7th WSEAS Int. Conf. on MATHEMATICAL METHODS and COMPUTATIONAL TECHNIQUES IN ELECTRICAL ENGINEERING, Sofia, 27-29/10/05 (pp168-174)

Agent-Based Negotiation Platform for Contract Electricity Market

XIONG MINPENG [1], YUAN JIAHAI [2], ZHAO CHANGHONG [3]
Business School
North China Electric Power University
79# postbox Zhuxingzhuang, Dewai, Beijing, 102206
CHINA
, ,

Abstract: - *A*n agent-based platform for contract negotiation in electricity market is presented. An intelligent agent implements the negotiation process by selecting a strategy based on learning algorithm in an interactive manner with the user. Two kinds of learning algorithm--fuzzy logic controller modification of basic Genetic Algorithm for negotiation strategy optimization, and reinforced learning algorithm for parameter modification of negotiation tactics--are provided for the agent. Protocol Operation Semantics that meet the requirement of sequential message exchange flexibly is used as agent communication mechanism. The paper presents the architecture of the agents and details with its software implementations.
Key-Words: - Agent-based system; Electricity market; Contract negotiation; Power system

1. Introduction

The electricity markets have evolved over the past decade to substitute ever-existing regulated monopolies in the electricity supply industry. The basic markets are: the power pool, power exchange auction, and bilateral contracts. Restructuring the electricity industry into an open market has created demands for new software tools to meet future challenges and expanding requirements of competitive power systems. Especially for market participators they need some intelligent system for decision support and implementation of competitive strategy.

There are many efforts discussing the bidding strategies of Generation companies in power pool and power exchange auction market. These include game theoretical method [1-4] and genetic algorithm [5]. However little effort has devoted to the negotiation strategy in contract market [6]. In this paper we propose an agent-based negotiation platform for participators in contract market. The platform combines the strategy optimization of the agents with negotiation implementation for contract market.

2. Agent- based system

An agent is an encapsulated computer system that is situated in some environment, and is capable of flexible, autonomous action in that environment in order to meet its design objectives [7,8]. Multi-agent system (MAS) is a system that is consisted of two or more interactive agents in a common environment. Currently the most widely used communication mechanism in MAS is KQML [9]. However KQML is too rigid and makes too strong assumptions when defining the semantics. More importantly, it only allows the specification of individual messages, but hardly permit to deal with sequences of message exchanges, which are unavoidable in complex negotiation situation, such as in contract market. So when designing a trading platform for negotiation in electricity market appropriate communication mechanism is needed.

Multi-agent system captures the decentralized nature of electricity supply industry after restructuring and is a natural tool for market research. Recently there are considerable discussions on application of agent-based system in electric market research [10-12]. However in these works the agent system is mainly for purpose of market simulation and not designed to deal with actual applications. To see an overview of MAS in electricity market research refer to [13]. In order to utilize MAS as real market trading system, the trading strategy, the learning algorithm of agent as well as the communication mechanism must be properly defined and related supporting functions for trading such as market forecasting and analysis must be incorporated within the agent platform. Our paper tries to provide such a realistic system.

3. Electricity negotiation framework

Negotiation is defined as a process by which a joint decision is made by two or more parties. In electricity market, the negotiating commodity is the electric energy contract. The negotiating parties are power generating and power-consuming companies (GenCo and ConCo) and/or energy brokers. Generally the contract has the following parameters:

1. z: Contract volume (Mega Watt hour MWh)
2. ts and te: Contract starting time and ending time
3. Pmax(t): Maximum amount of electricity (MWh) that can be drawn during time t
4. tl: Contract lead time defined as the minimum time between scheduling decisions and delivery of energy.
5. f: Contract price ($/MWh)

The negotiating parties communicate through the Internet by utilizing agents to automate the negotiation process. Agents are designed to be adaptive, responsive to market changes, and apt to learning-by-doing.

3.1. Formulation of negotiation

Let ng and nc be the number of GenCo and ConCo negotiating agents in the electricity market. The jth ConCo has a demand curve Dj(r,t) as a function of price r for each t, and the system total demand is then D(r,t). The rth GenCo owns ngr generating units. The power generated by the ith unit owned by company r, during hour t, is gri(t). The costs associated with the unit are: start-up, shut-down, operation, and maintenance. The Unit constraints are: generation bounds, minimum up and down times, and ramp up and down limits. The system constraints are: generation-load balance, spinning reserve requirements, emission bounds, and transmission line limits. These constraints and associated contract parameter limits are formally described by a set of inequalities as $h(X,t) \le 0$, where X represents a vector of contract parameters. Each ConCo is assumed to determine its contract volume according to the following proposition [6].

