Identifying Decision Makers For Large-Scale Group Decisions

Christian Stary

Institute for Applied Computer Science
Department of Distributed Databases and Expert Systems
Technical University of Vienna
Paniglgasse 16 - 181/2
1040 Vienna
Austria

Abstract

Important and complex decisions need large groups of decision makers. Linking decision makers along decision links constitutes an essential element for the operation of such groups. Unfortunately, the overhead of message passing becomes prohibitive in a flat organization. Models of group decisions are discussed in context with addressing and message passing algorithms. A hierarchical network representation scheme for decision makers is proposed to solve the linking problem. The two key features of the modeling approach are the hierarchical addressing mechanism and the adaptive message passing algorithm.

1. Modeling the Problem Space

Within the discussion of decision processes involving (large) groups there are several terms and topics of interest: decision maker, group, categories of participants in decision making (e.g., roles of individuals), addressing decision makers via decision links, and message passing. Based on a certain interpretation of the mentioned terms and topics, a representation scheme can be proposed for static entities concerning the discourse of decision making.

1.1 Elements

For our purpose, some terms (describing group decision support systems) have to be clarified, before decision making can actually be modelled. These definitions allow us to define static as well as dynamic concepts later on.

Decision Maker. We consider an individual to be a decision maker, if it takes part in solving a certain problem, and finally is responsible for the solution of this problem. A decision maker can be described by an identifier (e.g., the name), one or more roles (e.g., representative of special group interests, group moderator), and one or more fields of expertise (including the level of competency). The level of authority can also be given implicitly by the hierarchical position within a hierarchical group organization - where a decision maker is always a member of one of the involved subgroups.

Group. For our purpose a group is simply defined as a set of individuals performing activities in a problem area under certain conditions. These activities concern planning, implementation of plans, communication and referential actions (directing, delegating), etc. Conditional factors include the complexity of subproblems, group autonomy, requirements for decision makers, etc. We consider a group to be a large one if it contains more than 100 members (decision makers). A flat organization of group members include not more than four different (organizational) levels.

Group Decision Making. Group decision making is a special kind of problem solving because a combination of ideas is evaluated to serve as correct (and not only possible) solution to a problem. "A decision making group can be defined as two or more people who are *jointly* responsible for detecting a problem, elaborating on the nature of the problem, generating possible solutions, evaluating potential solutions, or formulating strategies for implementing solutions." (DeSanctis et al., 1987). Consequently, large-scale group decision making is considered to be a process which involves more than 100 decision makers organized as a group.

Decision Link. The chosen algorithm for decision making determines the links between decision makers. A decision link is established according to the requirements for decision processes. It depends on the current problem as well as on the level of competency of the involved decision makers (according to the problem).

Message. The transmission of information or knowledge between two or more decision makers is done by messages. In the case of voting algorithms it may be the number of open (sub)decisions or a list of missing votes (including the pro's and con's) of decision makers. In the case of a discussion information on a certain topic of interest and a list with further group members to contact with can be transmitted. A message is sent from node to node. Its content can be changed or not, dependent on its purpose (Malone et al., 1987).

1.2 Definition of a Representation Schema

To define an addressing scheme for decision makers and a message passing algorithm according to predefined decision links, in a first step the static entities of group decision algorithms have to be mapped into a representation scheme. Static entities are: decision makers, subgroups, and decision links. We decided to use a network for the representation of these entities because it allows strategic decomposition as well as different kinds of links between decision makers (Bonczek et al., 1979). Furthermore, roles of decision makers can be modelled without neglecting their organizational position. The mapping from static entities to modelled elements is defined in Figure 1.

Real World Entity	-->	Model Entity
decision maker	-->	network node
decision link	-->	network link
subgroup	-->	class
organizational hierarchy	-->	hierarchy of levels

Figure 1

Representation Mapping

The network nodes (i.e. members of a group decision process) are organized into m hierarchical levels of classes. A *class* is simply a group of nodes, where each node belongs to exactly one class at each level of the hierarchy. A group of nodes forms a class at level 1, called 1-class. Similarly, a group of (k-1)-classes forms a k-class. The classes defined in the network form a class hierarchy.

There are two types of nodes - simple and class nodes. They represent the only linkable entities in the network. A *simple node* is one that every node in its same 1-class knows to reach. A *class node* at level m (i.e. an m-class node) is one that every other node in the same (m+1)-class knows how to reach. Of course, a simple node can be considered a class node at level 0.

