) agents, the EVE market is predominantly player-controlled. Ships, modules, ammunition, and drones are all built by player manufacturers; rare items are sold by players who obtained the items through NPC pirate hunting. Production in EVE is a multi-step process that takes as input raw materials and gives as output a module or ship. The first step of the process is the acquisition of raw minerals through mining. These materials can be mined by the player or purchased through the market from other player miners.

Once the player obtains the necessary minerals they can install a manufacturing job at a station using a blueprint. These blueprints are acquired from the market and are one of the few items offered by the NPC agents. The blueprint describes what materials are necessary and what item will be produced. With the blueprint and the minerals in hand, the player installs the job in a manufacturing slot at either an NPC station or a player-owned structure, and after a set build time the item is returned. Nearly all goods in the game are produced through this process. Technology Level 2 ships and modules follow a very similar process, with the addition of Tech 2 component materials that are required for the manufacturing process. These Tech 2 component materials are manufactured in much the same way as normal materials, but the base materials needed to make the components are acquired through a different production method.

In this paper, we examine the problem of Tech 2 component production. Tech 2 components require a four step process to manufacture. First the base materials are mined from a moon. This requires the use of a Player Owned Structure and can only be done in low security and null security space. Next the base materials are processed by a simple reactor into a simple material. Two or more simple materials are combined in an advanced reactor to produce an advanced material. Finally two or more advanced materials are used to produce the T2 component through the normal blueprint manufacturing process. It is not necessary for a single individual to handle every step of the process, and in most cases players will focus their attention on an individual part of the process, by buying the materials they need from the market and selling their product back to others doing the same thing.

We designed our agent to handle the purchasing of simple materials to fill an advanced reaction chain. A single advanced reactor can be configured to produce any of six different advanced materials without significant configuration changes. The agent must determine which of the six different materials would be most profitable to produce, based on the market value of the base materials and the final advanced material. The prices of these materials fluctuate because of other participants in the market also trying to choose the most profitable reaction. Each of the materials go through cycles of increased supply causing the prices to fall, then increased demand so the prices rise again. The ebb and flow of the market for these materials is a direct effect of the players involved switching between the available reactions.

Approach

The large number of players involved in the EVE market is one of the reasons we selected EVE as a testbed for our trading agent. Due to the high level of human participation, the market in EVE appears to exhibit strong similarities to real-world markets (Seller 2008). CCP hired an economist to study and report on the health of the market in a series of quarterly economic newsletters similar in scope to shareholder reports released by major companies. With this in mind, we considered trading and simulation approaches

that had been proven successful in real-world financial markets (Nevmyvaka 2005). Since the underlying assumptions behind the motivations of actors in real-world markets, we turned our attention to empirical models—how observable variables such as prices, volumes, and spreads affect prices.

According to the theory espoused in market microstructure, real-world markets are constantly engaged in the process of price discovery, in which the actual value of the item is revealed through repeated negotiations between the market agents. In an order-driven market, unlike the quote-driven market in TAC SCM, these interactions occur in the form of sell orders and buy orders placed by market participants. There are two basic types of orders that traders can make: **market** or **limit** orders. A **market** order transaction occurs immediately at the best price currently offered, whereas in a **limit** order a reservation price is specified for the transaction. In a buy limit order, the agent indicates a willingness to buy a certain volume at a fixed price (or lower); in a sell limit order, the agent will accept no price lower than the limit. All market transactions which can be satisfied immediately occur, leaving an order book of remaining orders, arranged by price, that are used to fulfill incoming market orders. The two main approaches used to analyze the relationship between sequences of these transactions and the success of future transactions are: (1) using time-series analysis to characterize the relationship between stochastic processes; (2) machine learning to learn optimal transaction policies. In this paper, we focus on the second approach—learning bidding policies from data of past transactions.

Simulating an EVE Market

The EVE market interface has the ability to record a snapshot of available orders for an item and export them to a file. This user interface function was used to collect four weeks of hourly data from the EVE market for every item related to our Tech 2 supply chain. Data collection was restricted to the Jita 4-4 market hub to simplify logistics. In this form, the data is useful for retrieving statistics about the prices of the items over time, but it is not useful for simulating the agent's interactions with the market. To simulate the agent's interactions with the live EVE market it is necessary to infer the transactions occurring between market snapshots by comparing successive snapshots and determining the difference between them.

