Co-Evolution of Bilateral Negotiation Strategies in an Agent-Based Supply Web Environment

C. Russ
Dacos Software GmbH
Science Park 2
D-66123 Saarbrücken
+49 681 394671-44
christian.russ@dacos.com

A. Walz
Graduate School of Advanced Manufacturing
Breitscheidstr. 2c
D-70174 Stuttgart
+49 681 390031
alexander.walz@GSaME.eu

ABSTRACT

In this paper, we describe an evolutionary learning approach for the co-evolution of bilateral price negotiation strategies in an agent-based supply network environment. The effects of the adaptation processes of the intelligent agents are simulated and evaluated by using the multiagent supply chain simulation framework MACSIMA. This framework allows the design of large-scale supply network topologies consisting of a multitude of autonomous agents representing the companies in the supply network. MACSIMA supports a fine-tuning of the parameterization of the learning mechanism of each individual agent in a simulation scenario. Additionally, it enables the agents to exchange information about finished negotiations with other cooperating agents. In this way, the evolution of an agent's negotiation strategy is not only guided by his own experience but can also take the experience of other agents into account. We outline some evaluation results with a particular focus on the emergence of niche strategies within a group of cooperating agents at one tier of a supply chain for computer manufacturing.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Experimental design, Time series analysis, H.3.4 [**Systems and Software**]: Distributed systems, I.2 [**Artificial Intelligence**], I.2.6 [**Learning**]: Parameter learning I.2.11 [**Distributed Artificial Intelligence**]: Coherence and coordination, Intelligent agents, Multiagent systems, I.6 [**Simulation and Modeling**], I.6.6 [**Simulation Output Analysis**], I.6.7 [**Simulation Support Systems**]: Environments, I.6.8 [**Types of Simulation**]: Continuous, Distributed, J. [**Computer Applications**], J.4 [**Social and Behavioral Sciences**]: economics.

General Terms

Algorithms, Management, Design, Economics, Experimentation

Keywords

Learning intelligent agents, agent-based supply chains, supply networks, coordination mechanism design, bilateral negotiation, evolutionary adaptation, genetic algorithms, simulation

1. INTRODUCTION

A *supply chain* is a chain of possibly autonomous business entities that collectively procure, manufacture and distribute certain products. Since today's markets are highly dynamic, e.g. because of the high and frequent variations of customer demands, current supply chains are forced to respond to consumer demands more accurately, flexibly and quickly than ever before.

In order to stay competitive, supply chain partner companies are forced to form supply chains on the basis of more flexible and co-operative partnerships. For these reasons, so-called *supply webs* (see Laseter [4] and Porter [5] - i.e. B2B-enabled dynamic networks of supply chain units - will replace today's static supply chains to an increasing extent.

Higher dynamics and the widely distributed processes inherent in supply webs may result in an increase of expenses and delays, instead of higher efficiency and flexibility, if there are no coordination mechanisms, e.g. specialized auction or negotiation mechanisms instantiated that help to co-ordinate the local activities of the supply chain partners. Furthermore, the co-ordination and supply activities in a supply web, as well as their interrelations, can get too complex for humans to handle efficiently. In order to cope with the dynamics in supply webs, we suggest encapsulating operative supply chain units within agents.

But this approach can result in efficient supply processes only if the supply chain agents interacting within the instantiated coordination mechanisms possess the ability to adapt to varying events and circumstances fast and flexibly. For this reason, we suggest providing the supply chain agents with the ability to take part in bilateral negotiations, as well as with elaborated learning capabilities, enabling them to learn from negotiation successes and failures in order to adapt their negotiation strategy iteratively.

We will describe the application scenario and the agentification of the supply web units in the next paragraph, while we present their negotiating and learning capabilities in section 3 and section 4.

2. APPLICATION SCENARIO

Agents offer the advantage of being able to automatically and flexibly react to changes in the environment since they can autonomously perform tasks on behalf of their users.

Since supply processes are jeopardized by many external factors, the consumer demand for finished goods as well as the individual demand of the supply chain partners for resources and semi-finished goods varies dynamically. Moreover, in the case of B2B-enabled dynamic supply networks, the selection of partner companies producing and supplying needed resources and semi-finished goods is not fixed in advance but is carried out over electronic markets according to the current state of demand. This means that due to their dependence on the changing environment, the members as well as the topology of the supply network are constantly evolving. Hence, a flexible and robust coordination between the supply chain entities can only be reached if they are able to adapt themselves to varying circumstances and partner companies quickly and efficiently.

In this context, the agent-based encapsulation of supply chain units opens up the possibility to automate the selection of partner companies as well as the establishment of supply contracts between them. Both can be automatically managed by intelligent agents even without the interaction of human users. As a consequence, such an agent-based approach should on the one hand reduce transaction costs within the supply network and on the other hand increase the global welfare for the entire network, resulting in higher profits for all or at least some of its members. Whereby it can be assumed that the profit division within such networks will be significantly influenced by the elaborateness of the agents' learning capabilities.

