Modelling a Supply Chain as a Network of Markets

Thierry Moyaux **Peter McBurney**

Department of Computer Science
University of Liverpool
Liverpool L69 7ZF, UK
{moyaux, peter}@csc.liv.ac.uk

ABSTRACT

The field of Agent-based Computational Economics (ACE) studies the macroeconomic regularities emerging from the local interactions of agents. In this paper, we describe a model of supply chains inspired by ACE. To this end, we present how we have connected sets of trading agents with markets so that any supply chain structure may be simulated. The base of our model is a representation of a supply chain as a network in which every edge represents a marketplace and every node a set of companies. The companies in every node buy in the same market, and sell in another market. The base of our implementation is the JAVA Auction Simulator API (JASA) which simulates a single auction. As an illustration of our model, we outline three possible studies based on results from the ACE literature, and preliminary results about the first of these studies are reported. The common objective of these three studies is to find new methods to control a complex system.
Keywords: Simulation of supply chains, Agent-based Computational Economics, Stabilisation, Bullwhip effect.

1. INTRODUCTION

Agent-based Computational Economics (ACE) can be defined as "the computational study of economies modeled as evolving systems of autonomous interacting agents" [Tesfatsion, 2002]. While this area focusses on systems with a flat structure, we wonder whether its tools may be useful to structured systems, such as supply chains. Such an application of ACE results to supply chains may be very fruitful because ACE provides formal models of agent systems with their emerging properties [Mizuta et al., 2003, Steiglitz et al., 1996, Steiglitz and Shapiro, 1998], and among these properties, system stability and efficiency are of great interest to supply chains. In particular, we hope that applying some ACE results will allow controlling a supply chain, in order, for example, to reduce a phenomenon called the bullwhip effect. This effect is an amplification of order variability in supply chains [Lee et al., 1997]. Conversely to previous work applying multiagent techniques to supply chains made of few agents (e.g. contracts and conventions [Cloutier et al., 2001], COOrdination Language [Fox et al., 2000], empirical game theory applied to the Trading Agent Competition [Wellman et al., 2005], etc.), using ACE methods require many agents. Since the JAVA Auction Simulator API (JASA) [Phelps, 2006] is a quick simulator of a single market, we use it for this reason as a basis of our supply chain simulator. Indeed, we have extended JASA so that any structure of supply chain may be represented and simulated. To do so, our supply chain simulator relies on a simple model which sees a supply chain as a network of auctions.

The rest of this paper is organized as follows. Section 2 presents our model of supply chains, and Section 3 how it is applied to our simulator. Then, Section 4 illustrates the capacities of our simulation with three examples of studies we plan to perform with it. Some preliminary results about the first study [Moyaux and McBurney, 2006] are also reported. Finally, Section 5 discusses our simulator with regard to these three planned studies.

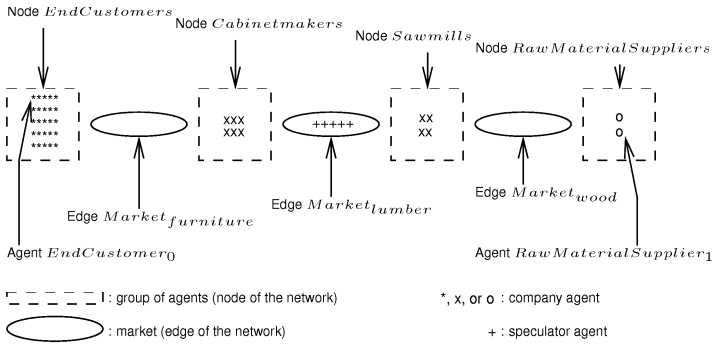

Fig. 1: Example of supply chain structure.

