

A Multiagent System Infrastructure and Negotiation Framework for Electric Power Systems

V. Vishwanathan, J. McCalley, and V. Honavar¹

{vijayav, jdm, honavar}@iastate.edu

Department of Electrical and Computer Engineering Department of Computer Science¹
Iowa State University, Ames, IA 50011, USA.

Abstract— Given the autonomous nature of various decision-making entities in deregulated power systems, this paper embraces a paradigm based on software agents and multiagent systems for distributed rational decision-making in electric power systems. Our current effort in this line of research is aimed at producing a platform-independent, robust application program interface for instantiating complex multiagent systems and a reference multiagent system application for the power systems environment. This paper describes the progress in our efforts on both these fronts.

Index Terms—agents, decisions, deregulation, multiagent, power systems.

I. INTRODUCTION

Complexities in information management and distributed decisions have the potential to evolve as a limiting factor for efficiently utilizing existing resources and optimally balance the interplay between economic and security concerns in the electric power system. Advances in artificial intelligence have made it possible to abstract a portion of a human decision-maker's authority as self-interested software agents, enable communication between these agents in a reasonably high-level language, and elicit coordinated behavior via suitable coordination models. To this end, this paper proposes the application of a distributed artificial intelligence paradigm known as multiagent systems to elicit coordinated and negotiated decisions from power system decision-makers. This technology is finding applications in coping with heterogeneous information systems [1]; intrusion detection in communication networks [2]; data-driven distributed knowledge discovery [3]; distributed problem solving, building distributed decision-support systems; enabling electronic markets; air traffic management system at Sidney airport, Australia [4], and many other application contexts. Applications of specific interest to us include those involving balancing system level security with an individual agent's economic self-interest. Instead of narrowly focusing on a particular application, we design and develop a Java-based software infrastructure, *MASPOWER*, to easily instantiate agents and distributed multiagent systems for complex environments and a generalized value-based negotiation framework for multi-objective decisions. In sections II and III, we declaratively specify the attributes of the agents and negotiation models, respectively, and a description of software implementation of these specifications is given in Section IV. Section V provides preliminary results from a negotiated decision-making scenario between power system decision-makers, and Section VI concludes.

II. THE AGENT MODEL

Foundational concepts and definitions of software agents and multiagent systems (MAS) together with a framework of application to power systems were described in [5]. An agent can be viewed as a software entity that maps percepts from the environment to actions on the environment in order to achieve its objectives. Percepts from the agent's environment can be thought of being composed of percepts from the *physical environment* consisting of power system components (usually abstracted as databases) and percepts from *multiagent environment* consisting of other agents. The complete sequence of percepts from the physical environment until the time instant t is denoted by ENV_t , while those from the multiagent environment is denoted by MAS_t . The complete history of percepts until the present time is therefore the union of percepts from the physical environment and the multiagent system. Denoting $t_{present}$ as the present time, the history of percepts is:

$$P^* = ENV_{t_{present}} \cup MAS_{t_{present}} \quad (1)$$

The agent usually has some knowledge about the world encoded as rules; this encoded knowledge is denoted by κ . This knowledge includes *common knowledge*, i.e., knowledge that is known by all the agents and knowledge unique to the agent. In our environment, common knowledge can include power system algorithms and other publicly available operating rules and procedures. Unless the agent is purely reactive, the agent first updates its internal states based on the new percepts and its knowledge about the world; the *state-generating function* that generates new states S is denoted by:

$$SG : (P^*, \kappa) \rightarrow S \quad (2)$$

Therefore the agent is unable to distinguish between some states of the environment; in other words, the agent has equal preferences for some environment states. This abstraction is often significant in complex environments like electric power systems that pass through a continuum of states, but allows the agent to deal with a finite number of states. One examples of a state-generating function is a single numeric risk index that lumps power system states together with various operational uncertainties into a quantitative value [6]. It is evident that designing such functions depends on the laws governing the physical system and requirements of the agent. Agents in our system make their decisions based on such internal state representation; we take a closer look at this process in what follows.