Proposition: let $\hat{\rho}$ denote the expected market clearing price, then the jth Conco contract quantity is determined by

$$z_j = D_j(\rho,t) - \frac{\hat{\rho} - f}{\partial\hat{\rho} / \partial z_j} \qquad (1)$$

Eq.(1) implies that the contract volume largely depends on f and $\hat{\rho}$. If Gencos are non-cooperative and $f > \hat{\rho}$, then there will always be a GenCo, at least, that would be willing to undercut a rival in the contract market. If a GenCo chooses to contract at f, then the jth ConCo determines its price maximizing contract volume according to (1).

Each GenCo, on the other hand, negotiates its selling price and associated contract parameters so as to maximize its profit, defined formally as

$$\max_{X} \quad \pi(\rho,t,X,P) \quad s.t. \quad h(X,t) \le o \qquad (2)$$

Where P represents a relational knowledge representation of contract parameters and negotiation expert domain.

3.2. Negotiation process

In electric market, the trading participator first makes judgments on market situations based on market forecasting and analysis, then selects trading counterparts and initiates bilateral or multilateral negotiation based on its risk attitude. In our system market participator take on the duty of market situation judgment and trading strategy selection while forecasting analysis and negotiation process is implemented in the agent.

There are three kinds of negotiation approaches that commonly used in agents system: game theoretic method, heuristics and argumentation. Game theoretic method assumes that agents is perfectly computational rationality and is often computationally intractable. Heuristics on the other hand searches in the negotiation space in a non-exhaustive fashion. The final result of heuristics may be sub-optimal. Argumentation allows the agent to propose new information and may alter the entire negotiation space and is perhaps the most complex one in implementation. Because of its complexity and constraints of communication mechanism, application of argumentation in agent system is seldom. In our paper, in order to avoid unnecessary complexity we assume that the argumentation between the negotiation parties involves only contract parameters.

4. Tactics and strategy

Tactics are the set of functions that determine how to compute the value of the contract parameter with respect to a given criterion. Varieties of tactics have been proposed to account for opponent behavior.

4.1. Time-dependent tactics

In this tactics model, the value of the contract parameter j during time t, is given by [14].

$$X_{a\to b}^{t}[j]=\begin{cases}\min_j^a+\alpha_j^a(t)\left(\max_j^a-\min_j^a\right) & if \quad V_j^a\downarrow\\ \min_j^a+\left[1-\alpha_j^a(t)\right]\left(\max_j^a-\min_j^a\right) & if \quad V_j^a\uparrow\end{cases} \tag{3}$$

$\max_j^a, \min_j^a$: Maximum and minimum value of parameter j accepted by agent a.

$V_j^a\uparrow, V_j^a\downarrow$: Increasing and decreasing, scoring function representing a score GenCo agent a assigns to x of j.

$a_j^a(t)$: Offer proposed by GenCo a to ConCo b for a contract parameter and is expressed as

$$a_j^a(t)=\left(\frac{\min\left(t,t_{\max}^{\alpha}\right)}{t_{\max}^{\alpha}}\right)^{\frac{1}{\beta_j}} \tag{4}$$

Where $t_{\max}^{a}$ is the maximum negotiation time for agent a and β_j is the parameter of agent type.

4.2. Behavior-dependent tactics

In these tactics agents base their actions on the behavior of their negotiation opponent. The value of the contract parameter j for a Relative Tit-for-Tat action is given by [15]:

$$X_{a\to b}^{t_{n+1}}[j]=\min\left(\mu,\max_j^a\right) \tag{5}$$

Where $\mu=\max\left(\xi,\min_j^a\right)$ and

$$\xi=\frac{X_{a\to b}^{t_{n-2\delta_j}}[j]}{X_{a\to b}^{t_{n-2\delta_j+2}}[j]}X_{a\to b}^{t_{n-1}}[j]$$

The functions min and max in (5) take the minimum and maximum values of the given arguments, respectively.