2. MODEL OF SUPPLY CHAINS

We model a supply chain as a network of inter-linked marketplaces. A single type of product is traded in each of these marketplaces which are implemented as double auctions. Figure 1 illustrates the notations used. As a network, a supply chain is represented as several sets of nodes (e.g. $EndCustomers$), each containing non-speculator agents (e.g. $EndCustomer_0$), and sets of edges (e.g. $Market_{furniture}$) linking these nodes. Companies can buy or sell in the markets represented by the edges linked to the considered node to which these companies belong. All the companies in a node are of the same type, but some parameters may differ (e.g. capacities, initial inventory levels, etc.), even if these parameters have to be drawn from the same random distribution. As can be seen in Figure 1, we consider four types of agents: EndCustomers (represented by *) can only buy, companies (pictured as x) buy in one market and sell in another, RawMaterialSuppliers (see o) can only sell, and speculators (represented as +) both buy and sell in the same market. As we shall see, we consider speculators because they may have a beneficial impact on a single market (increased stability and efficiency) [Steiglitz et al., 1996, Steiglitz and Shapiro, 1998], and Subsection 4.3 shall outline how this impact scales to a supply chain.

```
auction.n = 3

auction.0 = uk.ac.liv.supplyChain.SupplyChainRandomRobinAuction
auction.0.name = FURNITURE
auction.0.initialprice = 1
auction.0.closing = uk.ac.liv.auction.core.MaxRoundsAuctionClosingCondition
auction.0.closing.maximumrounds = 500
auction.0.console = true
auction.0.auctioneer = uk.ac.liv.supplyChain.ClearingHouseAuctioneer
auction.0.auctioneer.pricing = uk.ac.liv.supplyChain.SPricingPolicy
auction.0.auctioneer.pricing.k = 0.5
auction.0.report = uk.ac.liv.auction.stats.CombiAuctionReport
auction.0.report.n = 3
auction.0.report.0 = uk.ac.liv.auction.stats.PriceStatisticsReport
auction.0.report.1 = uk.ac.liv.supplyChain.MyGraphReport
auction.0.report.2 = uk.ac.liv.auction.stats.EquilibriumReport
(auction.1 and auction.2 are the same as auction.0, except their name)

agenttype.n = 6

agenttype.1 = uk.ac.liv.supplyChain.ManufacturerAgent
agenttype.1.numagents = 6 (We want 6 cabinetmakers)
agenttype.1.initialfunds = 555 (Cabinetmakers begin with 555)
agenttype.1.strategy = uk.ac.liv.supplyChain.SteiglitzStrategy
agenttype.1.strategy.margin = 3 (Always add 3 to ask shouts)
agenttype.1.valuer = uk.ac.liv.supplyChain.SteiglitzTraderValuer
agenttype.1.valuer.b00 = 2
agenttype.1.valuer.b01 = 4
agenttype.1.valuer.b0inf = 16
agenttype.1.source.n = 1
agenttype.1.source.0.initial = uk.ac.liv.supplyChain.RandomInt
agenttype.1.source.0.initial.minvalue = 15.0
agenttype.1.source.0.initial.maxvalue = 20.0
agenttype.1.source.0.capacity = 15 (Not used by any agenttype yet)
agenttype.1.source.0.buysInAuction = 1
agenttype.1.make.cost = 3
agenttype.1.make.capacity = 10
agenttype.1.make.speed = 1
agenttype.1.deliver.n = 1
agenttype.1.deliver.0.capacity = 15
agenttype.1.deliver.0.initial = uk.ac.liv.supplyChain.RandomInt
agenttype.1.deliver.0.initial.minvalue = 15.0
agenttype.1.deliver.0.initial.maxvalue = 20.0
agenttype.1.deliver.0.sellsInAuction = 0

(agenttype.0 is the same as agenttype.1, except for the following 11 lines:)
agenttype.0 = uk.ac.liv.supplyChain.EndCustomerAgent
agenttype.0.numagents = 25
agenttype.0.consumption = 1
agenttype.0.produceMoney = 5
agenttype.0.initialfunds = 60
agenttype.0.strategy.margin = 4 (Not useful to this agenttype)
agenttype.0.source.0.buysInAuction = 0
agenttype.0.make.cost = 3
agenttype.0.make.capacity = 1
agenttype.0.make.speed = 3
agenttype.0.deliver.n = 0
```