Usually, a human decision-maker transfers a portion of his authority to the software agent. Therefore the agent has a fixed set of actions that it is authorized to carry out, and this set is denoted by $\{a\}$. What actions can the agent engage in? This set could include control actions, acts of negotiating with other agents, communicating with human authorities, and so on. Given that agents are persistent software entities, the current internal state might require an action to be scheduled sometime in the future: the set of actions that the agent decides to carry out at a specific time instant t in future is denoted by $\{a_t\} \subseteq \{a\}$ (the actions that are scheduled at any time in the future should be a subset of the authorized actions for the agent). The agent uses its actions to move the environment from one state to another, thereby changing its internal states.

The agent usually has preferences over internal states. Going back to the earlier example, an agent may prefer a lower risk index over a higher risk index or vice versa. A sequence of actions on the environment can transform the internal state of the agent from one that is less preferred to a more preferred state:

$$S \times 2^{a_t} \rightarrow S \quad (3)$$

The method of transition is important: changes in the internal state of the agent necessarily come via the environment; otherwise, the agent can generate preferable internal states by merely manipulating its data structures. Every agent $a_k \in A$ has a *decision-making procedure*:

$$D_k : (S, S_G) \rightarrow 2^{a_t} \quad (4)$$

This procedure reasons about the internal state and the goals (S_G denotes any of the goal states of the agent) and formulates a plan of action to transform the internal state. The plan of action can consist of a sequence of actions indexed by time at which the actions have to be performed. Here again, identifying and implementing the decision-making procedure is specific to design of a single agent. Here, since we are taking a MAS perspective, we assume that every agent is equipped with a decision-making procedure, no matter how rudimentary. The optimal power flow algorithm is an example: it takes the current state and specifies a sequence of actions to change the current system state to a more preferred state. The current version of *MASPOWER* has a generalized decision-making procedure based on value functions as described in Section III.

Given the agent has recognized a need for performing some actions, how does it go about executing them? We use a notion of *tasks* within the agent to implement the mapping between internal states to actions. A *task* is defined as a finite state automaton that either maps a subset of internal states to actions or is composed of other tasks. We denote the various tasks currently under execution by an agent a as:

$$Tasks^a = \{task_1, \dots, task_i, \dots, task_n\} \quad (5)$$

Assume that the subset of internal states of the agent that is routed to any $task_i$ is S_i . If the task is not composed of other

tasks, then it maps the internal state of that task to a subset of allowable actions to be carried out a specific time:

$$task_i : S_i \rightarrow \{a'_i\} \quad (6)$$

It is possible that the decision-making procedures of different tasks recommend the same actions to be carried out at the same time instant. To avoid repetition, the set of actions that is carried out finally by the agent's effectors at a time t is a set-theoretic union of the actions recommended by all the tasks for that time instant:

$$\{a_t\} = \bigcup_i \{a'_i\} \quad (7)$$

The above declarative specification is independent of the internal reasoning model employed by the agent and the application under consideration.

III. THE NEGOTIATION MODEL

In this section, we describe a multiagent framework where agents can coordinate and negotiate with other agents for mutual benefit.

A. The Case for Negotiated Decisions

Decision-making authority is fragmented in deregulated power systems. Distributed coordination using decision-making based on rational multiagent system negotiation is a feasible alternative to pseudo-decentralized decision-making currently in place in the electric power industry. The intuitive arguments in favor of this are based on the observation that the two main components in distributed coordination between humans are communication and individual decision-making. Communication serves to disseminate relevant information to various entities; the content of such communication is usually formulated in a high-level language (like English) and mediums of communication are telephones, hotlines, e-mail, etc. A significant portion of decision-making by human entities is already software driven, usually relying on complex optimization programs, although it must be acknowledged that some part of decision-making (especially in the operational time-frame) relies on human intuition. Encoding a portion of human decision-makers' preferences and decision-making procedures, and enabling inter-agent communication in a reasonably high-level language, provides the essential to automate some of the tasks performed by human operators. The obvious advantages for the human entity include reduction in workload and stress. The act of negotiation is therefore included in the set of allowable actions in which an agent can engage. Negotiation is very complex because it has to accommodate multiple self-interested agents in its framework together with constraints on the physical system. In the next section, we describe a framework for negotiation based on multi-issued value functions.