To examine the dynamics within a supply network built up by adaptively negotiating agents, we have instantiated MACSIMA (**M**ulti **A**gent Supply **C**hain **SIM**ul**A**tion Framework). MACSIMA has been implemented in Java and offers a set of generic supply chain agent types that are instantiated for building up the different tiers of a supply network scenario to be simulated. The generic agent types consist in

1. ***resource or supplier agents (R_i)*** that supply raw materials to the network and possess a reserve price that marks a minimum price threshold for the whole system.
2. ***producer agents (P_i)*** that stand in the middle of the value chain and buy *raw materials* or *semi-finished goods* from resource agents or other producer agents as input goods to their *production process*. This process is managed by an individual *production function* that specifies the necessary *input goods*, the *output good(s)*, *production time* and *production cost* whereby these parameters may differ from agent to agent. Output goods are offered to the other supply agents for purchase. ***Trader agents (T_i)*** and ***deposit agents (D_i)*** can be derived as subtypes of the generic producer agent by adjusting the production function appropriately.
3. ***consumer agents (C_i)*** that stand at the end of the added value network and buy products from the producers. They cannot run out of money, but however, have a *consumption function* that specifies their maximal willingness to pay, i.e. a ceiling price, in their negotiations as well as a maximal number of end products that they demand.
4. ***good agents (G_i)*** that keep an account of the number of successful (*deals*) or unsuccessful (*rejects*) price negotiations concerning a specific good together with the negotiated prices etc. These agents are not necessary for conducting simulation runs but simplify the statistical evaluation of simulation results.

In MACSIMA, simulation scenarios can be defined and parameterized in a very detailed way as well as with a high degree of freedom, so that the user is totally free to decide on the number of tiers and the number of interacting agents on each tier.

In our simulation runs conducted so far we have mainly concentrated on different instantiations of a five-tier-supply-network for computer manufacturing, as sketched in figure 1. We usually run scenarios with 25 to 100 agents on an Intel Quadcore-architecture with 3 GB RAM and 32bit Windows Vista as operating system.

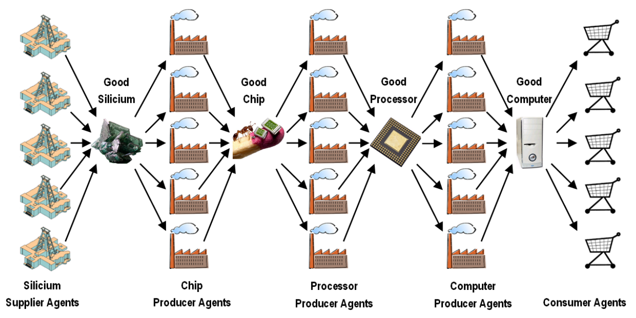

Figure 1: Instantiated supply network for computer manufacturing

As described, MACSIMA makes it possible to define not only such simple scenarios but also significantly larger ones with very complex and voluminous topology graphs by combining generic supply web building blocks as shown in the following figure.

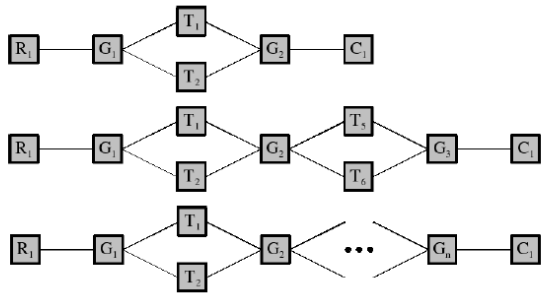

Figure 2: An example for supply network building blocks

Therefore, MACSIMA offers a graphical user interface that not only simplifies the definition of topologies but also enables one to parameterize the learning capabilities of <u>each</u> agent that is instantiated for a simulation run in detail as described in section 4. The instantiated agents find each other by a directory service that administrates all the agents within a simulation scenario with their *unique names*, *addresses*, *agent type and properties, i.e. capabilities*, *offered and demanded goods* etc.

Supply agents use this directory to find other agents with possibly intersecting goals, i.e. who offer desired goods or who are looking for desired goods. To find out if a mutually beneficial transaction can be carried out, each agent may select another agent in the network for starting a bilateral negotiation. The used negotiation protocol is a kind of strategic choice protocol that is also adopted by humans in economic real-life negotiations and is described in detail in the following section.

3. NEGOTIATION PROTOCOL

Since the internal price ideas of an autonomous agent are not visible to another agent in advance, the agents have to gather information about the "room to negotiate", i.e. the negotiation space, while they are negotiating with each other. In MACSIMA an agent can start no more than two parallel negotiations concerning the <u>same</u> product (not for conceptual reasons but for reducing the computational complexity of the simulation run).