Fig. 2: Configuration file used in moyaux06ae – part 1/2.

```
(agenttype.2 is the same as agenttype.1, except for the following 3 lines:)
agenttype.2.numagents = 4
agenttype.2.source.0.buysInAuction = 2
agenttype.2.deliver.0.sellsInAuction = 1

(agenttype.3 is the same as agenttype.1, except for the following 6 lines:)
agenttype.3 = uk.ac.liv.supplyChain.RawMaterialSupplierAgent
agenttype.3.numagents = 2
agenttype.3.initialfunds = 60
agenttype.3.strategy.margin = 1
agenttype.3.source.n = 0
agenttype.3.deliver.0.sellsInAuction = 2

agenttype.4 = uk.ac.liv.supplyChain.SpeculatorAgent
agenttype.4.numagents = 5
agenttype.4.initialfunds = 60
agenttype.4.strategy = uk.ac.liv.supplyChain.SpeculatorStrategy
agenttype.4.strategy.activationdate = 0
agenttype.4.strategy.margin = uk.ac.liv.supplyChain.SplitEqually
agenttype.4.strategy.margin.numintervals = 4   (The first speculator has
agenttype.4.strategy.margin.minvalue = 0.0      margin=0, the second has
agenttype.4.strategy.margin.maxvalue = 0.5      margin=(0.5-0.0)/4, etc.)
agenttype.4.valuer = uk.ac.liv.supplyChain.AVGPriceForecastValuer
agenttype.4.valuer.smoothing = 0.008
agenttype.4.source.n = 1
agenttype.4.source.0.capacity = 10
agenttype.4.source.0.initial = uk.ac.liv.supplyChain.RandomInt
agenttype.4.source.0.initial.minvalue = 15.0
agenttype.4.source.0.initial.maxvalue = 20.0
agenttype.4.source.0.buysInAuction = 1
agenttype.4.source.0.sellsInAuction = 1
agenttype.4.make.cost = 3
agenttype.4.make.capacity = 10
agenttype.4.make.speed = 1
agenttype.4.deliver.n = 0

(agenttype.5 is the same as agenttype.4, except for the following 3 lines:)
agenttype.5.numagents = 0 (None of these speculators are created)
agenttype.5.source.0.buysInAuction = 2
agenttype.5.source.0.sellsInAuction = 2

prng = uk.ac.liv.prng.MT32
seed = 2704 (Initialise the PRNG – pseudo-random number generator)
log4j.rootCategory=INFO, stdout
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%m%n
```

Fig. 3: Configuration file used in moyaux06ae – part 2/2.

3. SIMULATION OF SUPPLY CHAINS

We now see more technical details, that is, how our model is implemented as an extension to the open source JAVA Auction Simulator API (JASA) [Phelps, 2006]. More precisely, our extension allows JASA to run several marketplaces in parallel. The overview of the JASA configuration file in Figures 2 and 3 illustrates how the structure of the supply chain in Figure 1 may be set. Lines starting with 'auction' define the parameters of marketplaces, and lines beginning with 'agenttype' contain agent parameters. After one of these two words, either the letter 'n' defines the number of auction edges or agent nodes to create (conversely to agenttype.x.numagents which indicates how many agents are in the node x), or a number x indicates which market edge or agent node is defined. The next two subsections detail Figures 2 and 3 in order to explain what are the additional parameters after such a number x.

3.1. Auctions

JASA provides several types of auctions (ascending auction, double auction, etc.) to represent each of the marketplaces. We currently use a double auction (an instance of the JAVA class SupplyChainRandomRobin) for every marketplace which is the same as in JASA, except that its 'run()' func-

tion executes a single auction round, while the JASA version executes as many auction rounds as are indicated in the line auction.x.closing.maximumrounds in Figure 2. Notice that x=0, 1 or 2 because three marketplaces are defined in the first line of this figure. What JAVA class to use for Auction x=0 is specified in the second line of Figure 2, and the next thirteen lines define this marketplace, e.g. the third line of Figure 2 sets the name of this auction to 'FURNITURE' (in accordance with Figure 1). Two other auctions similar to Auction 0 are next created, but not represented here, and this third line is the only one which is different between Auction 0, and Auctions 1 and 2.