B. The Negotiation Framework

Negotiations between agents revolve around one or more *issues* of common interest. For example, negotiation between a generator agent and a load agent may involve the issues of

quantity, unit price and quality. The current version of *MASPOWER* uses a framework based on value functions for multi-objective decisions. A paradigm based on value functions does not accommodate uncertainty in the consequence of an action. Value functions can be extended to utility functions to accommodate this uncertainty.

Foundations of multi-objective decisions using value functions have been formulated in the seminal work of Keeney and Raiffa [7] in the context of human decision-making. The framework does not easily apply to negotiations between self-interested software entities, so it has been further extended by Faratin, et al in [8]. The paradigm proposed for multi-issued negotiation is that of value tradeoffs. The agents negotiate over values for a set of what can be termed as *preferentially independent* issues [7, p. 101].

When two issues are preferentially independent, it should be possible for the agent, by definition, to assert its preferences for an increasing or decreasing value of one issue without any relation to the other issue. When there are more than two issues, preferential independence is similarly defined for every subset of the set of issues and its complement. The set of issues for negotiations between any two entities usually revolve around some notion of $\{quantity, quality, unit\ price\}$. It might seem that some of the issues could be combined together into a single quantity, like multiplying unit price and quantity, to generate a single issue of total cost. However, this precludes the possibility of keeping the quantity fixed and bargaining only over the unit price. As proposed in [8], agents negotiate on the value of an issue that is within a delimited range. That is, the agents first mutually agree on a range of allowable values for every issue, for example to negotiate a value of the quantity between 200MW and 300MW.

In a multiagent system A consisting of at least two agents, an agent a wants to negotiate a value for a set of issues with agents in a subset of $A - \{a\}$; let $X_a = \{x_1, x_2, \dots, x_n\}$ ¹ be the set of issues about which agent a wants to negotiate. The set X_a is called a *negotiation set*. The agent uses a non-decreasing or non-increasing scoring function² $V(x)$ to score the value of each issue between 0 and 1³. Non-decreasing or non-increasing functions serve to enforce transitive preference structures. If the agent prefers an outcome x' to x'' for a single issue, then $V(x') > V(x'')$; if the agent is indifferent between two outcomes x' and x'' , then $V(x') = V(x'')$. *MASPOWER* uses a model of additive value functions [7, p. 90] to get the net value of a negotiation set. The agent assigns relative importance (weight) to each issue in the negotiation set; w_i is the relative importance of issue x_i to the agent, such that the

¹ The subscript "a" used to identify the agent is omitted when the discussion involves only one agent.

² The terms "scoring functions" and "value functions" mean the same, hence they are used interchangeably.

³ " x " is used to denote the negotiation issue and also the value for that issue whereas $V(x)$ denotes the score for that value of the issue. For example, an issue x can have a value of 50 when it is defined in the range [0,100] with have a score of 0.6.

weights are normalized:

$$w_i \geq 1 \quad w_i \geq 0 \quad (8)$$

In simple cases, finding the relative importance of issues could be based on a subjective assessment by the human owner. The agent a 's scoring function for the negotiation set is defined as:

$$V^a(X) = \sum_{i=1}^n w_i * V(x_i) \quad (9)$$

The additive scoring function as defined above is the simplest multi-issued value function with useful properties: two agents using additive scoring functions are sure to maximize social welfare between them⁴. Essentially, agents agreeing to negotiate on a set of issues have to agree on a single mutually acceptable value for each issue. Since the negotiation is an action performed by the agent, it is implemented as a *task* according to the definition in Section II. The next step in applying this framework is to generate a value for a single issue.

