A Multi-Agent Based Negotiation Support System for Distributed Transmission Cost Allocation

Yonghe Yan, Jerome Yen, and Tung X. Bui

1 Department of Computer Science & Information Systems
The University of Hong Kong, Pokfulam Road, Hong Kong
yhyan@csis.hku.hk

2 Department of System Engineering & Engineering Management
Chinese University of Hong Kong, Shatin, New Territories, Hong Kong
jyen@se.cuhk.edu.hk

3 College of Business Administration, The University of Hawaii at Monoa,
Hololulu, Hawaii, USA
tbui@busadm.cba.hawaii.edu

Abstract

In the paper we developed a multi-agent system, based on network flow model and KQML, called MASCAN—Multi-Agent System to support negotiation of Cost Allocation on Network. Such problem is very important to the utility industries, such as, electricity and gas, transportation and logistics industry. In our system, each agent represents a node in a network, it does not receive any centralized controls or centralized information sources. In which all the decisions were made locally based on the rules or knowledge that each agent has in order to contend with other agents for a possible cheapest path under a fair-play practice. We also assume that each agent is rational. The solution to the cost allocation is the equilibrium point of a noncooperative game subject to the constraints of the given network. In this study we applied MASCAN to model and support the negotiation of cost allocation of power transmission after the deregulation of electricity industry.

1. Introduction

In recent years, utility industries in several countries have been forced to undertake deregulation to introduce competitiveness in their generation, transmission, and distribution. Cost allocation methods based on loading flow of power transmission, such as MW-mile [4], have been criticized as having no obvious grounding on economic theory [2, 5]. There were also other analytical approaches to support cost allocation [17, 18, 19]. All these methods did not consider the behavior of each participant by studying his/her generation/load pattern during the calculation of transmission path (circuit). We should develop a method that can help each individual to make decision individually in order to determine the transmission path that each participant is willing to share.

Based on the marginal cost of power generation and transmission, an operational model of generation and transmission can be represented by a network flow model [3]. Therefore, we adopted the network flow model to determine how to allocate the costs of generation and transmission in power industry. After deregulation of power industry, all generation and transmission networks at all level should be opened to all the participants, such as, owner of generation stations and groups of consumers. Access should be granted based on a payment of a point-of-connection.

Furthermore, all services are unbundled from generation and electricity sales [12]. Therefore, network flow model can be used to study the generation and transmission services. Based on network flow model, behaviors of supply/customer (generation/load) can be investigated in the cost allocation of generation and transmission service.

Although network flow decomposition determines the transmission path for each agent, which represents a node in a given network, of generation/load, the path can not uniquely be determined by network flow decomposition algorithm [3]. Hence transmission paths that given by network flow decomposition algorithm will not be accepted by individuals that access the network. The decision of each agent represents an individual would affect the flow path during the network flow decomposition. Therefore we need a multi-agent system to solve the problem.

In this paper, we proposed and have developed a multi-agent system to analyze the insight behaviors and
intuitions of each agent which represents a participant, such as a generation or a load, of the transmission network. The multi-agent system simulates the power industry and models each participant as an agent. In this system, agents communicate with each other to search for potentially the cheapest path as the base to determine their transmission costs. For rational customers, they search for the cheapest (shortest) possible paths that satisfy their demands.

Agents of this system have to work collaboratively to finish certain tasks, for example, determining the possible transmission lines to reduce the overall costs. Agents are assumed to be rational, that is, maximizing its own utility. Each agent is also assumed to be independent and autonomous, who will not accept any plan that generated by a centralized planner. We also assume that no agent has the global information about the network. Each agent has only the information about himself and of his immediate neighbors. The reason is that ownership of global information may impose or favor certain centralized control. Owner of the global information may have certain advantage over the others and it is difficult to maintain totally neutral. It is the situation that we must avoid.

In the rest of the paper, we will first introduce the software agent and multi-agent system in section 2. In section 3, we will discuss the minimum cost flow problem as base on which MASCAN agent works. The multi-agent approach on network flow will be given in section 4, and some issues about the implementation and complexity of MASCAN are discussed in section 5. Finally in section 6 is given the discussion.