The order in which auctions are created is important because, at each round, our modified JASA first runs Auction 0, next Auction 1, etc. Then, in the next round, Auction 1 is first run, etc. As a consequence, all markets between EndCustomers and CabinetMakers should be declared first, next all markets between CabinetMakers and Sawmills, etc. As in JASA, transportations are included in these auction rounds, that is, a round corresponds to (i) the placement of ask and bid shouts by sellers and buyers, (ii) the calculation and broadcast of the clearing price by the auctioneer of the auction, and finally (iii) the transportation (currently supposed instantaneous) and payment of the agreed exchanges.

3.2. Agents

We now focus on the nodes of our model. Most of what is presented here also applies to speculators which will be described in Subsection 4.2 since we plan to use them only in our first study. Three concepts about our agents should be defined here:

- *An agent type* corresponds to the internal structure of an agent (e.g. our company agents may trade in several marketplaces and have several inventories) and/or the behaviour with regard to the moment at which the considered agent is allowed to place a shout (e.g. JASA comes with two types of agents designed to bid either anytime or in only one of the several rounds in a day[1]).
- *A valuation function* calculates the value of the good exchanged in a marketplace. This is the maximum price a buyer may bid (thus, pay) or the minimum price a seller may bid (thus, be paid). As we shall see, the valuation function of our agents is the same as in [Mizuta et al., 2003, Steiglitz et al., 1996, Steiglitz and Shapiro, 1998], rather than constant randomly-generated values as in JASA.
- *A bidding strategy* defines how an agent places a bid (when its type allows to do so) so that either a buyer tries to pay less than its valuation of the good, or a seller tries to get more money than its valuation. More precisely, an agent bids strategically with regard to its belief about the value of a good, and this value is calculated by the valuation function.

Agent type: Figure 4 presents how the representation of our companies complies with the first level of the model SCOR from the Supply Chain Council [Supply Chain Council, 2001], that is, companies are made of three functions: (i) deliver is made of inventories of finished products (one inventory per market in which products are sold), a valuation function, and a bidding strategy

[1]Further details of these trader types can be found in [Phelps, 2006].

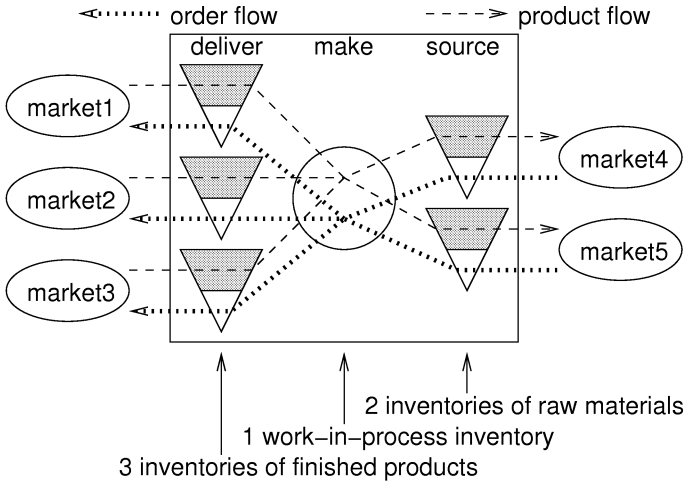

Fig. 4: General model of a company.

(note that each inventory holds products to be sold in a different auction, and the bidding function is the same for all these auctions), (ii) make is the (unique) work-in-process inventory in which batches of items have to spend the production time before moving to the finished product inventory, and (iii) source is similar to deliver, except that its inventories contain raw materials and its function places ask shouts to buy the different kinds of products. For the sake of simplicity, the valuation functions and the bidding strategies used by deliver and source are the same.

The basic class of all non-speculative agents is uk.ac.liv.supplyChain.ManufacturerAgent. Such agents always buy goods in at least one marketplace, transform these goods, and sell them in at least another marketplace. The class uk.ac.liv.supplyChain.RawMaterialSupplierAgent is a simplification on ManufacturerAgent because its source has no inventories. It is also slightly different because these agents do not transform goods but create them from nowhere. Finally, the class EndCustomerAgent is the opposite to RawMaterialSupplierAgent, that is, these agents consume goods instead of creating them, and create money.