Negotiation functions serve to generate a value of a single negotiation issue. Although negotiation functions can be complex functions of the internal states of the agent, the current version of *MASPOWER* uses the notion of resources to model this dependency [8]. Resources that the agents could use in their decision-making criteria can include time to complete the negotiation, money that the agent has at its disposal, equipment usage, and so on. If the agent wants to negotiate on an issue, say the money it is going to receive for a particular service, the agent's valuation of this issue is modeled solely a function of its resources. In other words, the value of an issue for an agent, at any instant during the negotiation, does not directly depend on the offers that this agent receives from other agents. Therefore, from a game theoretic perspective, the agent does not behave strategically in response to bids from other agents. The agent uses the current state of its resources to determine a value for an issue. For an issue x that depends only on resource r , a function is defined as follows:

$$x = f(r) \quad (10)$$

This function sets the value of the issue x based only on the current state of the resource r . Although the function $f(\cdot)$ can be used to model complex dependencies, the current implementation of *MASPOWER* uses a family of parameterized functions for this purpose. The implementation details of these functions are given in [10]. When the issue depends on multiple resources, the dependence is modeled as:

$$x = f(r_1, r_2, \dots, r_n) = \sum_{i=1}^n w_i * f(r_i),$$

where $\sum_{i=1}^n w_i = 1$ and $w_i \geq 0$ (11)

The agent's decision-making algorithm under this negotiation paradigm is based on its score for the negotiation set.

⁴ Therefore, for a negotiation process that has successfully concluded, the outcome of the negotiation is *pareto optimal* for the agents. The proof is straightforward and is available in [9, p. 164].

Suppose agent a 's private scoring function for a negotiation set X at the time instant t is $V^a(X)$. This score is purely based on the internal state of this agent and is a function of its value functions, tradeoffs, current state of resources, and negotiation functions. When agent a receives an inter-agent message containing an offer X' from another agent for the same negotiation thread at a time $t' > t$, it requires a decision-making procedure; such a procedure is shown in Fig. 1. This procedure is the basic structure used by the agent as constrained by the implementation of the platform and the negotiation paradigm; further constraints can be enforced by the conversation protocols. It should be noted that this implementation of a value-based framework is a stepping stone to utility-based decisions, where we accommodate uncertainty in the consequences and the risk preferences of the human decision-maker.

IV. IMPLEMENTATION OF MASPOWER

Considering the functionalities desired of agents described previously, it is evident that instantiating such software entities could lead to considerable complexity in design and implementation. Our approach employs object-oriented software design methodology and develops an abstract *generic agent* that has most of the desiderata of software agents. Software agents for specific functions can be implemented by extending the *generic agent*. The *generic agent* and the associated infrastructure, all implemented using the Java programming language [11], provide most of the functionalities required by software agents like managing multiple concurrent tasks, managing multiple conversation threads, perceptors for accessing local and remote percept sources, and receiving messages from remote agents. Java was chosen because of its inherent distributed programming support, ease of programming, safety features and support for multithreading. Many common programming bugs in large programs do not occur in Java because of automatic garbage collection and type-safe references. The distributed computing components of *MASPOWER* are engineered by using the functionalities provided by Voyager ORB, Version 3.2 [12]. Voyager ORB is a high-performance object request broker that simultaneously supports RMI, CORBA and DCOM. Its innovative dynamic proxy generation, naming service, synchronous and asynchronous messaging support simplifies the development of a distributed multiagent system. Details of the design have been documented using UML notations; descriptions of the most important methods are in [10].

V. A NEGOTIATION EXPERIMENT

Numerous decision-making contexts in deregulated power systems can be cast in the framework of Section IV. Several such decision-making contexts have been described in [10]. In the rest of this section, we describe and implement one such negotiated decision-making context and provide preliminary

```

if (no appropriate task is defined for handling
this message)
    reply (not-understood)

else if (no "handler" is defined for this per-
formative)
    reply (not-understood)

else if (time for completing this negotiation
has elapsed)
    reply (refuse)

else {
    if ( $V^a(X) \leq V^a(X')$ )
        reply (accept,  $X'$ )
    else
        reply (propose,  $X$ )
}

```

Fig. 1. Basic Decision-Making Procedure

results.