4.3. Strategy

A ConCo agent receives an offer from a GenCo agent opponent. If the offer is unsatisfactory, the ConCo agent generates a counter offer. Different combinations of tactics can be used to generate a counter offer. A weighted counter offer $X_{a\to b}^{t+1}[j]$ would then be a linear combination of the tactics given in a matrix $\Gamma_{a\to b}^{t}$ [16], that defines a state of an agent MS containing information about the agent knowledge, resource, attitude, goals, obligations and intentions. The agent counter strategy is then

$$X_{a\to b}^{t_{n+1}}[j]=\left(\Gamma_{a\to b}^{t_n+1}T^a\left[MS_a^{t_n+1}\right]\right)[i,j] \tag{6}$$

in which $\left(T^a\left[MS_a^{t_n+1}\right]\right)[i,j]=\left(\tau_i\left(\left[MS_a^{t_n+1}\right]\right)\right)[j]$ and $\Gamma_{a\to b}^{t_n+1}$ is a newly updated matrix, which is a function of agent state and previous matrix $\Gamma_{a\to b}^{t_n+1}=f\left(\Gamma_{a\to b}^{t_n},MS_a^{t_n}\right)$.

5. Learning and optimization

Two kind of learning algorithms are designed for the agents. One is tactics parameter modification based on reinforced learning algorithm, the other being fuzzy logic modification of basic GA for strategy optimization.

5.1. Fuzzy logic controller genetic algorithm for strategy optimization

In the paper we utilize the fuzzy controller genetic algorithm proposed in [17] for strategy optimization. A negotiation strategy X is encoded as a chromosome with a number of genes. The genes represent the strategy parameters related to the contract and tactics of the strategy. Each chromosome has a fitness value defined by the profit function p for GenCos or the benefit function for ConCos. The main idea of fuzzy logic controller is to adjust the rate of crossover and mutation based on fuzzy logic in the optimization process. If the difference of average fitness value among several sequent generations is small, then crossover and mutation rate is enlarged until the difference becomes large. If the average fitness value declines, then crossover and mutation rate should be diminished. If the difference of average fitness value among sequent generations is nearly to zero then crossover and mutation rate should be increased greatly. Introducing fuzzy logic controller can speed up the convergence process and meet the time requirement of bilateral negotiation.

By applying genetic operators: selection, crossover and mutation, a new population of chromosomes is generated. The new population is expected to consist of individuals with higher fitness values. Then fuzzy controller logic for adjustment of crossover and mutation rate is conducted in optimization process. Because genetic algorithm is a rather standard algorithm the detail of its operation is omitted here. To see the detailed operation process of fuzzy logic controller GA, see [17] for reference.

5.2. Reinforced learning for tactics parameter modification

Often the agent needs to adjust the parameter of negotiation tactics or strategy as the external and internal situation changes. Agents can learn about the appropriateness of tactics and strategy parameter based on reinforced learning.

For simplicity, the state of market that the GenCo or ConCo agent faces is an enumerable set of elementary outcomes, $S = \{S1, S2, ...Si, ...\}$

After making market situation analysis each agent get an imprecise impression of the state of the market

$\Theta^t = \{\upsilon_1^t, \upsilon_2^t, ...\upsilon_j^t\}$, Where $\upsilon_j^t \subseteq S, \quad \Theta^t \subseteq 2^s$.

For each parameter of tactics or strategy suppose it is defined in a discrete space and denumerable. Agent adjust the value of parameter according to Eq. below [18,19]:

$$F_k^{t+1} = F_k^t \frac{f(\pi^t(a_k))}{\sum_k f^k(\pi^t(a_k))} \qquad (7)$$

$$P^t(a_k) = \frac{F^t{}_k}{\sum_i F_i^t}$$

Where

This selection mechanism induces a stochastic process on the strengths assigned to competing value of each parameter. If the action of selecting kth value under certain state in the past gives out better payoff then its strength in next period is enhanced, else the strength is decreased.

6. Infrastructure of the platform

6.1. Agent infrastructure

We design the agent platform based on intelligent engineering proposed in [20,21]. In Fig. 1 the main idea of agent design is illustrated. Knowledge base is user-defined rules for agent inference. The knowledge is open to avoid knowledge extraction and expression puzzles and to incorporate the experience and judgment of user into the system. Interference machine is an expert system to fulfill the reasoning process. Method base manages a set of methods such as forecasting methods, optimization methods for problem solving, while Model base is subordinated to method base and manage sets of concrete models, such as short-term forecasting model, long-term forecasting model, fuzzy logic Genetic Algorithm etc. Finally the database manages and stores the data and information for the agent.