Valuation function: The class uk.ac.liv.supplyChain.SteiglitzTraderValuer is the exact implementation of Steiglitz and his collegues' valuation function $B(\bar{f},\bar{g})$ [Steiglitz et al., 1996, Steiglitz and Shapiro, 1998], where $\bar{f} = f/r$ is the level f of the considered inventory normalized by its target level r (we set r to be the value of f at the beginning of a simulation, according to how we understand the parameters in [Steiglitz and Shapiro, 1998, p.43]), and $\bar{g} = g/(P.r)$ is the funds owned by the agent normalized by the value of the target inventory level (P is the clearing price of the auction in which the considered inventory trades). As a result, the valuation $B(\bar{f},\bar{g})$ of the product to be traded depends on the internal state of the agent. $B(\bar{f},\bar{g})$ is around 1 and is multiplied by the previous price P in order to calculate how much to bid (i.e. $B > 1$ means the agent places a buy shout BP at a price greater than P). Technically, $B(\bar{f},\bar{g})$ is [Steiglitz et al., 1996, Steiglitz and Shapiro, 1998]:

$$B(\bar{f},\bar{g}) = [b_{0\infty} - (b_{0\infty} - b_{00})]^{(1-\bar{f})} e^{\gamma \bar{g}} \qquad (1)$$

where b_{00}, b_{01} and $b_{0\infty}$ are constants defined in Figures 2 and 3 (cf. agenttype.x.valuer.b00, b01 and b0inf), and γ is also the constant:

$$\gamma = \ln\left(\frac{b_{0\infty} - b_{00}}{b_{0\infty} - b_{01}}\right) \quad (2)$$

Bidding strategy: The class uk.ac.liv.supplyChain.SteiglitzStrategy is a truth-telling strategy adapted to supply chains. This adaptation to supply chains matches the current auction with the inventory trading in that auction. In comparison with the strategies which can be found in the literature this strategy does not try to pay less or sell for more than the valuation, that is, this strategy bids the exact value calculated by uk.ac.liv.supplyChain.SteiglitzTraderValuer.

Normally, the bidding strategy is somewhat more complex when it places ask shouts rather than buy shouts because products have to be sold at a higher price than what they were bought plus production cost, which is managed by adding a $margin$ (cf. agenttype.1.strategy.margin = 3 in Figure 2) to the normal output of Steiglitz and his colleagues' strategy when a ask shout is placed. In other words, a buy shout is equal to $B(\bar{f}, \bar{g}) \times P$, while an ask shout is $B(\bar{f}, \bar{g}) \times P + margin$.

4. EXAMPLES OF STUDIES WITH OUR MODEL

As aforementioned, we designed our model in order to check how results from ACE scale to supply chains in order to control such a complex system. In a first phase, we plan to study how speculators may reduce the bullwhip effect. When this first approach is completed, we are going to address again the bullwhip effect by modifying the auctions so that their respective auctioneer calculates a clearing price stabilising the supply chain. Finally, in the longer term, we will focus on inter-supply chain competition (while the previous two steps focus on inter-company competition during auctions). Since Subsections 4.2 and 4.4 both address the bullwhip effect, we first present that phenomenon.

4.1. The bullwhip effect

The bullwhip effect is an amplification of demand variability along a supply chain. In other words, the CabinetMakers in Figure 1 place orders more variable than the demand received from EndCustomers. Next, Sawmills receive CabinetMakers orders and place orders to RawMaterialSuppliers which are even more variable, etc [Lee et al., 1997]. Besides increased variability, the bullwhip effect also makes demand signal less predictable. Both variability and unpredictability of demand cost money to a supply chain, because they *reduce supply chain agility* and *increase the inventory levels* of companies.