In the electric network-operating environment of the past, decision-making authority was centralized, as the transmission owner and the system operator were the same company. This is no longer the case in the U.S.A, as the Federal Energy Regulatory Commission (FERC) has mandated that the system operator exists as an organization-independent neutral entity having operating authority but no equipment ownership. This mandate is mainly to motivate unbiased use of this authority. However, this arrangement fragments certain types of decisions that require consideration of equipment integrity (of interest to the transmission owner) and system integrity (of interest to the operator). We desire to explore one such decision-making problem in this context in order to illustrate the manner in which multiagent systems may be applied and to identify the benefits and drawbacks for doing so. The decision problem is a familiar one: how much do we operate a transmission circuit in excess of its identified rating, where by doing so we incur the following influences:

- (i) The energy supplied to the load is less expensive
- (ii) The security-level of the system is lower, meaning that the system is at higher risk with the additional flow. In this work, we measure system security as a function of the number of (N-1) overload violations.
- (iii) The revenues from transmission usage are higher.
- (iv) The circuit itself may incur some loss of life through the higher operating conditions requiring expending resources to perform unscheduled maintenance and overhauls to extend equipment life.

With centralized decision-making in vertically integrated utilities, these influences are assessed together by a single decision-maker, the operator, as they are all of significance to the operator's organization. The decision made is based on a desire to find a balance between these four influences that is most attractive to the operator's organization. Under fragmented decision-making authority, however, the operator considers only the first two influences, the remaining are of no significance to his organization. On the other hand, the transmission company considers only the last two issues. Yet, from a societal point of view, all four influences should be included in the decision-making. Casting the problem and the decision-making entities into a multiagent system framework enables this.

This negotiation scenario involves the interests of two agents, viz., the independent system operator agent (ISOA) and the transmission company agent (TA). By definition, the ISOA does not have any profit motivation while its responsibility is to implement all the transactions coming from the markets with minimal alterations while not violating any system-level security constraints. However, the ISOA cannot violate the normal transmission limits of the line without the consent of the concerned TA. We simplify the decision-making responsibilities of these entities by asserting the following roles:

- *ISOA*: The system level security constraints considered are emergency limit violations on transmission lines under loss of a single component; i.e., a *N-1* contingency analyses.
- *TA*: The TA increases its revenues as it accepts more flows on its lines, but at the expense of reduction in equipment life. The TA may have to spend money for unscheduled maintenance and overhauls to increase equipment life. Therefore the TA must decide the tradeoff between accepting flows beyond the normal limits and loss of equipment life.

Both agents are instantiated using *MASPOWER*. The ISOA uses money and time as its resources whereas the TA uses money, time, and equipment life of the concerned transmission line(s) as resources. The details of this experiment are provided in [10]. The agents use the 24-bus RTS as the physical system. We generated a dispatch schedule that created an overload of 10.1 MW on the line between buses 14 and 16. The TA and ISOA negotiate on whether this excess flow can be allowed in the line for a suitable compensation. The agents agree that the allowable interval for the overload (MW) is [0,10.1]; and compensation (\$/MW) is [5,20]. This compensation can be considered as transmission service revenues. Although these revenues might very well originate with the generation or load-serving entities, we assume without loss of generality that the ISOA has the authority to negotiate it. The conversation protocol used by the agents and enforced by *MASPOWER* is shown in Fig. 2 using COOL notations. These

notations have been described in [5]. The agents iteratively exchange proposals, and finally agree on an increase in flows by 5.54 MW for a 11.54 \$/MW in compensation. The agents exchange 42 proposals over a span of 225 seconds. The score of this outcome to the ISOA is 0.536 and TA is 0.474. The progress of the negotiation is shown in Fig. 3. The drawing convention used in Fig. 3 showing the progress of the negotiation is as follows: dark points are used to plot the score for the agent's private value of the negotiation set, whereas lighter points are used to plot this agent's score for the offer it as received from the other agent.