2. Identifying Participants for Decision Making

For a particular decision algorithm - that assumes a flat group hierarchy - each node must know how to reach relevant group members in the network. This means that the overhead of maintaining correct information about identifiers becomes excessive when hundreds of participants are involved. Commonly used algorithms (e.g., Turoff et al., 1982) are too inefficient to converge after participant dismissions and subgroup formations (network partitions), because incorrect entries in the nodes' addressing tables may form linking table loops. Algorithms in use also assume, that every node knows the complete topology of the network. This means that an update message must be sent to all the nodes in the network every time the requirements for decision support change. In a network with a great amount of nodes, it is clear that the communicational overhead to maintain efficient information flows becomes excessive.

2.1 Hierarchical Identification

To alleviate the identification overhead problem of flat linking algorithms, decision making can be simplified by determing representatives, and the frequency of updates can be reduced. However, this inhibits the network's ability to adapt to changes, which may actually increase the communicational overhead because nodes may pass messages towards unavailable group members. A more viable solution to this problem is hierarchical identification. The central concept in hierarchical identification (or linking) is to employ a hierarchical addressing scheme in the network, to reduce the overhead associated with its linking.

The addressing scheme is based on making the name of a node (called class node) the address of a group of other nodes (called simple nodes). The name of a network node within a flat hierarchy constitutes its address. A name-to-link mapping can be specified directly in the nodes' linking tables according to a certain decision process. The fact, that reducing the amount of linking information at each node creates an addressing problem, is often overlooked. A node has to know where another node is located before it can send a message to it. For example, if a node i wants to send a message to node j in another class, node i must first find out that the address of node j is class xy. Only then can it use its linking table to send the message. The need for this intermediate name-to-address mapping in turn presents some important limitations.

First, linking-table loops may occur which prohibit effective decision making. Second, in the general case, name-to-address mapping requires maintaining an address data base listing the name of every network node and its address (i.e. its affiliated class), and a class data base listing the name of each class and a boundary (entry) node that can be used to reach it. To some extent, introducing address or class data base defeats the hierarchical-linking goal of reducing the amount of information each node must have about other nodes, since this would require updating additional addressing information when nodes change their class affiliations or dismiss the network (i.e. a participant leaves the group).

Although a centralized data base could be used to reduce the nodes' storage overhead, it would be unacceptable, because the server maintaining the data base would be subject to too many queries and updates. The result would be a bottle neck for message passing. In fact, for time-dependent decision processes, where linking has to be done very quickly, the address and class data bases should be available for every network node.

To avoid the need for class and address data bases, hierarchical addresses can specify the nodes' affiliation to classes. For instance, node *i* in cluster *xy* could be named '*xyi*', thus enabling every other node to know that the address of name *xyi* is *xy*. In turn, classes can be defined according to the connectivity among nodes (that is, the structure of the network). Its main advantage is that there is no need for tables containing the mappings of node names into class names as long as the affiliation of nodes is static. Unfortunately, this scheme fails in the event of organizational changes, because of the need to dynamically maintain the affiliation of nodes in classes, classes to superclasses, and so on in the presence of organizational changes. Even if all network nodes are static, a class can be partitioned due to link failures or node dismissions. And once a class is partitioned, a subset of the network may be unable to communicate with the remaining components of a class unless new names are assigned to the class component and its nodes.

2.2 Hierarchical Addressing

Nodes are addressed with respect to class nodes and not the classes themselves. Each simple node is affiliated with one 1-class node. Similarly, each k-class node is affiliated with one (k+1)-class node. Each k-class node is assumed to always belong to the same k- and (k+1)-classes. Furthermore, each simple node belongs to only one 1-class. From the viewpoint of node i, the address of node j in its same k-class is the (k-1)-class node to which node j is affiliated. If node j lies in the same 1-class as node i, node j's address is its own name.

In a network of m hierarchical levels, the name of a node in the network is formed by concatenating m k-identifiers, where a k-identifier is formed by a k-class label and a k-class node label. A 0-identifier corresponds to a simple-node label. Notice that all k-class nodes in the same k-class share the same k-class level. Furthermore, the 1-class-node label of a class node is the same as its 0-identifier. The label of a k-class is unique within its (k+1)-class in the class hierarchy and a k-class-node label is unique within the k-class of the class node. As a result, a k-identifier is unique within the naming scope of a (k+1)-identifier. The concatenation of the m-1 i-class labels (where $1<=i<=m$, and the m-class refers to the whole network) and the 1-class-node label of a node uniquely specify the 1-class-class to which the node is affiliated. The m-1 class-node label constitute a linking hint from the source to the forwarding (passing) nodes.