3.1 NEGOTIATION ACTS

In the bilateral negotiation process, all agents are provided with the same action alternatives that are used by humans in negotiation situations. According to Pruitt [6], human agents show specific behavior in economic scenarios when they are in an alternating negotiation where two parties have to think over their positions and make proposals for a conflict solution. He formalizes this behavior in his ***Strategic Choice model*** and states that in every negotiation round humans select from among five basic strategies, namely

1. ***unilateral concession***: the party at turn decreases unilaterally the distance between the conflicting positions,
2. ***competitive behavior***: the party at turn remains with his position and tries to argue the other party out of its position by pressure, arguments, threats, etc.,
3. ***coordinative behavior***: both parties collaborate and try to dissolve the controversy,
4. ***idleness***: the party at turn does not continue the negotiation and makes no counteroffer, or
5. ***demolition of the negotiation***: the party at turn withdraws unilaterally from the negotiation and communicates this.

Eymann [2] states that these basic building blocks of human negotiation strategy can be further formalized, i.e. reduced, to three negotiation action alternatives sufficient for negotiations in multi-agent domains. Therefore, MACSIMA-agents are only equipped with the following ***negotiation actions***:

1. ***Accept***: the price offered by the other agent is accepted and the transaction is conducted. The buyer pays the negotiated price to the seller and receives the product.
2. ***Propose***: the agent at turn does not agree to the price proposal and makes a new proposal on his part. This new proposal can correspond with his own last proposal, if the agent is not willing to make a concession in this round of negotiation. Otherwise he calculates a new price proposal for his counteroffer.
3. ***Reject***: the agent breaks off the negotiation and an arrangement cannot be reached anymore. The agents thus have to search for other negotiation partners to fulfill their needs.

3.2 STRATEGY PARAMETERS

How humans decide on their next negotiation actions depends on their individual negotiation strategy. Thus, Pruitt [6] further introduces the terms ***demand level*** and ***concession rate***.

The *demand level* is the utility increase, i.e. economic profit or cost reduction, a negotiator would realize by accepting the current price proposal of his opponent, e.g. the profit a seller makes by accepting the price proposal of a potential buyer.

The *concession rate* defines the speed in which a negotiator is willing to make concessions, i.e. to reduce the price distance between his last price proposal and the current counter-offer of his negotiation partner.

For modeling complex and not easily predictable strategic behaviour in automated negotiations we use six interrelated strategy parameters that determine the negotiation strategy of an agent. These six parameters are stored in a so-called ***genotype***, a data structure suitable for processing by a genetic algorithm.

In general, an agent possesses several genotypes that can be evolutionarily optimized and adapted to varying negotiation partners and market conditions as described in section 4. The genotypes of an agent consist of six value parameters with values lying, all in the interval [0;1]:

1. acquisitiveness
2. delta_change
3. delta_jump
4. satisfaction
5. weight_memory
6. reputation

Numerous of these genotypes are stored in a genetic pool containing the genotypes employed in negotiations. Futhermore, miscellaneous plumages, i.e. combinations of genotypes with ascertained ***fitness*** values are stored in a data structure called population.

Each parameter of a defined genotype influences (non-deterministically, i.e. within a probabilistic variance) an agent's individual negotiation behaviour as described in the following.

The negotiation strategy of the agents is implemented as a finite state machine where these parameters on the one hand define the probabilities of changing from one of the negotiation states (accept, propose, reject) to another and on the other hand define how quickly, how often and to which extent concessions are made etc.

The ***acquisitiveness*** of an agent defines the probability that he will offer a unilateral concession on his next "move", i.e. as the seller lowering his asking price towards the price proposal of the buyer, which means that the agent will change the price in the next step. Therefore, this parameter taking on a value of 1 means it would prohibit an agent from making any price concessions; while a value of 0 would motivate him to concede in each negotiation step.

The ***delta_change*** parameter defines the step size of a monetary concession, i.e. by specifying a percent value by which the price distance between one's own price proposal and the price proposal of the negotiation partner is reduced. The size of the concession step arises in percentage from the *difference of the asked and offered price*. This keeps the negotiation mechanism of the attribute ***symmetric*** since neither sellers nor buyers are handled in a preferred manner (for the definition of attributes of a coordination mechanism see [3]).

However, another more unsophisticated implementation, e.g. calculating the percentage size of the concession step from the height of one's own last price proposal, would result in a disadvantage for the seller because the calculation would be based on a higher price, which is shown in the following example:

A seller A wants to sell a good for a price of 80 ***monetary units (MUS)****, but the potential buyer B only offers 60. If both are willing to make a concession of 20% then A would make a concession of 16 MUS and ask in the next negotiation step a price of only 64 MUS, i.e. he would reduce his asking price by 16 MUS, while B would be willing to dispense only 12 MUS more.*

Thus, the increase in welfare by a negotiated transaction would not be non-symmetrically distributed. Firstly, this would lead to a reduction of the market price for the sold goods and thus to a decrease of sales and probably a decrease in the overall welfare for the entire multi-agent system.

Secondly, this would result in a coordination mechanism by which the increase in welfare from a negotiated transaction would be distributed non-symmetrically representing a non-desired feature of a coordination mechanism in general.