2. Software Agent and Multi-agent System

With the advances in IT, growing complexity and decentralization of the utility markets, and the increasing pressure to lower the costs has pushed the demand for new tools or systems to remove the burdens of human decision makers from those tedious and repeating tasks. One of such applications is the software agent. Franklin and Graesser proposed a “mathematically formal” definition of agent: An autonomous agent is a system situated within and, as a part of an environment, that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future [6]. Detailed discussion about software agents can also be found in [7, 8, 10, 11].

Many definitions of agent, such as those advocate by Franklin and Graesser, are based on Wooldridge and Jennings’ [7] notion of “strong” AI-based agents: agents with “beliefs”, “desires”, and “intentions”, for example. However many of these agents that have been developed for engineering applications are of the “weak” notion in the sense that there is no commitment to powerful reasoning by the individual agents [8]. We also believe that agents should work together to form the agent community. Which is a multi-agent system (MAS) that agents communicate with each other in order to accomplish a task, and agents are autonomous, proactive, and adaptive [13]. In a multi-agent system, each agent holds some information about the environment and can apply certain actions to affect the environment.

The agents in our system are owners of power stations and groups of customers, which represent nodes in a given network. For simplicity, each node is represented by an agent, which represents the supply/demand at that node. Therefore the objective of the proposed multi-agent system is to find a possible path for each supply/demand with the lowest costs under the fair-play practice subject to the given constraints.

Decisions of each agent on the network flow may affect the flow path. That is, in order to reach an optimal condition, the action at one point may depend on not only the current state of the network, but also decisions to be made by other agents and states occurred in earlier states. So communication and cooperation are two most important capabilities to the multi-agent systems. Multi-agent systems are designed to have the capability to either collaborate, for example, decompose a problem and jointly solve the problem, or contend, such as searching for the best deals for the users. The agents cooperation used in this paper is assumed to include both collaboration and competition.

Communication is vitally important by which relevant information to support cooperation is exchanged. KQML (Knowledge Query and Manipulation Language) is a language that supports communication among agents [14]. KQML is both a message format language and a message-handling protocol to support run-time knowledge sharing among agents. KQML can be used as a language for an application program to interact with an intelligent system or for two or more intelligent systems to share knowledge. KQML was among the first agent communication languages to be developed and used. Moreover, it is the only one to date that has enjoyed substantial use by people other than its developers [20]. KQML was therefore selected as the agent communication language among agents in our implementation of MASCAN -- the Multi-Agent System for Cost Allocation on Network.

One key idea in multi-agent system is the emergent functionality. The functionality of an agent is viewed as an emergent property of the intensive interaction of the system with its dynamic environment [9]. The goal pursued by agents in MASCAN is to find the possible shortest-path flows to satisfy the demand/supply of agents with constraints of the network. When each agent achieved its goal, they also jointly solved the minimum cost flow problem of the network. It shows that pursuit of local utility maximization of each agent will finally
achieve global utility maximization for the community of agents. MASCAN agents are actually played a non-cooperative game on network. The cost allocated to each agent is the equilibrium point of the game. We will elaborate these points in the following sections.

3. Minimum Cost Flow Problem

If all the agents are rational\(^1\), the total cost allocated to supply or demand nodes will equal the minimum cost of the given network, when supply or demand of each node in the network is satisfied. Therefore, minimum cost flow problem is essential to the cost allocation on network flows. We follow Ahuja, Magnanti and Orlin [3] to define the minimum cost flow problem.

Let \( G = (N, A) \) be a network defined by a set \( N \) of \( n \) nodes and a set \( A \) of \( m \) arcs. Each arc \((i, j) \in A\) has an associated cost \( c_{ij} \) that denotes the cost per unit flow on that arc. The flow cost varies linearly with the amount of flow. We also associate with each arc \((i, j) \in A\) a capacity \( u_{ij} \) that denotes the upper bound of the arc and a lower bound \( l_{ij} \) of the flow on the arc. We associate with each node \( i \in N \) an integer number \( b(i) \) representing its supply or demand. If \( b(i) > 0 \), node \( i \) is a supply node; if \( b(i) < 0 \), node \( i \) is a demand node with a demand of \(-b(i)\). If \( b(i) = 0 \), node \( i \) is a transshipment node. The decision variables in the minimum cost flow problem are arc flows. We represent the flow on an arc \((i, j) \in A\) by \( x_{ij} \). The minimum cost flow is an optimization problem that formulated as follows:

\[
\text{Minimize } \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{1}
\]

subject to
\[
\sum_{j \in \text{successors of } i} x_{ij} - \sum_{j \in \text{predecessors of } i} x_{ji} = b(i) \text{ for all } i \in N, \tag{2}
\]
\[
l_{ij} \leq x_{ij} \leq u_{ij} \text{ for all } (i, j) \in A, \tag{3}
\]

where \( \sum_{i \in N} b(i) = 0 \), which means that the total supplies always equal total demands. For simplicity, we ignore the losses during transmissions.