- For orders that are on the market in both snapshots, the volumes are monitored and changes are attributed to market order interactions.
- When an order disappears between snapshots it is assumed that the order has been consumed and thus a market order is inferred.
- When a new order has appeared in the second snapshot it is assumed that a new limit order has been placed during the interval.

In this way a list of transactions for each snapshot are built and stored in the database. With these transactions it is now possible to simulate interactions with the market by building the order book at each time step and then simulating the following transactions. By placing an order into the order book prior to applying the transactions, it is possible to simulate that orders' performance in the market. Using our assumptions, we can reconstruct which transactions have occurred, but the order and timing of the simulated transactions can differ from the actual transactions, which cannot be extracted with the tools currently available in the EVE Online client. Based the length of time between successive snapshots, it is expected that some orders have been placed and consumed in the interval that are not reflected in the data collected, but this problem can be mitigated by increasing the data collection frequency.

Although it would be preferable to have direct access to the sequences of transactions, simulating the market in this way provides a close approximation of the true nature of the market in question. We chose this approach because it fits well with the data collected from the live market, and has shown good results as a basis for training a RL algorithm (Nevmyvaka 2005). It allows for an analysis of true order books from the market to be incorporated into the agent's decision process and for the evaluation of the simulated orders against real transactions. The main drawback of this method is that the simulated market does not react to the new order placed by the agent as a real market would. However, it is reasonable to assume that an autonomous agent that is continuously monitoring the market can place transactions to counter unexpected shifts in the market more rapidly than the typical human traders who interact with the market on a daily basis.

Trading and Production Scenario

We evaluated our trading agent on the amount of wealth amassed over the course of the scenario. To be successful, the agent must minimize the costs associated with purchasing the supplies that serve as the input for the manufacturing process and maximize the gains associated with the selling the product. Additionally, the agent uses market information as an input for identifying which manufacturing process is likely to result in the most profit.

When buying items from the market, the agent has the option of placing a market order to buy directly from the limit sellers or placing a limit buy order. When buying directly from the sellers the buyer is essentially paying a premium above the perceived price of the item for the privilege of immediate satisfaction. When placing a limit order the agent is providing liquidity for someone else and forcing them to pay the premium, but runs the risk of the buy order not being fulfilled. The agent's task is to identify the optimal policy specifying which order to place at every timestep.

The most relevant features affecting the agent's decision are:

Timesteps: timesteps remaining, ranging from $t - 1$ to 0 where t is the time horizon.

Volume: the volume left to be bought ranging from V to 0 where V is the initial volume.

Order Book Spread (OBS): size of the order book spread, $BestAsk - BestBid$. We discretize the order book

Table 1: Agent trading options (ISK: EVE Currency)

Agent Action	Description
0	Do Nothing
1	Bid 10% lower than the current Best Bid
2	Bid 1 ISK below the current Best Bid
3	Bid 1 ISK above the current Best Bid
4	Bid 10% higher than the current Best Bid
5	Place a Market Buy order

spread into $-1, 0, 1$ for low, average, and high, based on the average and standard deviation of past values.

Immediate Market Order (IMMO): the cost of placing an immediate market order.

Signed Volume (SV): This features denotes whether the buy volume is increasing (-1), remaining constant (0), or whether the sell volume is increasing(+1).

Learning Trading Policies

We use an offline version of the reinforcement learning algorithm described in (Nevmyvaka 2005) to learn a set of trading policies for the agent based on historical data. The trading policy describes the optimal sequence of transactions to buy or sell a particular item, assuming that the agent has previously decided which item to acquire. Based on the training data, we exhaustively search all possible actions and update our cost estimate of taking various actions using the following rule:

$$c(s,a) = \alpha c(s,a) + (1-\alpha)[c_{im}(s,a) + \arg\min_a p(s',a)],$$

where s is the initial state, a is the action taken, s' is the new state, $p(s',a)$ is the expected cost of an action in the new state, n is the number of times a has been tried in s, α is $\frac{n}{n+1}$, and c_{im} is the immediate cost of the action. The immediate cost c_{im} is calculated by placing the bid associated with the action and simulating the market until the next time step. If the order is fulfilled or partially fulfilled the cost is calculated as: $c_{im} = \frac{p-p_0}{p_0}$, where p is the average price per volume and p_0 is the initial perceived price. The perceived price is calculated based on the order book spread, $(BestAsk - BestBid)/2$.