Some of the preferences of the human decision-maker is

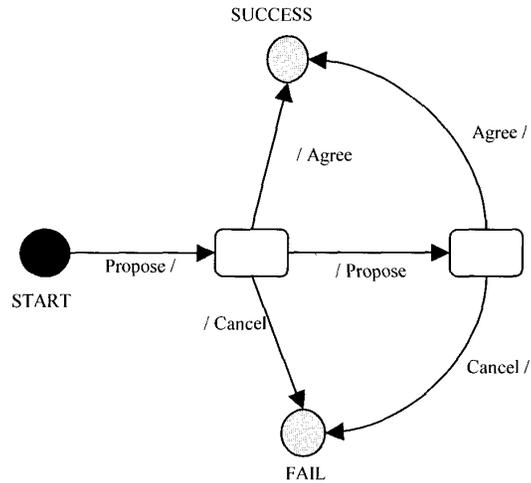


Fig. 2. Negotiation Protocol in COOL Notations

encoded in the agent as value functions and tradeoff values. The decision made by the software agent reflects these preferences. It is evident that negotiation proposals and the score of the outcome to the self-interested agents depend on the internal states of the agents. We have conducted several negotiation simulations to interpret the outcome of the negotiation as a function of the internal states. Details of these simulations are available in [10], and the results will be reported in a future publication.

VI. CONCLUSIONS

In this paper, we have described a design of *MASPOWER*, a Java-base API to instantiate complex agents and multiagent systems. We described the value-based negotiation model that has been implemented as part of *MASPOWER*. In section V, we described a negotiated decision-making scenario between the transmission company and the system operator

and provided preliminary results of the negotiations. We are presently engaged in conducting more such simulations and interpreting the results of this simulation as a function of the internal states of the agents. An additional dimension of our current effort is to provide a richer abstraction of the electric power system and its security levels to the agents. For achieving this, we are working to integrate a sophisticated long-time power system simulator [13,14,15,16,17] and risk-based security assessment programs [18,19] with *MA-SPOWER*. The simulator will serve as the environment for the agents and risk-based security assessment capability will enable the agents to quantify the security of the system or equipments considering the current state and the uncertainty in various operating parameters.

VII. ACKNOWLEDGEMENTS

This research has been supported by the EPRI and Department of Defense through the Complex Interactive Systems/Networks initiative (Grant WO8333-01), and National Science Foundation (Grant 0087152).