In constraint 2, the first term represents the total outflows of the node and the second term represents the total inflows of the node. The constraint states that the outflow minus inflow must equal the supply/demand of the node. If the node is a supply node, its outflow exceeds its inflow; if the node is a demand node, its inflow exceeds its outflow, and if the node is a transshipment node, its outflow equals its inflow. The flow must satisfy the lower bound and capacity of constraints 3. Without losing generality, we assume that the lower bounds on arc flows are zero and \( u_{ij} \) is the capacity of arc \((i, j)\). It is obvious that the cost \( c_{ij} \) represents the marginal cost of either a generation or a transmission line.

In this paper, we assume that all the data are integer, i.e., all arc capacities, arc costs, and supply/demands of nodes are integer. The integrality assumption is not restrictive for most applications because we can always transform rational data to integer data by multiplying them by a suitable large number. Moreover, we necessarily need to convert irrational numbers to rational numbers to represent them in a computer.

If \( x \) is the optimal flow of minimum cost flow problem for a given network under the condition that we relax constraint 2. We obtain \( x' \) from \( x \) by sending flow \( \delta \), \( \delta > 0 \), along a shortest path, i.e. the lowest cost path, from node \( s \) to some other node \( t \); and then \( x' \) is also the optimal flow for the relaxed network\(^2\). Note that \( x = 0 \) is always the optimal flow for any relaxed network.

If flow \( \delta \) is sent along a shortest path \( P \) from node \( s \) to node \( t \), cost is then allocated to nodes \( t \) should be

\[
U_t(\delta) = \sum_{(i,j) \in P} f_{ij} c_{ij} \delta \quad (4)
\]

where \( f_{ij} \) is the direction factor of the path flow on arc \((i, j)\). If the net arc flow is from node \( i \) to node \( j \) (i.e., \( x_{ij} \geq 0 \)) and the path flow \( \delta \) is sent from node \( i \) to node \( j \), \( f_{ij} \) have value of 1; whereas the path flow \( \delta \) is sent from node \( j \) to node \( i \) along arc \((i, j)\) that the net arc flow from node \( i \) to node \( j \) is not less than \( \delta \) (i.e., \( x_{ij} \geq \delta \)), the value of \( f_{ij} \) is -1. It implies that sending a unit flow from node \( i \) to node \( j \) on the arc \((i, j)\) increases the net flow cost by \( c_{ij} \) units. Similarly, sending flow from node \( j \) to node \( i \) on the same arc decreases the flow cost by \( c_{ij} \) units since the flow decreases the net flow on the arc, and we are saving the cost that we used to incur in sending the flow from node \( i \) to node \( j \).

While each node is represented by an autonomous agent, the shortest path \( P \) and the flow \( \delta \) must be approved by all the agents along the path \( P \). During the process, agents need to communicate and cooperate among each others. Moreover, an agent can have its own plan that may affect path and flow amount. When a flow is determined, the network is then changed, i.e., the environment changed when some actions are conducted by agents. We will discuss the behaviors of agents of MASCAN in next section.

4. Multi-agent Approach on Network Flow

In this section, we discuss the Multi-Agent System to support negotiation of Cost Allocation on Network, in short MASCAN. In MASCAN there is no agent who has

\(^1\) In this paper, a node in the terminology of network is represent by an agent. We use these two terms exchangeable.

\(^2\) See [3], pp. 306-321, for a detail proof.
The algorithm is also called Bellman-Ford algorithm in [15].

4.1. Multi-agent Approach on Shortest Path Problem

Shortest path problem can be solved by a few algorithms [15]. But they are all centralized algorithms that aim to optimize the costs of the entire network, which may request some agents to make sacrifice. Based on label-correcting algorithm\(^3\) [3], we can solve shortest path problem by multi-agent approach. In our multi-agent system, a node is represented by an autonomous agent, which has the information of the arcs that connect to it. Agents communicate with neighbors to correct the violation of shortest path optimality condition. The shortest path optimality condition is given by formula (5). The shortest path is found, when no agent finds an arc that is incident to it violating the shortest path optimality condition.