As the state-action pairs are visited, their costs are updated according to the update rule; this process is repeated for all the combinations of t and v for all of the training data. The Q-table is updated in reverse order since logically at the final timestep the only option to successfully complete the task is to place a market order for all of the remaining volume (action #5). This state is a terminating state and its only costs are the immediate costs. Under the assumption that our state space is Markovian, we use dynamic programming to construct the Q-table. The only features that have an effect on the algorithm's run time are t and v. The other market-based feature variables are only affected by the sample data, hence it is possible to add more features without substantially increasing the amount of time required to learn the Q-table.

We make the assumption that our orders do not have a large enough effect on the market between timesteps to significantly change the state of the market. With this assumption we can determine the next state by simulating the results for each action and updating the remaining volume accordingly. In this situation the next state will be the state at $t+1$ and v volume, where v is the amount traded because of the action, and the market variables remain fixed for that point in time.

To learn a trading policy using our RL approach, buying and selling policies for each of the items being studied are trained on three weeks of live game data, with one week of live data held in reserve for testing purposes. Buying and selling are treated separately because the cost associated with each action is reversed. When selling an item it is desirable to sell above the perceived price, while when buying the reverse is true.

Results

To evaluate the effectiveness of the trading policies learned by the RL algorithm, we compared it to the following trading strategies: spread market order (SMO), upfront market order (UFMO), last minute market order (LMMO), and variable weight average pricing (VWAP). SMO, LMMO, UFMO, are all methods that place direct market orders, and differ in the timing of the orders. VWAP places both limit orders and market orders. At the beginning of the time step VWAP places a limit order slightly better than the current best limit order price, and at the end of the time step whatever volume is left over is placed in a market order. To compare the strategies the test data is divided up into five step epochs. For every item, each strategy is evaluated for every epoch in the test data. The average cost associated with each strategy is listed in Table 2. The costs are normalized against the perceived price of each item so that comparison between items is possible. Negative costs indicate that prices better than the perceived price were attained. Table 2 shows the RL performance of the best performing set of market variables and which market variables were used.

In all cases of the buying task, RL outperformed all other strategies for every combination of the market variables. However in the selling task there were some combinations of the market variables that were slightly outperformed by the VWAP strategy, but in every item the best market variables outperformed all other strategies.

It is also apparent from Table 2 that selling and buying are asymmetric and experience different levels of cost. In every item in the sell group the average cost of the RL and VWAP strategies is negative. This indicates that they both trade for better than the perceived price of the items. However, in the buy group only carbon polymers and hexite trade for better than perceived price. These differences can be attributed to the different styles of market behavior in the EVE market. In most situations EVE players who are buying are looking for immediate transactions while players who are selling items are willing to be patient and place limit orders. This leads to a large amount of market buy orders, a large amount of limit sale orders, and low amounts of limit buy orders and

market sell orders. Hence limit sale orders have a high probability of transacting. This is also the cause of the VWAP strategy performing much closer to the RL strategy on the sell task, because the limit orders placed by VWAP are consumed much more frequently than they are in the buy task.

In most cases the RL learns a policy of placing limit orders at slightly better than best limit price at all time steps but the last, and then placing a market order for the remaining volume. This strategy mimics a strategy commonly followed by many human participants in the EVE market. The RL also learns when to place orders inside the order book in order to reduce costs further. It learns this strategy more frequently in the sell case, as a result of the high volatility on the sell side of the order book. Figure 1 shows the actions chosen by the RL agent for titanium carbide using the OBS, and IMMO market variables. As shown the RL agent chooses to place a limit order just above or just below the BestAsk most of the time while choosing to place a market sell or to do nothing in just a few cases. The policies learned by the RL approach are effective in buying or selling in terms of the simulation techniques used, and should transition well to the live EVE market since they are based on data from the market.

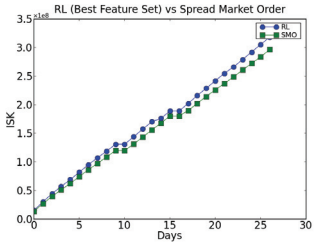
Supply Chain Management

In the supply chain scenario, a single factory is simulated over the course of four weeks. The trading agent must decide which of six reactions to run. Each of these reactions takes as input items from the buy group, and produce items from the sell group.