Fig.1: Schematic Map of Agent Design Idea

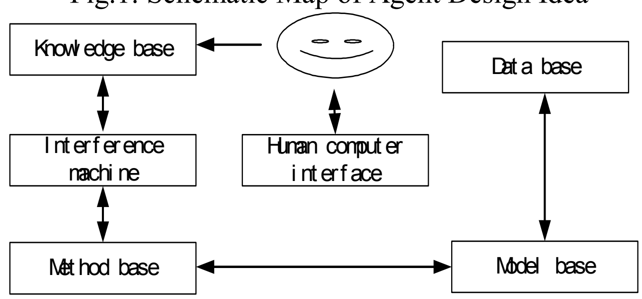


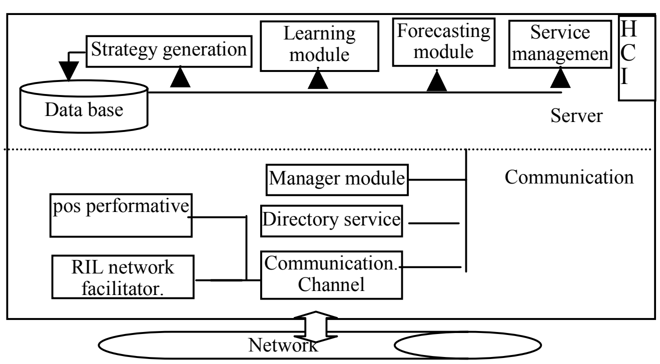
Fig. 2 illustrates the infrastructure of the agent. It represents the activity functions [6]: market situation forecasting and analysis, optimization, negotiation, management, directory service, and communication. Forecasting and market analysis module makes market situation analysis for the decision period and then the user bases on the analysis to make his or her judgment on strategy selection. This is a very important function because in market situation all the decisions made by market participators are based on right judgment of market situation. Because there are so many forecasting models such as time series model, ANN model and others we are not going to detail with them. The negotiation process is implemented within the negotiation server. The server handles strategy optimization, evaluation, and registration. Management keeps track of connected agents. Directory keeps record of connected agents. Finally, the agent communication component implements agent communication using Protocol Operation Semantics (POS). The peer agents are identified by the Internet Protocol (IP) addresses of their servers. A server carries out the negotiation process by checking the offer parameters against its user-registered information. Based on this information, the server then consults with the user for decision-making, generates a counter-offer to be returned to the opponent or sends a POS-wrapped reply. The opponent peer agent would then either reject the offer or suggest possible changes.

Human-Computer Interface includes input and output interface. The input interface is to receive the instruction and demand of user to the agent, such as the strategy parameter set by the user, etc. Also the user is to interfere with the negotiation process by the input interface. The output interface is to visualize the negotiation process, the payoff of the agent during the negotiation, and status of other agents in the system.

6.2. Negotiation server

Negotiation server [6] stands for GenCo and ConCo to fulfill the negotiation task and accomplish three functions: (1) Registration service to allow a ConCo or GenCo specifying the requirements and constraints of the electricity contract. The user can also register the agent with a set of negotiation rules, which specify the negotiation strategies to be followed when constraints are violated during the negotiation phase. (2) Evaluation service to evaluate offers and possibly generate counter-offers, and (3) an event trigger service to detect and manage events and to trigger proper rules when events occurred.

Fig.2: Functional Architecture of Agent



6.3. Communication

In the system Protocol Operational Semantics [22] is selected as the communication implementation mechanism. Basically, making use of the Protocol Operational Semantics allows to specify for each agent a set of rules (called a protocol) that will monitor the use of basic behaviors depending on predicates or evaluations about the world (internal or external) and message exchanged with other agents. The abstract architecture of an agent using a protocol is a coupling of two subparts: the set of rules defining the protocol and the set of basic behaviors/actions and predicates/evaluation the agents can perform. POS's strength and specificity is in that it uses algebraic data types and pattern matching, allowing describing powerful protocols in a very compact way. Another advantage of POS is that it can deal with sequential message exchange and meet the requirements for contract market negotiation.

The three kinds of rules available in POS are:

$$\textbf{type 1}\quad \overbrace{\langle parSt, \phi(world)\rangle}^{\text{trigger}} \xrightarrow{[Send]} \overbrace{\langle parSt', [\mathcal{A}(world)]\rangle}^{\text{consequence}}$$

$$\textbf{type 2}\quad \langle parSt, \phi(world)\rangle \xrightarrow[msg]{} \langle parSt', [\mathcal{A}(world)]\rangle$$

$$\textbf{type 3}\quad \langle parst, \phi(world)\rangle \xrightarrow{\epsilon} \langle parst', [\mathcal{A}(world)]\rangle$$