Thirdly, based on both of the above reasons it would not be ***individually rational*** for a seller agent to take part in the negotiation mechanism.

Therefore, an agent A calculates his current *individual* step size for a concession at the beginning of a negotiation using the following formula:

$$current.stepSize_A = (asked\_price - offered\_price) * del\_change$$

The ***delta_jJump*** parameter defines the margin an agent wants to realize according to his consecutive buy (input goods) and sell (output good) transactions (the higher *delta_jump*, the higher the aimed margin between buying cost and demanded selling price). For this purpose, *delta_jump* modifies the first price proposal of agent A in a negotiation as follows:

```
// new starting price for negotiations is agreement price plus del_jump%
lastSellPrice = agreement;
sellMaxPrice = agreement * (1.0 + del_jump);
```

The fourth parameter, ***satisfaction,*** defines the probability that an agent aborts the negotiation and thus ensures that a negotiation does not continue on and on. The abort probability after the n^th negotiation round amounts to $(1 - p\_satisfaction)^n$.

To avoid individually nonsensical behaviour the agents have a learning function which validates an offer before starting a negotiation. Without a validation an agent would negotiate no matter which price was offered by the negotiation partner as illustrated by the following example:

Agent A wants to sell a good for 1000 MUS but agent B offers only 10. Without validation they would negotiate and perhaps they would make a deal for around 500, which results in a disadvantage for one of them, since, if the market price were 900, B made a fantastic deal which would be very irrational for A.

This circumstance is based on the "common value" principle. To detect such circumstances the agents store transaction prices, i.e. the end prices of successful negotiations, from their negotiation history in a data structure *memory* and calculate an internally ***"sensed" market price (smp)*** for each good of interest.

This is necessary because there is no central institution for declaring market prices. The information stored in *memory* is used to compute his *smp* with exponential smoothing. Thereby, the parameter ***weight_memory*** specifes how fast market changes have influence on the market price.

```
// update our memory of initial prices
memory = offer.price * w_memory + memory * (1-w_memory);
```

On this basis, at negotiation start each agent checks the first price proposal of its opponent against his sensed market price. All counter-proposals exceeding the doubled "sensed" market price are rejected directly to avoid extortion offers.

All counter-proposals lying between the sensed market price and its doubled value are estimated as uncertain and a possible negotiation abort is tested according to *p_satisfaction:*

```
if (offer.price >= memory) {
// ...then use memory to check for too high prices
If (randomNumberIsHigherThan(p_satisfaction)) {
reject = true;
}
if (memory != 0 && offer.price > 2 * memory) {
reject = true;
}
```

The last parameter ***reputation***, defines the probability to finish a deal correctly according to the reputation of a negotiation partner in the system.

These six parameters describe completely the behaviour of an agent during a negotiation. They are predefined, but their values are adapted by the following evolutionary algorithm during a simulation run that may last hundreds of thousands of rounds.

4. EVOLUTIONARY ADAPTATION

4.1 Negotiation and Learn Process

Each MACSIMA agent possesses a pool of genotypes and a population of ***plumages*** (genotypes with estimated fitness values). Their sizes are to be defined at the start of the simulation. After the start of a bilateral negotiation, the first step of an agent is to choose a genotype - determining his strategy for this negotiation - out of his pool of genotypes. Then, both agents negotiate until one of them aborts the negotiations or their price proposals are crossing and they make a mutually beneficial deal. After a successful negotiation both agents calculate a fitness value for the used genotype and store the resulting combination of genotype and estimated fitness as a so-called ***plumage*** into the population data structure.

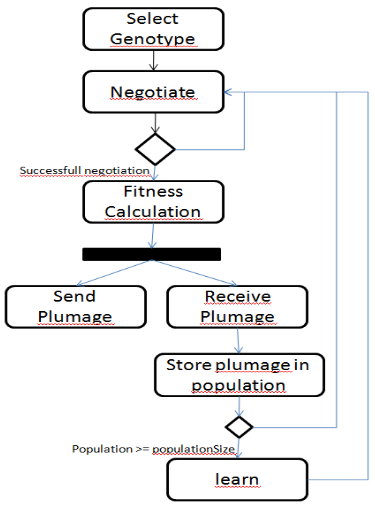

Figure 3: Negotiation and learn process

If their information exchange mode is set to external or mixed, they afterwards send the resulting plumage to other agents, receive plumages from other allied agents and store the self-calculated as well as the received plumages in their population. If the number of stored plumages is larger than the population size the agents start their learning process by using their individually parameterized evolutionary learning mechanism. This process is sketched in the following figure.

In the learning process all plumages within the population are assigned to a ***selection method***, which selects some plumages and assigns as many plumages as the pool size allows to a ***recombination process***. Optionally, the plumages are modified by probabilistic **mutation** after the recombination step and the newly generated pool is assigned to the agent. In the last step of the learning process, the old population of the agent is deleted and the agent can now start new negotiations.