From the above, we observe that a k-class consists of the nodes that are affiliated with any one of the k-class nodes that share the same k-class label. The key benefit of this approach is that all the algorithms required for the management of the linking hierarchy can be centered at the class nodes. Because of space limitations, the focus of this paper is on the hierarchical-linking problem, that is, on maintaining consistent information in a network given that the addresses of network nodes (simple or class) are known. In the rest of this paper, we assume the existence of algorithms that affilitate a node with a class node and create classes by the class nodes.

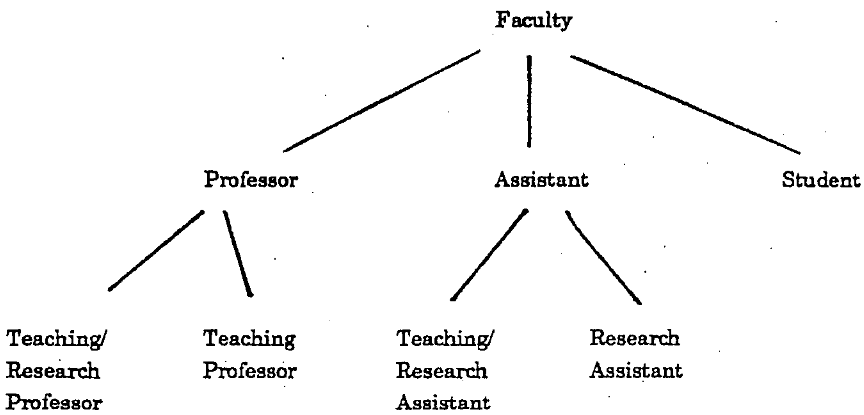

Figure 2

Structural Decomposition of a Faculty

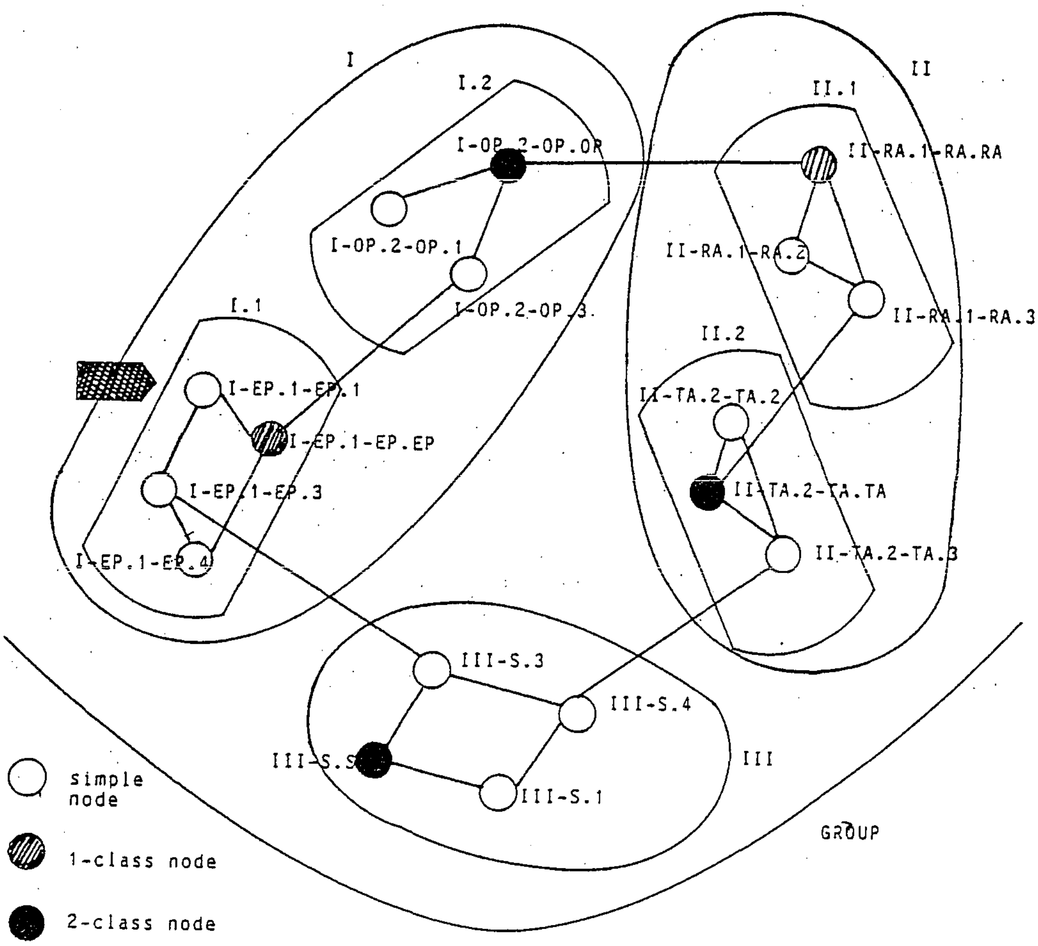

Figure 3

Exemplifying Hierarchical Identification

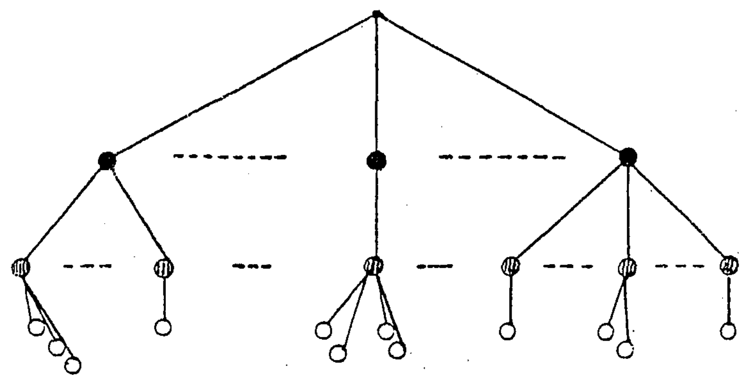

Figure 4

Node Hierarchy

Figure 3 shows a network organized into classes according to the hierarchical addressing scheme. The example is based on the structure illustrated in Figure 2 which decompose a large group into subgroups within the faculty of an university. In Figure 3, the k-identifiers of a node's name are separated by dots, and the k-class label is separated from the corresponding k-class-node label by a hyphen. So, for example, the name I-EP.1-EP.1. means that the node is affiliated with 2-class node EP of 2-class I and with 1-class node EP of 1-class I.1. Figure 4 shows the structure of a graph for the representation of the affiliation of nodes to class nodes. Notice that each node knows both all the class nodes at the same hierarchical level and descendants of the same class node of each of the nodes in the link to the top-level roots of the graph, and nothing else.

2.3 Addressing Information

Each simple and class node maintains a link table and knowledge according to the decision in process. The link table contains information of all adjacent links. Addressing information propagates outside 1-classes only if it refers to class nodes. A k-class node notifies changes that refer to l-class nodes (where $0<l<k$) to its neighbours, but nodes at the boundary between two classes stop the propagation of addressing information that refers to simple nodes within each other's 1-classes. Within a 1-class, then, every node knows how to reach every other node through links that only include nodes within the same 1-class. Within an i-class, every node knows which are the i-class nodes in the class and how to reach them through links that include only nodes inside the same i-class.