Where:

parst and parst' are parameterized states in form of objects with an algebraic data type on which pattern-matching can be performed. The rest of the rule refers to such patterns through their variable name. Thus, pattern matching can be seen as a unification procedure, which fails if the pattern does not match, and which possibly binds a number of variables if it succeeds.

msg is a message pattern, which is also given in the form of an object with an algebraic data type on which pattern-matching can be performed. In particular, if one makes a POS specification of KQML, there would be messages like,

ask-one Content(Price),
For Contract Volume z and Time (ts, te)
Receiver (generator-j server)
Language ('Prolog')
Ontology ('Business')

$\phi(world)$ is a logical combination of predicates over the world. Such a predicate is defined as a procedure that returns 'nil' if false or a non-nil value otherwise, that can additionally be linked to a variable.

$\mathrm{A}(world)$ is a list of side-effects performed by the agent onto the world, in the paper this corresponds to the activation of a basic behavior.

$send$ is a list of message sendings of one of the following possible types

$sendToId \quad id \quad msg$: send the message msg to agent id

$sendToGroup \quad list \quad msg$: send the message msg to a group of agents listed in list

$sendToALL \quad msg$: send the message msg to all agents.

Type 1 rules correspond to the sending of messages, hence they are called sending rules. Type 2 rules correspond to the reading of a message from the agent's mailbox, so they are denoted as receiving rule. With type 3 rules no message is exchanged, they are denoted as ε rules.

A rule is fired when its parameterized state matches the current agent state and its predicate is true. Then, the corresponding list of message is sent and the side effects are performed upon the world. Furthermore, the agent state changes to the parameterized state indicated by the rule. For constraint of space for the detail of POS see Ref. [22].

6.4. POS architecture

The POS architecture consists of two specialized modules: a router and a library of interface routines named Router Interface Library. The Router gives an application a single interface to the network; provide both client and server capabilities, managing multiple simultaneous connections, and handing some POS interactions autonomously. The RIL is a programming interface between the application and the router, embedded in the application and has access to the applications tools for analyzing the content.

6.5. Negotiation process

After receiving an offer, the server of agent can do the following operations: accept; modify and reply; reject or terminate the negotiation. The process modeled by Petri net is shown in Fig. 3 [23].

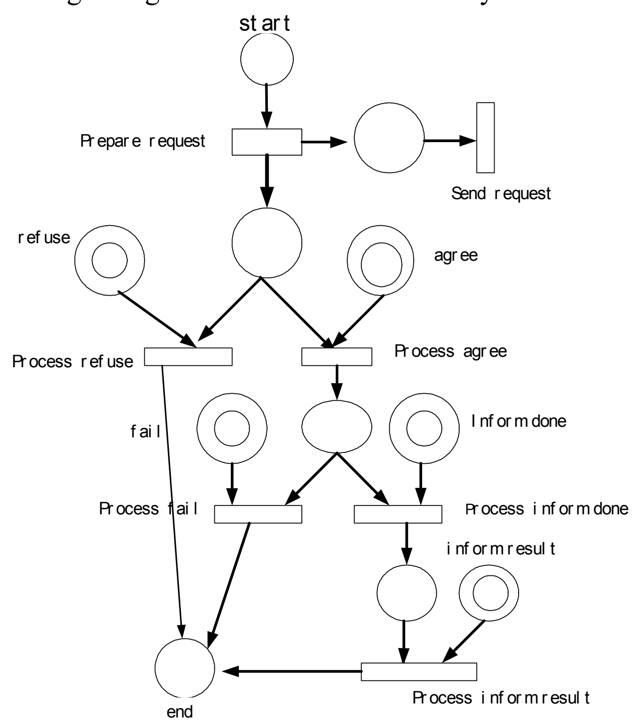
6.6. Software implementation

The agent is implemented on AgentBuilder [24]. It utilizes high agent-oriented programming language and is appropriate for our purpose. By defining beliefs, actions and commitments of agent an agent is designed and activated. The designing of agent is in a modular fashion. The intelligent optimization and communication components of the agent are designed separately and then encapsulated together. Because AgentBuilder can support advanced language such as JAVA and C++, the modules of market forecast and analysis and the fuzzy controller logic Genetic Algorithm can be programmed separately and then integrated into the agent platform.

7. Conclusion

The opening up of power market requires new trading system. In this paper we propose such an intelligent agent-based platform for contract negotiation. In real market the negotiation process is very complex and it is impossible to get an impression of market situation and the opponent's behaviors precisely, so in further research we are going to incorporate fuzzy logic in the platform to enhance its robustness.