Summarized, the evolutionary learning mechanism of an agent is divided into 4 different phases. First of all, a fitness value for each genotype used successfully in a negotiation is calculated (see 4.3 Fitness Calculation). Then, the best evaluated genotypes are selected (see 4.4 Selection). After the selection process two genotypes are recombined to new genotypes (see 4.5 Recombination). In the last phase the newly built genotypes can be mutated (see 4.6 Mutation). These 4 phases of the learning process and the different modes for information exchange allows a flexible set up of supply chain scenarios to be simulated. The possible settings for information exchange and the parameterization of the learning process are described in the following.

4.2 Information Exchange

An agent learns either only by himself (***internal learning mode***), i.e. does not use the experiences (in the form of plumages) of other agents. Or an agent has "colleagues" that exchange information about the nature of successful finished negotiations with him in such a way that he can use this information in his next evolution step (***external learning mode***). An agent may also use both modes in parallel (***mixed learning mode***).

If the agents learn not only internally, the system designer can build plumage-exchanging groups of collaborating agents at each tier of the supply network. So it is possible to build e.g. two groups at the same tier and use different evolutionary algorithms for each group. We call such simulations "tier tests". Another option is to set the information exchange to "Everybody" so that all the agents exchange plumages with each other and reserve no private information. In a real world supply environment of autonomous, self-interested agents the latter would rarely occur but rather the agents would concentrate on some "allies" and learn from their own made experiences and the hints of their allies, i.e. they would be assumed to learn in mixed mode. Of course, the issue with information provided by others is always if they are trustworthy or not, which can be modeled by applying the reputation parameter described above.

4.3 Fitness Calculation

After a successfully finished negotiation, i.e. a mutually closed deal or respective transaction, the fitness value for the genotype used in the successfully closed negotiation is calculated. MACSIMA offers the agents' designer the following ***fitness calculation methods,*** the evolutionary algorithm an agent may be provided with.

The most simple way to compute the fitness value of a used genotype is through the ***price_minus_average (PMA)*** fitness function using the margin between the current average price of a good and the current price: *fitness = avgPrice-currentPrice.*

Thereby, the duration of a negotiation has no influence on the calculated fitness value.

The more complex ***percental_average_proceeds (PAP)*** method takes the duration of a successful negotiation into account by dividing the PMA fitness value by the average price times the number of rounds in which this genotype was used.

$$\text{fitness} = \frac{avg\,\mathrm{Pr}ice - current\,\mathrm{Pr}ice}{avg\,\mathrm{Pr}ice * roundForCurrentGenotype}$$

The resulting PAP fitness value is influenced by the stored average price, like the following example shows:

Agent A has an average price of 10 MUS and makes a deal with a transaction price of 9 in only a single round. Then his fitness is 0.1. When an Agent B has an average price of 8 calculated from his past experience and closes a deal for 9 then the fitness value is -0.125. Accordingly, the same price can lead to different fitness values because it is influenced by past experience.

By the ***percental_absolute_proceeds (PAB)*** method the value of the PMA variant is divided by a fixed basicPrice times roundForCurrentGenotype.

$$fitness = \frac{basic\,\mathrm{Pr}ice - current\,\mathrm{Pr}ice}{basic\,\mathrm{Pr}ice * roundsforCurrentGenotype}$$

The last fitness calculation method implemented in MACSIMA is the ***percental_mean_proceeds (PMP)*** method using the mean value (mediumPrice) of the starting price proposals of both partners in the negotiation in the calculation.

$$fitness = \frac{medium\,\mathrm{Pr}ice - current\,\mathrm{Pr}ice}{medium\,\mathrm{Pr}ice * roundsForCurrentGenotype}$$

4.4 Selection

4.4.1 Binary competition

Two randomly selected individuals are compared and the one with the higher fitness value is selected and copied in the new population. This is done repeatedly until the new population has reached its defined maximum size. For avoiding that - because of the random selection - the same individual is contained several times in the new population, the winner of a comparison is deleted out of the old population. Random selection may have the effect that only individuals with bad fitness value are selected.

4.4.2 Roulette-Wheel-Selection

Each individual is assigned a section on a wheel based on his fitness value according to the formula:

$$\alpha = 360\frac{f(I_k)}{\sum_{n=1}^{N} f(I_n)}\text{ , with}$$

α : Angel assigned to the k-th individual

$f(I_K)$: Fitness value of the k-th individual

N : Number of individuals

Each individual has a chance to be transferred into the new population and so the generic diversity remains. On the other hand there is a chance that not all good individuals are transferred.

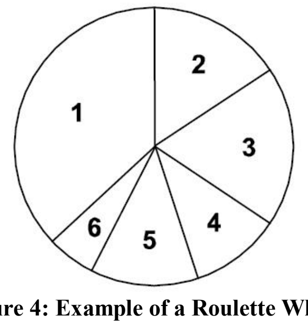

Figure 4: Example of a Roulette Wheel

4.4.3 Deterministic Selection

Based on the fitness value an expectation value is calculated:

$$\textit{Expectation } E(X) = f_i \frac{N}{\sum_{k=1}^{N} f_k}$$

The expectation numeralizes the number of expected successions of an individual. In a first step each individual is assigned the whole-number part. Following this, the individuals with the highest decimal places get successions until the new population is complete. This method prefers individuals with high fitness values.