4.2. Use of speculation to reduce the bullwhip effect

Our first approach of reduction of the bullwhip effect is inspired by a work by Steiglitz *et al.* [Steiglitz et al., 1996, Steiglitz and Shapiro, 1998]. In fact, they proposed a model of a simple economy with endogeneous instability. Specifically, their agents exchange gold for food, consume one unit of food per day, and produce gold and food. Production skills of gold and food are different from each other, specific to individuals and constant over a simulation. Simulations show that such a simple model leads to large price fluctuations. Next, these authors added speculators which are of two types called DER (for derivative) and AVG (for average):

- *DER Speculators* estimate the second derivative of the price slope. If we call $P(t-1)$ the price in previous period $t-1$, these speculators post an ask shout $P(t-1).(1+margin)$ when the slope of the price is increasing, and a bid shout $P(t-1).(1-margin)$ when the slope is decreasing.
- *AVG Speculators* use a moving average to update their forecast of the price of the product. If we call P the actual price and $\hat{P}$ the forecasted price, then the forecast of the current price is $\hat{P}(t) = \beta.\hat{P}(t-1)+(1-\beta).P(t-1)$ for some coefficient $\beta \in (0,1)$. Then, a ask shout $P(t-1).(1+margin)$ is placed when $P(t-1) < \hat{P}.(1-margin)$, and a bid shout $P(t-1).(1-margin)$ when $P(t-1) > \hat{P}.(1+margin)$.

All speculators own the same small non-negative amount of money at the beginning. Steiglitz and his colleagues found that adding these two kinds of speculators stabilizes the price of the only considered good and this stabilization "results in an overall increase in market efficiency and fluidity, in the sense that individual production decisions are more closely matched to skill, and the numeraire is more easily converted into accumulated wealth" [Steiglitz et al., 1996, p. 3].

However, a later paper written by Steiglitz and two other coworkers [Mizuta et al., 2003] shows that these results are not as good as first thought, because price bubbles may still occur. Moreover, speculation is often seen as a destabilizing factors of markets, which made Kaldor [Kaldor, 1960, p. 34] write that speculation likely exerts neither a price-stabilising influence, nor the opposite, or rather that it is both simulatenously.

Despite the drawback pointed out in [Mizuta et al., 2003], the results of [Steiglitz et al., 1996] remain very convincing, which is the reason why we would like to transfer them to supply chains. Besides that, we do not want to address questions as general as Kaldor, that is, our goal is not to study when speculation stabilises a market. On the contrary, our first goal is to quantify the impacts of the previous two types of speculation on the bullwhip effect. Finally, a second step here will be to compare different types of speculation (financial vs. stock) and study the potential benefits of futures and options contracts in supply chains.

4.3. First results about the use of speculators

This subsection only outlines the first results we have obtained with our simulator when studying the impact of speculators. Full details are in another paper [Moyaux and McBurney, 2006]. We show the supply chain dynamics without speculators in Subfigures 4.3, 4.3 and 4.3 (Figures 2 and 3 present the configuration file used here, except that the line "agenttype.4.numagents = 5" should be modified), then with five AVG speculators trading in the second (lumber) auction in Subfigures 4.3, 4.3 and 4.3 (Figures 2 and 3 present the exact configuration file used).

First of all, both Figure 5 and our paper [Moyaux and McBurney, 2006] do not deal with the bullwhip effect itself, but with its consequence on the price in the three auctions of the considered supply chain. Specifically, we can see that, instead of an amplification

of demand variability, there is an amplification of price variability: the price in Subfigure 4.3 is more stable than in Subfigure 4.3, which is itself less variable than in Subfigure 4.3. The consequence of the bullwhip effect on price thus occurs in our simulator. In other word, our simulator already implements a link between the three markets, that is, the product and demand streams of the supply chain create a relationship among its markets. In fact, adding speculators in the lumber market seems to both stabilise this market – cf. Subfigure 4.3 vs. 4.3 – and the wood market – cf. Subfigure 4.3 vs. 4.3.

Unfortunately, we cannot draw any conclusion about the effect of speculation on price stability yet because only two simulation runs are presented here. If we want to draw more general results about the considered model, we should (i) run many more instances of the simulation when the random parameters are initialised differently by making the seed of the random number generator vary (e.g. from "seed = 0" to "seed = 2704" in Figure 3), and (ii) change all the other (constant and randomly initialised) parameters. In this case, we may draw conclusions about the (de-)stabilising effect of speculators.