Fig.3 Negotiation Process Modeled by Petri Net



8. References

[1] Fang De-bin, Wang Xian-jia et al. A Double Auction Bayesian Model With Supplier And Demander In Open Bilateral Electricity Market. Power System Technology 2003, 27(12), pp. 1-5.

[2] Shang Jincheng, HuangYonghao et al. A model and algorithm of game theory based bidding strategy for an independent power provider. Automation of electric power systems. 2002, 9, PP. 12-15

[3] Zhang Yubo, Luo Xianjue et al. Game solution to competitive bidding of generation companies under incomplete information. Proceedings of EPSA, 2003,15(1), pp. 27-32.

[4] Yeung C, Poon A, Wu F. Game theoretical multi-agent modeling of coalition formation for

multilateral trades. IEEE Trans Power System. 1999, 14(3), pp. 929–934.

[5] Richter C, Sheble G. Genetic algorithm evolution of utility bidding strategies for the competitive market place. IEEE Trans Power System, 1998, 13(1), pp. 256–261

[6] Al-Agtash S, Al-Fayoumi N., A trade server for electricity e-commerce. Computer Industry, 2002, 47(1), pp. 89–97.

[7] M. Wooldridge. Agent-based software engineering. IEE Proceeding on Software Engineering, 1997,144, pp. 26-37.

[8] M. Wooldridge, N. R. Jennings. Intelligent agents: theory and practice. The Knowledge Engineering Review, 1995, 10 (2), pp. 115-152.

[9] Yannis Labrou and Tim Finin. A Proposal for a new KQML Specification. TRCS-97-03, February1997, Computer Science and Electrical Engineering Department. University of Maryland Baltimore County, Baltimore, MD21250 (Report)

[10] Hong-jin Liu, Bin Yuan et al. Framework design of a general-purpose power market simulator based on multi-agent technology. Proceedings. Of IEEE Power Engineering Society Summer Meeting, 2001,3, pp. 1478-1482

[11] Vladimir S. Koritarvo. Real market representation with agents: modeling the electricity market as a complex adaptive system with an agent-based approach. IEEE Power& Energy Magazine, 2004 Jun/July, pp. 39-46.

[12] Isabel Praça, Carlos Ramos, et al. A multi-agent system that simulates competitive electricity markets. IEEE Intelligent System, 2003 Nov./Dec, pp. 54-60.

[13] Yuan Jiahai, Ding Wei, HU Zhao-guang. A critical study of agent based computational economics and its application in research of electricity market theory. Power System Technology, 2005,29(7), pp.37-41.

[14] Matos N, Sierra C, Jennings N. Determining successful negotiation strategies: an evolutionary approach. Proc. 3rd Int.Conf.on Multi-Agent Systems (ICMAS-98), Paris, France, pp. 182-189

[15] Guttman, Joel M. Rational actors, tit-for-tat types, and the evolution of cooperation. Journal of Economic Behavior & Organization,1996, 29(1), pp. 27-56.

[16] Shaheen S. Fatima , Michael Wooldridge, Nicholas R. Jennings. An agent-based framework for multi-issue negotiation. Artificial Intelligence, 2004 152 (1) pp. 1–45

[17] Wang Pingyang,Wang Guang-sheng, Hu Zhaoguang. Speed up the search of genetic algorithm. Power System Technology, 1999,23(5), pp. 3-7.

[18] A. E. Roth and I. Erev. Learning in extensive form games: experimental data and simple dynamic models in the intermediate term. Games and Economic Behavior. 1995, 8(2), pp. 164-212,.

[19] Giovanni Dosi, Luigi Marengo, Giorgio Fagiolo. Learning in evolutionary environments. 1996. Available at: ideas.repec.org/p/ssa/lemwps/2003-20.html (Report)

[20] Hu Zhao-guang. Studying on electronic laboratory of power economy by intelligent engineering. Electric Power, 2005, 38(1), pp. 7-11.

[21] Hu Zhao-guang. Intelligent space. IEEE International Conference on Fuzzy Systems, 1999, 3, pp. 1621-1625.

[22] J.-L. Koning , P.-Y. Oudeyer. Formalization, implementation and validation of conversation policies using a protocol operational semantics. Cognitive Systems Research, 2003, 4, pp. 223–2

[23] Gu feng,Yao shu-zheng. Research on Petri Nets based Modeling of Multi Agent Systems. Journal of System Simulation, 2003, 15, pp.105-108.

[24] AgentBuilder User's Guide. Version 1.3 April. 30, 2000.