Agent maintains a distance label \(d(i)\) at every stage. The label \(d(i)\) is either \(\infty\), indicating that the agent has yet to discover a path from supply agent to agent \(i\), or the length of the path from the source to agent \(i\). For agent \(i\) it maintains a predecessor index, \(\text{pred}(i)\), which records the agent prior to agent \(i\) in the current path of length \(d(i)\).

At termination, the predecessor indices allow an agent to trace the shortest path from a supply agent back to agent \(i\). Numbers \(d(\cdot)\) represents distance of shortest path if and only if it satisfies the following shortest path optimality conditions:

\[
d(j) \leq d(i) + f_{ij}c_{ij} , \text{ for all } (i, j) \in A
\]  

A summary of the steps in the process of searching shortest path by multi-agent approach is given as follows:

**Initialization:**

1. Agent collects information of arcs that connect to it. If agent \(i\) is a supply node, set \(d(i) : = 0\). If agent \(i\) is not a supply agent, set \(d(i) : = \infty\).

2. Announce a message “end of initialization.” The message is used to synchronize the initialization of agents.

3. When all the agents have announced “end of initialization”, announce a message “violation of shortest path optimality condition.” The message informs all agents to begin shortest path calculation.

**Shortest path calculation:**

Calculation of the shortest path optimality condition must be done whenever agent receives from any other agents the message “violation of shortest path optimality condition.” The calculation of agent \(j\) is as follows:

1. Inform every neighboring agents to “send me your \(d(\cdot)\)”. Agent \(i\) is defined as the neighboring agent of agent \(j\) if there is an arc \((i, j)\) and the arc flow \(x_{ij} < u_{ij}\). Notice that if \(x_{ij} > 0\) on arc \((i, j)\), then \(x_{ji} = -x_{ij} < u_{ij}\) and agent \(i\) is also the neighboring agent of agent \(j\).

2. Check shortest path optimality condition: whenever agent \(j\) received distance label of neighboring agent, \(d(i)\), from neighboring agent \(i\), which can be any neighboring agent of agent \(j\); if \(d(j) > d(i) + f_{ij}c_{ij}\), violation of shortest path optimality condition for agent \(j\) is found. Then agent \(j\) must announce a message

\(\text{message } \text{“violation of shortest path optimality condition”}\).
4.2. Multi-agent Approach on Cost Allocation

When a shortest path has found, agents can begin negotiation with agents along the shortest path to augment a flow along the shortest path, that is, to send a flow from a supply node to a demand node along the shortest path between the two nodes. Initially, we assume all arc flows are zero. For any flow \( x \), we define the imbalance of agent \( i \) as

\[
e(i) = b(i) + \sum_{j \in N} x_{ji} - \sum_{j \in N} x_{ij} \quad \text{for all } i \in N
\]

(6)

where \( N \) is the set of nodes of the network, and \( b(i) \) is the amount of supply/demand of agent \( i \). The negative value of \( b(i) \) represents the amount of demand, and vice versa.

If \( e(i) > 0 \), \( e(i) \) is an excess of agent \( i \); if \( e(i) < 0 \), we call \( -e(i) \) the deficit of agent \( i \), and we refer an agent with \( e(i) = 0 \) as balanced. Notice that

\[
\sum_{i \in N} e(i) = \sum_{i \in N} b(i) = 0,
\]

and hence

\[
e(i) = -e(i), \quad \text{where } E \text{ and } D \text{ the set of excess agents and the set of deficit agents respectively. Consequently, if the network contains a deficit agent, it must also contain an excess agent.}
\]

In MASCAN, agents along the shortest path \( P \) between excess node \( k \) and deficit node \( l \) need determine cooperatively an augmenting flow with amount of

\[
\delta = \min\{e(k), -e(l), \min\{r_q : (i, j) \in P\}\}
\]

(7)

where \( r_q = u_q - x_q \), \( (i, j) \in P \), \( x_q \) is the current arc flow of \( (i, j) \); or \( r_q = x_{ji} \), \( x_{ji} > 0 \) is the current arc flow of \( (i, j) \). Notice that when \( r_q = x_{ji} \), \( f_{ji} = -1 \).