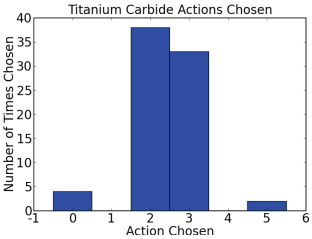
Some experiments were carried out to ascertain the usefulness of predicting which reaction will produce the most profit based on the market data. Greedy reaction choice was compared to an oracle with perfect knowledge of the future market prices to evaluate the potential worth of prediction techniques. For the three day cycle being simulated having the oracle's prediction only affected one decision over the course of a month. This result is expected, since the short cycle time leaves very little time for the prices of the items involved to shift significantly in relation to each other. We believe that over a longer period of time the oracle's choices would diverge more from the greedy choice and thus the profit acquired by the oracle would be greater. For the short cycle times studied here and the relatively short timeframe of the test data, prediction of the market does not give significant gains.

Discussion

There are several roles that autonomous trading agents can fulfill in MMORPG marketplaces. They can provide liquidity for human players in less active markets in the same way that NPC vendors serve as a reliable outlets for players to obtain and sell items at fixed prices. Also, introducing a population of trading agents can be a market control mechanism for the game designers to subtly manage markets and deflate prices. Even though trading agents in a virtual marketplace do not function as "adversaries" for the human players, it is equally important for them to be adaptive



(a)



(b)

Figure 1: (a) The total profit gained over the course of the scenario using the RL approach vs. SMO.(Greedy Selection) (b) The distribution of actions in the policy learned by the RL agent for selling titanium carbide.

and robust to player exploitation. Giving the agents multiple strategic options and enabling them to learn from market data has the potential of making them both adaptive and robust. There are two types of events that occur in EVE Online that cause large market shifts. Periodically the designers introduce game patches that change resource scarcity or modify the wealth generation mechanisms and cause dramatic shifts in the relative value of items. For instance, one game patch was the addition of "invention" into EVE as a mechanism for producing blueprint copies for manufacturing items. These blueprint copies (BPCs) enabled players to produce limited numbers of items at a reduced efficiency and deflated the artificially high prices caused by blueprint original (BPO) monopolies. Market shifts can also be initiated by players that decide to wage economic "war" by using their wealth to deny others profit rather than maximizing their own gain. By entering and systematically undercutting a market, a wealthy player can take consistent but manageable losses while completely denying others profit generation opportunities. This strategy can be as effective at crippling other players as destroying their manufacturing facilities. Frequent retraining on recent market data can increase the resiliency of a reinforcement-learning based trading agents to these externally instigated market shifts.

Conclusion and Future Work

The line between virtual and real-world economies is blurring. Economists routinely create computer simulations of real-world economies, and virtual MMORPG economies like the one in EVE Online are beginning to approach the size of real-world markets. Virtual worlds such as Second

Table 2: Average normalized cost for all buying and selling policies.

Item	OBS	IMB0	SV	RL (best)	VWAP	SMO	UFMO	LMMO
Buying								
Carbon Polymers	0	0	1	**-0.028504579**	0.082028063	0.14643748	0.158661913	0.133918673
Ceramic Powder	1	0	0	**0.002357998**	0.094899403	0.145539648	0.15691056	0.127830972
Crystallite Alloy	1	0	0	**0.159725478**	0.201720732	0.219629257	0.228338457	0.222763274
Fernite Alloy	1	0	1	**0.109048314**	0.145763227	0.165234565	0.16647955	0.167223286
Hexite	0	1	0	**-0.00674535**	0.052654967	0.102081265	0.102325669	0.096176695
Platinum Technite	1	0	0	**0.026604871**	0.053103663	0.066995723	0.065303677	0.066490438
Rolled Tungsten	0	0	1	**0.109001374**	0.213010068	0.250940359	0.261046177	0.248051276
Silicon Diborite	1	0	0	**0.14209394**	0.293296089	0.396878418	0.368555284	0.398358207
Sulfuric Acid	1	0	1	**0.035423585**	0.086367207	0.108323317	0.111466645	0.102229287
Titanium Chromide	0	0	1	**0.03438384**	0.066614538	0.080647685	0.0836483741	0.076908795
Selling								
Crystalline Carbonide	0	0	0	**-0.042070845**	-0.031703087	0.059422992	0.058534444	0.058171544
Fernite Carbide	1	0	0	**-0.022444448**	-0.008468892	0.037583333	0.03736483	0.039398718
Fullerides	0	0	0	**-0.025773631**	-0.021543334	0.032569338	0.029058182	0.03510464
Sylramic Fibers	0	0	0	**-0.03238247**	-0.030585387	0.04085	0.044264554	0.0448763
Titanium Carbide	1	1	0	**-0.030455756**	-0.020375989	0.041083696	0.04590129	0.035326763
Tungsten Carbide	1	0	0	**-0.03466989**	-0.003565254	0.05535917	0.055273444	0.054097885