VIII. REFERENCES

- [1] V. Honavar, L. Miller and J. Wong, "Distributed knowledge networks," In *Proc. of the IEEE Information Technology Conference*, Syracuse, NY, 1998.
- [2] G. Helmer, J. Wong, V. Honavar and L. Miller, "Intelligent agents for intrusion detection," In *Proc. of the IEEE Information Technology Conference*, Syracuse, NY, 1998.
- [3] J. Yang, V. Honavar, L. Miller and J. Wong, "Intelligent mobile agents for information retrieval and knowledge discovery from distributed data and knowledge sources," In *Proc. of the IEEE Information Technology Conference*, Syracuse, NY, 1998.
- [4] M.P. Georgeff and A.S. Rao, "A profile of the Australian AI institute," *IEEE Expert*, vol. 11-6, pp. 89-92, 1996.
- [5] V. Vishwanathan, V. Ganugula, J. McCalley, and V. Honavar, "A multiagent systems approach for managing dynamic information and decisions in competitive electric power systems," *Proc. of the 2000 North American Power Symposium*, Oct. 2000, Waterloo, Ontario.
- [6] J. McCalley, V. Vittal, N. Abi-Samra, "An overview of risk-based security assessment," In *Proc. of IEEE Power Engineering Summer Meeting*, vol. 1, pp. 173-178, 1999.
- [7] R.L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, Cambridge University Press, 1993.
- [8] P. Faratin, C. Sierra and N. Jennings, "Negotiation decision functions for autonomous agents," *Int. Journal of Robotics and Autonomous Systems*, Vol. 24:3-4, pp. 159-182, 1997.
- [9] H. Raiffa, *The Art and Science of Negotiations*, Harvard University Press, 1982.
- [10] V. Vishwanathan, "Multiagent system applications to electric power systems: Negotiation models for automated security-economy decisions," M.S. thesis, Dept. Elec. and Comp. Eng., Iowa State Univ., Ames, 2001.
- [11] K. Arnold, J. Gosling and D. Holmes. *The Java™ Programming Language* (3e), Addison Wesley, 2000.
- [12] Objectspace Inc. Voyager ORB homepage: <http://www.objectspace.com/products/voyager/>.
- [13] Y. Dai, J. McCalley, V. Vittal, "Simplification, Expansion, and Enhancement of Direct Interior Point Algorithm for Power System Maximum Loadability," *IEEE Transactions on Power Systems*, Vol. 15, No. 3, Aug., 2000, pp. 1014-1021.
- [14] Y. Dai, J. McCalley, V. Vittal, "Annual risk assessment for overload security," to appear in *IEEE Trans. On Power Systems*.
- [15] Y. Dai, J. McCalley, V. Vittal, and M. Bhuiyan, "Annual risk assessment for voltage stability and generation adequacy," *Proceedings of the VI International Conference on Probabilistic Methods Applied to Power Systems*, September, 2000, Madeira Island, Portugal.
- [16] Y. Dai, J. McCalley, V. Vittal, "A heuristic method to arrange unit commitment for one year considering hydro-thermal coordination," *Proceedings of the 1998 North American Power Conference*, pp. 382-387, Cleveland, Ohio, Oct., 1998.
- [17] Y. Dai, J. McCalley, and V. Vittal, "Annual risk assessment for thermal overload," *Proceedings of the 1998 American Power Conference*, Chicago, Illinois, April, 1998.
- [18] H. Wan, J. McCalley, V. Vittal, "Risk-based voltage security assessment," *IEEE Transactions on Power Systems*, vol. 15-4, pp 1247-1254, 2000.
- [19] W. Hua, J. McCalley, V. Vittal, "Increasing thermal ratings using risk analysis," *IEEE Transactions on Power Systems*, vol. 14- 3, 1999.

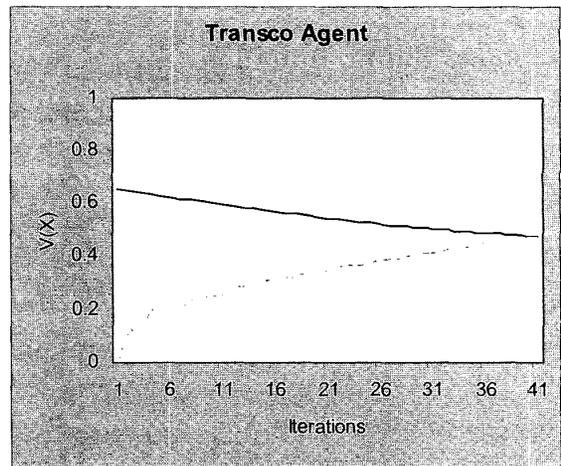
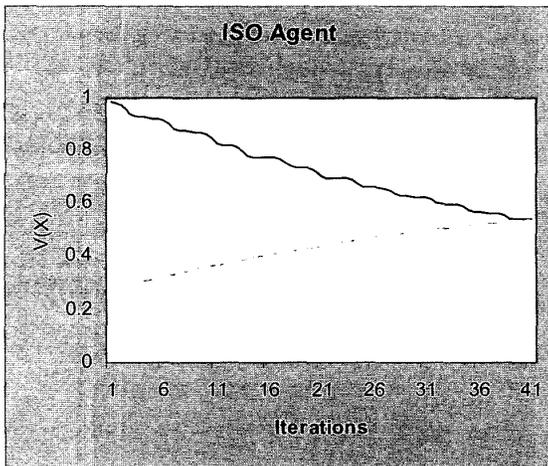


Fig. 3: Progress of negotiation between software agents representing the ISO and Transco