If an agent is on the shortest path of other agents, at each time the agent can only allow one flow to be augmented through the agent. Also notice that path flow of a given problem is not unique. These imply that agent can choose different strategies to select the augmented flow that requests to pass through the agent.

When an agent is the deficit agent, the agent can determine augmenting flow by selfish plan. With the selfish plan, an agent will only agree to augment the flow that can help him decrease the deficit of himself before the deficit of the agent becomes zero. After the agent becomes a balanced agent, i.e., the deficit of the agent is zero, the agent may change to execute other plan that may let the agent agree to augment a flow that pass through himself. Therefore the selfish plan will only allow inflows and do not allow outflows. By unselfish plan, an agent agrees at first to augment a flow that is requested by his succeeding agents, and will require to decrease the deficit of himself after all the requests of his succeeding agents have been satisfied. Unselfish plan is not rational, therefore, we will not further discuss it.

For deficit agent and balanced agent, it can choose modest plan that the agent will not make any decision on the selection of requests of augmenting flows, and it only transmits requests to its preceding agent. For an excess agent, it is more likely to take up egocentric plan. When an agent received more than one augmenting flow requests, with egocentric plan, the agent can make decision according to the distances and/or the amounts of the requests. An agent can adopt a plan and change plan independently. For example, an agent may execute selfish plan when it is a deficit agent; and change to modest plan when it become balanced agent.

When all imbalances of agents become zero, the transmission costs are allocated to agents. A summary of the steps of MASCAN is given as follows:

**Initiation:**

\[\text{Let all the arc flows on the arcs that connect to agent } j \text{ be zero, and } e(j) = b(j).\]

**Shortest path flow negotiation:**

**Request process of Selfish plan for agent } j:**

1. If agent } j is a deficit agent, announce shortest path calculation. When all the agents are ready, start shortest path calculation. When the shortest paths are found, go to step 2.

2. Let agent } i be the predecessor agent on the shortest path between an excess agent to agent } j. Request agent } i to approve "send } \delta_j \text{ unit flow along path } (i, j)." \( } \delta_j = \min(-e(j), r_q) \).
3. If agent i replies that "δ unit flow can be sent along path P" and agent j is on the end of path P, agent j has gotten δ unit flow along path P. Then update $x_{ij}$ and $e(j)$, i.e., $x_{ij} := x_{ij} + \delta$ and $e(j) := e(j) + \delta$.

4. If agent is still a deficit agent, announce "agent j is ready to go to next iteration of shortest path calculation." Go to step 1.

Reply process of Selfish plan for agent j:
1. If agent j is a deficit agent and received a request from agent i, "send $\delta_i$ unit flow along path P", reply a rejection to agent i.
2. If agent j becomes a balanced agent, execute either reply process of modest plan or reply process of egocentric plan.

Request process of modest plan for agent j:
1. If agent j is a deficit agent, announce shortest path calculation. When all the agents are ready, start shortest path calculation.
2. If agent j is a deficit agent, let agent i be the predecessor agent on the shortest path between an excess agent to agent j: request i to approve "send $\delta_j$ unit flow along path (i, j)," and $\delta_j = \min(-e(j), r_{ij})$.

3. If agent i replied that "δ unit flow can be sent along path P" and agent j is on the end of path P, agent j has gotten δ unit flow along path P. Then update $x_{ij}$ and $e(j)$, i.e., $x_{ij} := x_{ij} + \delta$ and $e(j) := e(j) + \delta$.

4. If agent is still a deficit agent, announce "agent j is ready to go to next iteration of shortest path calculation." Go to step 1.

Reply process of modest plan for agent j:
1. When agent j received a request, "send $\delta_k$ unit flow along path P", from agent k, and let agent i be the predecessor agent on the shortest path between an excess agent and agent j. Agent j requests agent i to approve "send $\delta_j$ unit flow along path (i, j)," $\delta_j = \min(\delta_k, r_{ij})$. Notice that more than one requests can be received from the neighboring agents of agent j.

2. If agent i replies that "δ unit flow can be sent along path P"); agent j replies agent k, who is the succeeding agent of agent j within the path P, a message "δ unit flow can be sent along path P". Update $x_{ik}$ and $x_{jk}$, i.e., $x_{ik} := x_{ik} + \delta$ and $x_{jk} := x_{jk} - \delta$.