Life and Entropia Online use currencies that can be traded for real-world dollars. Within EVE Online, players can use real-world money to buy virtual objects (time cards) that are traded in the EVE marketplace. These time cards can be used to offset the cost of game subscription fees so a player who is sufficiently financially successful can play EVE Online for free. Even purely virtual objects have some intrinsic real-world value based on the investment of real-world resources such as playing time and membership fees. Dedicated MMORPG players are willing to spend real-world money purchasing virtual items through eBay or other vendors to enhance their gaming experience.

As these virtual markets approach the complexity of real-world markets, developing agents to populate these virtual worlds as capable, autonomous, non-player characters poses a daunting research problem. In this paper, we have demonstrated some initial steps towards the problem of developing adaptive trading agents for inhabiting virtual marketplaces. Our reinforcement-learning approach for learning a trading policy from a short time-window of market data, outperforms standard trading policies such as VWAP (variable-width average price). Over the time intervals considered in this study, price prediction, even through the use of a market oracle, did not prove to be a decisive advantage for supply chain management, although we believe that in cases where the market is not exhibiting a stable downward trend it is likely that the use of prediction would outperform the greedy scheduling strategy.

In future work, we are interested in addressing the following problems. First we will evaluate the use of different function approximators as a replacement for our Q-table, since exactly calculating a Q-table for a large state space is intractable (Sutton & Barto 1998). Second, we are interested in expanding the agent's supply chain management options, including allowing the use of more complicated schedules and calculating logistics for moving materials between markets. Finally, we are interested in evaluating the performance and resiliency of our agent in real-time trading against human players.

References

EVE Online. 2007. Quarterly economic newsletter (Q4). Retrieved Apr 2008 `http://ccp.vo.llnwd.net/o2/pdf/QEN_Q4-2007.pdf`.

Greenwald, A., and Boyan, J. 2004. Bidding under uncertainty: Theory and experiments. In *Proceedings of Uncertainty in Artificial Intelligence*, 209–216.

Greenwald, A., and Stone, P. 2001. Autonomous bidding agents in the trading agent competition. *IEEE Internet Computing* 5(2).

Kirkpatrick, D. 2007. It's not a game. Retrieved July 2007 `http://money.cnn.com/magazines/fortune/fortune_archive/2007/02/05/8399120/`.

Nevmyvaka, Y. 2005. *Normative Appraoch to Market Microstructure Analysis*. Ph.D. Dissertation, Carnegie Mellon.

Pardoe, D., and Stone, P. 2006a. Predictive planning for supply chain management. In *International Conference Automated Planning and Scheduling*.

Pardoe, D., and Stone, P. 2006b. TacTex-2005: A champion supply chain management agent. In *Proceedings of National Conference on Artificial Intelligence*.

Seller, J. 2008. What can virtual-world economists tell us about real-world economies? Retrieved Apr 2008 `http://www.sciam.com/article.cfm?id=virtual-world-economists-on-real-economies`.

Sutton, R., and Barto, A. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

van Lent, M.; Riedl, M.; Carpenter, P.; MCalinden, R.; and Brobst, P. 2005. Increasing replayability with deliberative and reactive planning. In *Proceedings of the First Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Wray, R., and Laird, J. 2003. Variability in human behavior modeling for military simulations. In *Proceedings of Behavior Representation in Modeling and Simulation Conference (BRIMS)*.