4.4. Coordination of auctioneers to reduce the bullwhip effect

We have already mentioned the paper [Mizuta et al., 2003] which presents a modification of the model in [Steiglitz et al., 1996, Steiglitz and Shapiro, 1998] outlined in Subsection 4.2. In [Mizuta et al., 2003], stabilization is no longer obtained by the addition of speculators, but by letting the auctionneer broadcast more information to the agents. Specifically, every agent i bids a price p_i for an amount a_i, and this decision is based on one of the three following price signals, instead of the clearing price P:

1. *Unweighted average of bids*:

$$P_0 = \frac{1}{n} \sum_i p_i \tag{3}$$

2. *Average of bids weighted by quantity*:

$$P_1 = \frac{1}{\sum_i a_i} \sum_i a_i p_i \tag{4}$$

Simulations show that P_1 has a destabilizing effect with regard to P_0. The authors explain this destabilization is due to the fact that "agents bidding for large quantities are generally farther from their desired reserves, and their bids are therefore farther from equilibrium – farther above for buyers who have a severe deficiency, and farther below for sellers who have a severe surplus" [Mizuta et al., 2003, p. 245]. As a consequence, they modified the price signal in the opposite direction (c is a scaling parameter setting the extent of inverse weighting):

3. *Average of bids weighted by the inverse of quantity*:

$$P_2 = \frac{1}{\sum_i 1/(c + a_i)} \sum_i \frac{1}{c + a_i} p_i \tag{5}$$

As in Subsection 4.2, the basic idea here is to apply these results to supply chains in order to measure their impact on the bullwhip effect. That is, the auctioneers in the supply chain would calculate the clearing price P_2, and we could study how to choose its parameter c. Further research would seek to coordinate (e.g. with some reinforcement learning algorithm) the value of c in the different markets so that the team of auctionners would become a decentralized coordinator.

4.5. Competition of different types of supply chains

This last phase is planned in the longer term in order to compare different kinds of supply chains, and in particular, to check how well a decentralized supply chain performs in comparison with a centralized one. What we call 'decentralized' refers to how decisions are made. In the decentralized case, every agent decides on its own how she behaves in every auction, which is what occurs in rest of this section. Since agents are selfish, an issue such as that of the prisonner's dilemma may occur (or, equivalently, the tragedy of the commons or the problem of the free rider). On the contrary, in the centralized case (e.g. when the supply chain is integrated, or when an actor is big enough to command the other companies), all company goals are aligned on the maximization of supply chain profit. As a consequence, a centralized supply chain could serve as a benchmark to a decentralized one, in which companies maximize their individual profit but may also make trade-offs to improve group profit.

Of course, nothing needs to be modified for the decentralized supply chain, we may just use again what will have been implemented in the two other studies. For the centralized supply chain, CabinetMakers will be connected to the furniture market for selling to EndCustomers. Next, the other companies will be implemented with the same JAVA classes as in the decentralized supply chain, except that their behaviour will not be controlled by their strategy but by a central coordinator. The goal of this central coordinator is its CabinetMakers be more efficient than the CabinetMakers of the decentralized supply chain.

We do not develop further these general questions about supply chain management, because the goal of this section is only to show how our simulator is able to address such questions. We now discuss this simulation tool.

5. DISCUSSION

First, according to a literature review about the bullwhip effect [Moyaux et al., 2006], neither speculation nor coordination achieved by auctioneers have been proposed as solutions to this effect before. As a consequence, the studies outlined in Subsections 4.2 and 4.4 may allow our simulator to explore new area in supply chain management.

Next, we have a remark about the relative complexity of ACE and supply chain models. The agents programmed by Steiglitz and his colleagues [Mizuta et al., 2003, Steiglitz et al., 1996, Steiglitz and Shapiro, 1998] are characterized by only three parameters, namely b_{00}, b_{01} and $b_{0\infty}$. On the contrary, agenttype.1 in Figure 2 uses fourteen parameters to describe each of its six agents. Moreover, the two inventory capacities in this agent node are not used yet, we have not included transportation lead time yet (probably implemented as a company in future work), and simulation outcomes may well depend on the supply chain structure. As a consequence, a supply chain simulation is necessarily less general than a single market, and many more experiments are necessary to modify parameters in order to generalize

results. Therefore, using a quick simulation platform such as JASA is an advantage.