4.4.4 Deterministic Average Selection

All methods above calculated the single fitness value for each individual genotype. But the same genotype could be contained in the population more than once since it might have been used in several negotiations with differing success and a therefore differing fitness values. The deterministic average selection calculates the average fitness value for each individual genotype before the selection starts. The individual with the best fitness value comes directly into the new population; the worst one is deleted. The remaining genotypes are selected in the same manner as in the deterministic average selection method.

4.4.5 Deterministic Average Selection with deletion of the worst individual

The new population is filled up in the same way as with the standard deterministic average selection until the number of individuals in the new population equals pool size minus the value of BestIndividualsToSurvive. For the last place a new genotype is generated as the mean value of each gene of the genotypes already included in the new population.

4.5 Recombination

Apart from the selection, the population has a hand in the recombination. The main idea behind this is building new and better individuals from two good old individuals. This crossover is a kind of macro mutation which creates jumps in the search space and therefore helps to find a global solution.

4.5.1 n- Point- Crossover

The two best individuals are taken out and with a probability, the crossover probability, these two are recombined or they are put back in the population unchanged. If the two individuals are recombined they are cut at ‘n’ randomly chosen positions and linked crossover. The new generated genotypes replace their parents in the population.

4.5.2 Random Crossover

Two individuals are selected and for each gene it is decided which is taken for the new one according to a probability identified by preferBetterparent.

4.6 Mutation

The main target of mutation is to keep the diversity in the population and find solutions that are not reachable with the other phases. Usual modifications according to the Gaussian distribution are suggested in [1]. In MACSIMA, all values are in the interval [0;1] and calculated according to the formula:

$$\textit{Gen} = \textit{Gen} + \textit{gaussWidth} * \textit{nextGaussian}(),$$

where nextGaussian returns a random number and gaussWidth is the breadth of the Gaussian distribution. For each genotype in the population it is decided if it is mutated - and if so it is changed at exactly one position.

If, instead of this, a so-called single mutation mode would be used each gene would be checked according to the mutation rate whether it is mutated or not. Thus a single mutation mode can lead to a completely changed and therefore new genotype.

4.7 Replacement scheme

After the creation of the new population it has to be decided what to do with the old one. By simply deleting the old population there is the risk that all new genotypes are worse than their parents because of the recombination and mutation process.

4.7.1 Elitism

To protect the best ‘n’ individuals from being modified the individuals with the highest fitness value are transferred into the new population unchanged. This can result in a strong dominance if the fitness values vary too much.

4.7.2 Weak Elitism

As before, the best ‘n’ individuals are transferred into the new population but before this they can be mutated, as described above. This avoids dominance and even more so it assures that good individuals are reused.

After this description of the evolutionary algorithm used by the agents, the question is now when the agents learn and which states an agent can reach.

5. SIMULATION RESULTS

The agents in MACSIMA log their internal status as well as their activities, negotiation steps and final outcomes, with regard to their content, in several separated and statistically evaluable data files in csv-format. This raw data comprises information about the *evolution* of the agents’ individual negotiation strategies, the course of negotiations together with their outcomes etc. This raw data can be easily transformed into diagrams that show the course of the evolution process of the agents’ strategy parameters together with the emerging price fluctuations for the traded goods in the time elapsed.

5.1 Basic simulation scenario

A scenario with a network topology as shown in figure 1 is defined, the evolutionary learning mechanism of the agents is turned off and all the agents are provided homogeneously with the same static strategy parameters (*acquisitiveness = 0.5*, *delta_change = 0.25*, *delta_jump = 0.15*, *satisfaction= 0.75*, *weight_memory = 0.2* and *reputation = 1*). If a simulation is started, a price distribution emerges that is uniformly distributed around the start prices of the different goods with a spread according to the adjusted DeltaJump-parameter (see figure 3).

The x coordinate of the diagram indicates the number of simulation rounds, the y coordinate shows the fluctuating good prices and each symbol represents a successful negotiation outcome, i.e. transaction. Since all agents started with the same strategy parameters, no tier of the supply chain, i.e. group of agents of a certain type, is able to force the agents on other tiers to significant and long-lasting concessions.