3. If a rejection is received from agent i, agent j then forwards the rejection to agent k according to path P.

Reply process of egocentric plan for agent j:
1. If agent j is an excess agent, agent j can execute egocentric plan. No request process is needed in egocentric plan. When agent j received requests "send $\delta_k$ unit flow along path P"; agent j can choose an agent $k'$ among the request agents according to some criteria; reply agent $k'$ that "δ unit flow can be sent along path P," $\delta = \min(\delta_k, e(j))$; and reject all other requests. Update $x_{jk'}$ and $e(j)$, i.e., $x_{jk'} := x_{jk'} - \delta$ and $e(j) := e(j) - \delta$.

2. If agent j is an excess agent, agent j can execute the selfish plan when they are deficit agents. The egocentric plan is executed by excess agents and balanced agent. Under this circumstance, the costs allocated to balanced agents, deficit agents will therefore request to augment flows through the shortest path from excess agents. For one excess agent only one augment flow will be granted in each iteration.

Computation of MASCAN is driven by request process. In any iteration, after all agents are ready, request process will first start the computation of shortest path, deficit agents will therefore request to augment flows through the shortest path from excess agents. For one excess agent only one augment flow will be granted in each iteration.

In MASCAN, the network flow will also be the minimum cost flow of the network. If all agents are rational, agents will execute the selfish plan when they are deficit agents. Modest plan will be executed for those balanced agents. The egocentric plan is executed by excess agents and balanced agent. Under this circumstance, the costs allocated to balanced agents and supply agents will be zero, and the cost allocated to a demand agent, such as agent $t$, will be

$$U_i = \min \sum_{k} U_i(\delta_k)$$

(8)

for all $\delta_k$ subject to $\sum_k \delta_k = -b(t)$.

---

4 We will discuss it later.
Based on game theory, it is assumed that behaviors of players (agents) are rational. That is, they wish to obtain the maximum payoff (minimum cost), assuming that their competitors act the same. This is exactly the behavior of agents that we have shown in MASCAN. If agents have different behaviors or different goals, then more complicated models should be used, for example, multi-criteria decision making (MCDM).

In addition, MASCAN also solves the minimum cost flow problem, the total transmission costs is the minimum when flows on the network satisfy all the agents. It implies that the total minimum cost is unique. Therefore

\[
e(i) = (c_{ij} + u_i) \quad \text{for } j \in N_i
\]

\[
\begin{align*}
\text{(a) shortest path: agent 4: } & 2-5-4 \\
& \text{agent 5: } 2-5 \\
& \text{augmenting flow: 2 unit along } 2-5 \\
& \text{cost allocated to agent 5: } 3 \times 2 = 6
\end{align*}
\]

\[
\begin{align*}
\text{(b) shortest path: agent 4: } & 1-3-5-4 \\
& \text{agent 5: } 1-3-5 \\
& \text{augmenting flow: 3 unit along } 1-3-5 \\
& \text{cost allocated to agent 5: } (2+2) \times 3 = 12
\end{align*}
\]

\[
\begin{align*}
\text{(c) shortest path: agent 4: } & 2-1-3-5-4 \\
& \text{agent 5: } 2-1-3-5 \\
& \text{augmenting flow: 1 unit along } 2-1-3-5 \\
& \text{cost allocated to agent 5: } (1+2+2) \times 1 = 5
\end{align*}
\]

\[
\begin{align*}
\text{(d) shortest path: agent 4: } & 2-3-5-4 \\
& \text{augmenting flow: 1 unit along } 2-3-5-4 \\
& \text{cost allocated to agent 4: } (4+2+1) \times 1 = 7
\end{align*}
\]

\[
\begin{align*}
\text{(e) shortest path: agent 4: } & 2-3-4 \\
& \text{augmenting flow: 3 unit along } 2-3-4 \\
& \text{cost allocated to agent 4: } (4+4) \times 3 = 24
\end{align*}
\]

\[
\begin{align*}
\text{(f) total cost allocated: agent 4: } & 31, \\
& \text{agent 5: } 23
\end{align*}
\]

Fig.1. Cost allocation.
agents of MASCAN are playing a noncooperative game. The solution, costs that allocated to the agents as presented by equation 8, is the equilibrium point of the game.