6. CONCLUSION

This paper has described an adaptation of an auction simulator in which any supply chain structure may be represented thanks to the representation of a supply chain as a network of auctions. Three possible studies illustrated the capabilities of our simulator: (i) how speculation may be beneficial to the entire system, (ii) how the auctioneers may help the system be more efficient, and (iii) how the obtained system would compare with a central coordinator using optimization tools. These three studies were sketched, and some preliminary results about the first study were outlined. Of course, as future work, we are going to carry out these three studies and complete the programming required[2].

REFERENCES

[Cloutier et al., 2001] Cloutier, L., Frayret, J.-M., D'Amours, S., Espinasse, B., and Montreuil, B. (2001). A commitment-oriented framework for networked manufacturing coordination. *International Journal of Computer Integrated Manufacturing*, 14(6):522–534.

[Fox et al., 2000] Fox, M. S., Barbuceanu, M., and Teigen, R. (2000). Agent-oriented supply-chain management. *Int. J. of Flexible Manufacturing Systems*, 12(2/3):165–188.

[Kaldor, 1960] Kaldor, N. (1960). *Essays on Economic Stability and Growth*, chapter 1. Speculation and economic stability, pages 17–58. G. Duckworth, London, UK.

[Lee et al., 1997] Lee, H. L., Padmanabhan, V., and Whang, S. (1997). Information distortion in a supply chain: The bullwhip effect. *Management Science*, 43(4):546–558.

[Mizuta et al., 2003] Mizuta, H., Steiglitz, K., and Lirov, E. (2003). Effects of price signal choices on market stability. *Journal of Economic Behavior and Organization*, 52:235–251.

[Moyaux et al., 2006] Moyaux, T., Chaib-draa, B., and D'Amours, S. (2006). Information sharing as a coordination mechanism for reducing the bullwhip effect in a supply chain. *IEEE Trans. on Systems, Man, and Cybernetics*. (to appear).

[Moyaux and McBurney, 2006] Moyaux, T. and McBurney, P. (2006). Reduction of the bullwhip effect in supply chains through speculation. In *Proc. Symp. Artificial Economics 2006, Lecture Notes in Economics and Mathematical Systems (Springer)*, Aalborg (Denmark).

[Phelps, 2006] Phelps, S. (2006). Web site for JASA (Java Auction Simulator API). `http://www.csc.liv.ac.uk/~sphelps/jasa/` (accessed February 23, 2006).

[Steiglitz et al., 1996] Steiglitz, K., Honig, M. L., and Cohen, L. M. (1996). A computational market model based on individual action. In Clearwater, S. H., editor, *Market Based Control*, pages 1–27. World Scientific: Singapore.

[Steiglitz and Shapiro, 1998] Steiglitz, K. and Shapiro, D. (1998). Simulating the madness of crowds: Price bubbles in an auction-mediated robot market. *Computational Economics*, 12:35–59.

[Supply Chain Council, 2001] Supply Chain Council (2001). Overview of the SCOR model v5.0. http://www.-supplychainworld.org/WebCast/SCOR50_overview.ppt (accessed December 2001).

[Tesfatsion, 2002] Tesfatsion, L. (2002). Agent-based computational economics: Growing economies from the bottom up. *Artificial Life*, 8:55–82.

[Wellman et al., 2005] Wellman, M. P., Estelle, J., Singh, S., Vorobeychik, Y., Kiekintveld, C., and Soni, V. (2005). Strategic interactions in a supply chain game. *Computational Intelligence*, 21(1):1–26.

[2]This research was undertaken as part of the EPSRC funded project on Market-Based Control (GR/T10664/01).

[Furniture market without speculators]

[Lumber market without speculators]

[Wood market without speculators]

[Furniture market with 5 lumber speculators]

[Lumber market with 5 lumber speculator]

[Wood market with 5 lumber speculators]