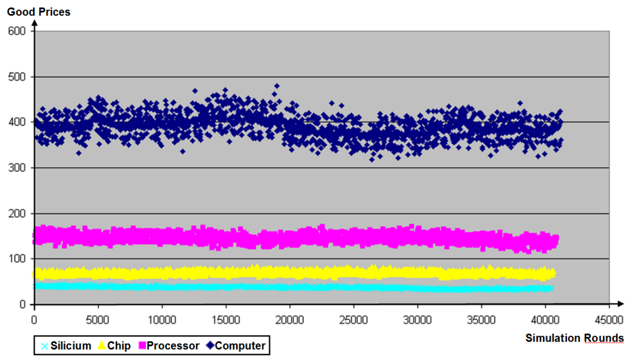

Figure 5: Price fluctuations in a setting with static strategy parameters and genetic adaptation turned off

5.2 Impact of the acquisitiveness parameter

If we essentially keep the same scenario (genetic adaption still turned off) but only set the acquisitiveness parameter of the agents producing processors to 0.51 instead of 0.5 they are able to realize a strategic advantage in their negotiations with the adjacent tiers, as pointed out by the figure 4.

One can see that the processor producers are now able to achieve lower prices for their input good as well as higher prices for their output good. Whereby they manage to raise the accepted price monotonously over a long period within the simulation run.

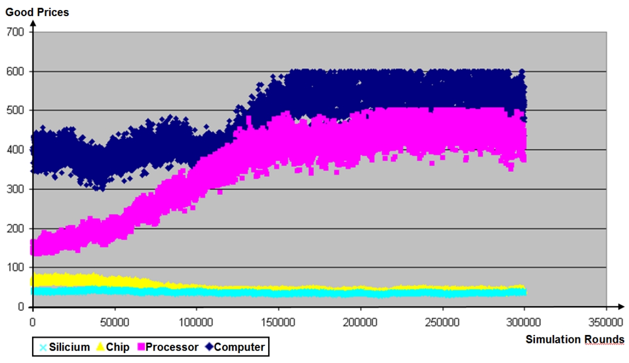

Figure 6: Price fluctuations in a setting with static strategy parameters and genetic adaptation turned off

At the end of the simulation, the prices of the goods stabilize since the consumers' maximum willingness to pay has been predefined at 600 monetary units.

5.3 Impact of evolution at one tier

Now we keep the settings of the foregoing scenario but exclusively provide the 10 processor producer agents with the ability to learn from previous negotiation outcomes. Their new learning capability empowers them to benefit much faster from their slight competitive advantage (acquisitiveness set to 0.51 instead of 0.5) compared to the foregoing simulation run.

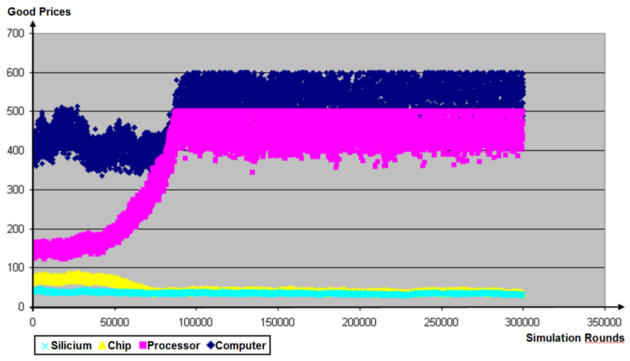

Figure 7: Learning at one tier results in a faster strategic adaptation and higher profits for agents of that tier

5.4 Emergence of niche strategies

But as we can see in the figure 6, this effect is not achieved because all of the 10 processor agents follow the same strategy, i.e. are learning that raising the value of their acquisitiveness parameter incrementally results in higher profit. Instead of this, it can be observed in the following diagram that a minority of the agents of the processor tier seem to benefit from following a niche strategy. Figure 6 shows the underlying adaptation and niche creation process for the acquisitiveness parameter of all of the 10 autonomously negotiating agents at the processor tier.

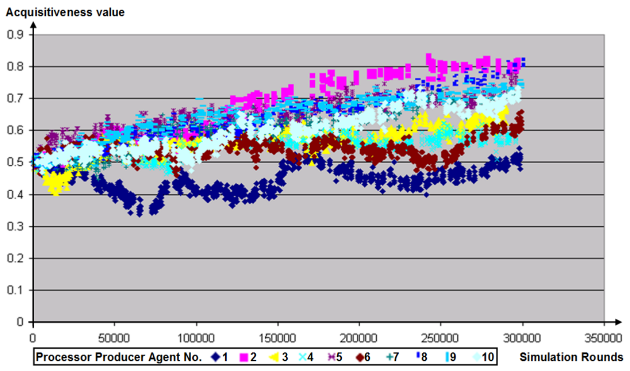

Figure 8: Heterogeneous parameter adaptation and the emergence of niche strategies

The indicated niche strategy consists of keeping the value of one's own acquisitiveness parameter slightly lower than the majority of the agents of the same type - or at least by stabilizing it at the lower spectrum of their parameter value range. This strategy can also lead to economic success since the majority of the processor producing agents reach high to very high acquisitiveness values during the simulation run and thus show a very tough negotiation behaviour by making almost no concessions. By their behaviour, they enforce an incrementally raising market price for the good processor against the downstream tier, represented by the computer producers.