In Fig. 1, agent 5 executes the selfish plan when it is a deficit agent, and executes modest plan when it becomes balanced agent; balanced agents always execute modest plan. The shortest path for agent 4 is 4-5-2 in (a), whereas agent 5 is also a deficit agent who is executing selfish plan; hence the request of agent 4 to augment a flow along agent 5 is rejected by agent 5. Similar situations happened in (b) and (c). Agent 5 received 2 unit flow along 2-5 in (a), 3 unit flow along 1-3-5 in (b), and 1 unit flow along 2-1-3-5 in (c). The cost allocated to agent 5 is 6, 12, and 5 in (a), (b), and (c) respectively. Agent 5 approved the request of agent 4 to augment a flow in (d) because at this time agent 5 has become a balanced agent and executes modest plan. Agent 4 got 1 unit flow along 2-3-4-5 in (a), whereas agent 5 is also a deficit agent who is executing selfish plan; hence the request of agent 4 to augment a flow along agent 5 is rejected by agent 5. Similar situations happened in (b) and (c). Agent 5 received 2 unit flow along 2-5 in (a), 3 unit flow along 1-3-5 in (b), and 1 unit flow along 2-1-3-5 in (c). The cost allocated to agent 5 is 6, 12, and 5 in (a), (b), and (c) respectively.

The total cost allocated to agent 4 is 31 for 4 unit flow, and 23 to agent 5 for 6 unit flow. Agent 4 takes less flow than agent 5 from the network but transmission cost is higher than agent 5. Some cheaper paths, like paths in (a), (b), and (c), must be through agent 5. The paths were occupied by agent 5 first and agent 4 can not take these paths before agent 5 becomes balanced agent.

It is obvious that if no agent who executes egocentric plan received more than one request within any iteration, the cost allocation is uniquely determined for the given network on the condition that all agents are rational. When an agent with egocentric plan must choose a request to approve among several requests, the decision of the agent, i.e., the criteria of choosing a request, will affect the cost allocation of other agents. For example, in Fig. 2(a) agent 5 received two requests of path flow, 5-4-3-1 and 5-4-6. If agent 5 approves the request of 5-4-3-1; agent 1 will take 2 unit flow along path 5-4-3-1. After that, agent 6 can only receive 2 unit flow from agent 2. The cost allocated to agent 6 is 12 and to agent 1 is 8 as showed in (b). If agent 5 approves the request of 5-4-6 in the situation of (a), agent 1 can only take 2 unit flow from agent 2, costs allocated to agent 1 and agent 6 are both 10, showed in (c). Egocentric plan gives supply agent a chance to affect the cost allocation among demand agents, but it seems there is no benefit for supply agent to choose different request. The situation will change if we allow the side-payment between agents. It means that future research should be done under the condition of cooperative game. Although agent with egocentric plan can affect the cost allocation among demand agents, the final arc flow will not be affected. As in Fig. 2 the

![Fig. 2. Cost allocation with egocentric plan of agent 5. (a) agent 5 received two requests of path flow, 5-4-3-1 and 5-4-6. (b) If agent 5 approves the request of 5-4-3-1; agent 6 can only take flow from agent 2, and the cost allocated to agent 6 is 12 and to agent 1 is 8. (c) If agent 5 approves the request of 5-4-6, agent 1 can only take flow from agent 2, costs allocated to agent 1 and 6 are both 10.](image-url)
decision of agent 5 in (a) has no affection on the final arc flows in (d).

In MASCAN, when agent executes the egocentric plan and more than one requests are received, the agent can choose one request to approve according to criteria such as lowest cost path, largest amount of flow, shortest path, etc., among the requests. These criteria will result in much complexity to determine the request to approve. An alternate method is the first come first served, i.e., the agent that executes egocentric plan always approves the first request received. Although it is very easy to implement with low time and message complexity, the speeds of agents execution and communication channels can affect the sequence of requests received. Hence it is out of the control of MASCAN for which request will be approved by first come first served. It implies that cost allocated to a demand agent may in some degree depend on elements that are not intrinsic to cost allocation and should not be involved in the procedure of cost allocation. It is unfair for some demand agents in the circumstances. We have already showed that in MASCAN the cost allocation is the equilibrium point of a noncooperative game. The first come first served policy will result in non-unique equilibrium point for different execution of the system. Therefore future researches are needed to achieve more fair cost allocation on the agent behaviors under the condition of cooperative game. Then effect of egocentric plan that incurs the unfair cost allocation among demand agents can be solved to get a more fair cost allocation for agents.