Agents of the same type but of lower acquisitiveness than their colleagues can benefit from this fact since their negotiation partners calculate their price thresholds according to the current market price and their past experience. Thus, they expect no concession from a processor agent and expect to pay at least the current market price for the processor at the end of the negotiation. Hence, they accept quickly, if unexpected concessions are made and the offered price proposal lies only slightly below the expected market price. By making more concessions than the other agents of the same type, some processor agents reach more negotiation deals than their "tougher" colleagues and achieve more sales and profit even though their negotiated price may be lower on average. This kind of behaviour is very similar to the behaviour adopted by humans in comparable economical situations.

6. DELIMITATION

The main advantage of MACSIMA is the ability to set up nearly all possible supply chain layouts or topologies and to instantiate them with numerous intelligent agents using a ***learning mechanism*** (see section 4) that can be adjusted to great detail.

This is a progressive step as compared to the limited learning features of similar approaches for the simulation of agent-based negotiations in market-like environments (see e.g. [2] and [7]) that offer only very limited learning capabilities. Besides this, one further differentiating factor of MACSIMA consists in the fact that the agents can use different ***learning modes*** with respect to the extent of information exchange with other agents in the network. In this way, the system designer or experimenter is able to build cooperating groups out of several agents on each tier and to examine the effects of coalition formation between agents in the network.

Both allows for directly comparing different learning mechanisms in combination with different information exchange modes under the same external influences and environmental constraints in a supply chain domain. Summarized, to our knowledge MACSIMA represents so far the most powerful simulation framework for the experimental analysis of the co-evolution of negotiation strategies within a society of adaptive supply agents that use bilateral negotiation as their coordination mechanism.

7. CONCLUSION AND OUTLOOK

We have describe an evolutionary learning approach for the co-evolution of bilateral price negotiation strategies in an agent-based supply network environment. The effects of the adaptation processes of the intelligent agents have been simulated and evaluated by using the MACSIMA framework.

MACSIMA is suited for modeling and instantiating large-scale agent-based supply networks, built up of a multitude of autonomous agents of different generic types. The agents coordinate their supply processes by using a bilateral negotiation mechanism based on a social choice model that is also adopted by humans in economic real-life negotiations. Additionally, MACSIMA provides all agents with elaborated genetic learning mechanisms that are individually parameterized, i.e. they are able to adapt their negotiation strategy parameters in a completely individual way according to their precedent negotiation successes. The parameterization of the learning mechanism of each individual agent in a simulation scenario can be fine-tuned.

Furthermore, MACSIMA enables the agents to exchange information about finished negotiations with other agents. Thereby, the extent of information exchange with other, possibly "allied", agents within the same tier can be adjusted in such a way that a loose cooperation between agents as well as predefined coalitions of agents can be modeled. In this way, the evolution of an agent's negotiation strategy is not only guided by his own experience but can also take the experience of other agents into account. We have outlined some evaluation results with a particular focus on the emergence of niche strategies within a group of cooperating agents at one tier of a supply chain for computer manufacturing.

We have defined a multitude of different supply network scenarios and have conducted comprehensive and extensive simulation runs. On this basis, we have outlined some evaluation results with a focus on the emergence of niche strategies within a group of cooperating agents. Naturally, the question occurs if a combination of a learning mechanism and information exchange mode does exist from which - if applied by all the agents in the network - social welfare's maximizing effects may be expected. Answering this question will the core of our future research work.

8. ACKNOWLEDGMENTS

The authors would like to thank the Multiagent Systems Group of the German Research Center for Artificial Intelligence (DFKI), the Dacos Software GmbH and the Graduate School of Advanced Manufacturing (GSaME) for their support. Furthermore we thank Christian Zenner and Heiko Graeber for their contributions.

9. REFERENCES

[1] Back, T., Fogel, D. B., and Michalewicz, Z., 1997. The Handbook of Evolutionary Computation. Oxford University Press, 1997.

[2] Eymann, T., 2000. AVALANCHE - Ein agentenbasierter dezentraler Koordinationsmechanismus für elektronische Märkte, Inaugural-Dissertation, Albert-Ludwigs-Universität Freiburg im Breisgau, 2000.

[3] Fischer, K., Russ, C., and Vierke, G. 1998 Decision Theory and Coordination in Multi-agent Systems. Research Report RR-98-02, DFKI, 1998.

[4] Laseter, T. M. 1998. Balanced Sourcing: Cooperation and Competition in Supplier Relationships. Jossey-Bass, ISBN: 0787944432, October 1998.

[5] Porter, A. M. 2000. Supply management in 2010: Experts see big future for e-procurement. In: Purchasing online, http://www.manufacturing.net/magazine/purchasing/, March 23, 2000.

[6] Pruitt, D. G. 1981. Negotiation Behaviour, Academic Press, New York, 1981.

[7] Smith, R.E., Taylor, N., 1998. A Framework for Evolutionary Computation in Agent-Based Systems, In: Looney, C., Castaing, J.: Proceedings of the 1998 International Conference on Intelligent Systems, S. 221-224. 1998.