5. MASCAN Implementation and Complexity

We implemented MASCAN as KQML agents running on Internet. For an Internet agent, it had better that the agents can be run anywhere and be platform-independent. Thus Java applet seems to offer a more advanced, flexible approach for our agents and be the best choice to implement MASCAN. The peer-to-peer communication is a necessary condition for our system.

The infrastructure of MASCAN is showed in Fig. 3. Agent communication is via Internet. An agent name server provides agent registration service and Internet connections for all agents. Fig. 4 shows a java applet that represents the agent 5 of the network in Fig. 1. The agent name and password are used to register the agent into the agent name server. Infobase provides the agent the local information of the network. The format of Infobase is given as

\[
<\text{supply} (\text{neighbor}, \text{cost}, \text{capacity}) \ldots (\text{neighbor}, \text{cost}, \text{capacity})> 
\]

where supply is amount that the agent provides to the network. The negative number of supply indicates the agent is a demand agent. Inside each pair of parentheses an edge that is incident to the agent is described, neighbor is the agent name of the other end of the edge, cost and capacity gives the cost and the capacity of the edge respectively.

MASCAN agent can begin the cost allocation computation only if all the agents in the given network are ready, whereas Infobase only provides the agent the local information and the neighborhood information. An agent by no means can get any information about agents beyond its neighborhood before these agents are started.

To build a rooted spanning root of the network gives the way to detect whether all of the agents are ready. That is, when an agent has built a spanning tree rooted by himself, it is sure for the agent that all the agents of the network are ready to begin the cost allocation computation. Moreover the rooted spanning tree facilitates message broadcast from the agent during the cost allocation computation. A breadth-first spanning tree is the most efficient path for the root agent to broadcast message over MASCAN infrastructure. Thus when an agent is started, the agent first try to build a spanning tree rooted by himself, and then begin the computation described in section 4.

The algorithm to build a spanning tree of a network with m edges and n nodes, given a root node, sends \(O(m)\) messages and executes in \(O(n)\) time. In MASCAN the number of spanning trees to be built
synchronization is based only on communication with the end of negotiation within an iteration. The message complexity of MASCAN is \(O(n^3)\), and time complexity \(O(n^2)\).

When agents negotiate for augmenting flow, agents will send \(O(n)\) messages and execute in \(O(n)\) time. There are several states that agents should coordinate with all other agents to achieve synchronization, such as the end of negotiation within an iteration. The synchronization is based only on communication with \(O(n^3)\) message complexity and \(O(n \log n)\) time complexity. Shortest path computation also needs to be executed once within an iteration. The time complexity of the multi-agent shortest path computation is \(O(nm)\), and message complexity is \(O(nm^2)\). At each iteration at least one augment flow will be approved, so there are at most \(O(nS)\) iteration in MASCAN, where \(S\) is the largest supply in MASCAN. Finally we can have that the message complexity of MASCAN is \(O(mn^3S)\), and the time complexity is \(O(mn^2S)\). Although the primary goal of our system is not the performance of a cost allocation computation, the performance of MASCAN is satisfactory.

### 6. Discussion

We have proposed and implemented a multi-agent system, MASCAN, for cost allocation with satisfactory performance on network flow model. The agents in MASCAN are autonomous agents and can also cooperatively solve the cost allocation problem. MASCAN analyze the insight behaviors and intuitions of each agent during the cost allocation. When the agent represents a generation or a load in electric transmission network, this system can be used to support the cost allocation problem under the deregulation of electric industry.

MASCAN is a fully distributed system. There is no centralized controller in MASCAN. The agent in MASCAN, representing a node of network, does not receive any centralized control and centralized information source, and makes decisions only on behalf of himself to contend lowest cost with other agents. While each agent in MASCAN pursues to satisfy his demand from the network with cheapest possible cost, MASCAN solve the minimum cost flow problem of the given network. It showed that pursuit of local utility maximization of each agent will finally achieve global utility maximization for the community of agents in MASCAN. There is no conflict between individual utility maximization and global utility maximization in MASCAN.

The cost allocated to agents in MASCAN is also the equilibrium point of a noncooperative game subject to the constraints of the given network. So agents in MASCAN play a noncooperative game to allocate the total cost of the given network. It implies that the inheritance of MASCAN is consistent with the underlying principle of economic theory.

### 7. Reference