The main purpose of classes in the proposed representation scheme for participant identification is to enable decision processes via representatives for real-time decisions. These representatives act on a set of decision makers, which are definitively involved in the group decision making process and available. Furthermore, the frequency with which specific information about decision makers needs to be updated can be reduced.

3. Adaptive Message Passing

We have already stated that intraclass information does not propagate outside the classes. Also, because a node is affiliated with class nodes and not the classes per se, messages can be delivered to their destination even after a class has been partitioned, and there is no need to redefine the node's address immediatly after a class' partition.

The passing of messages between two nodes that lie within the same class occurs the same way as in a flat hierarchy. Each message contains the name of the destination node (i.e. an identifier of a decision maker) and the name of class node that belongs to the same class as the destination node. Thus, message passing is carried out as follows:

(1) The source node determines whether the destination node belongs to its class or not.

(2) If the destination node belongs to the source node's class, the source node looks up its linking table to determine if the destination is reachable. If it is, the source node passes the message accordingly. If the destination node is unreachable, the source node treats the destination as if it is in a remote class.

(3) If the destination node belongs to a remote class, the source node looks up its linking table for address of the node in the destination node-class. If a node is unreachable, the message is passed to an alternate nodes according to the decision process. If no alternate node is needed, the decision process has come to an end.

(4) Each node in the decision process from the initialisation to the result does the same.

From the above, we observe that messages need not all go through class nodes to get a decision result. Furthermore, if an intended node is unreachable, a representative from the same class can ensure a relevant contribution to the decision process. So, the decision process does not have to stop after a class has been changed. This feature is of great importance for time-dependent decision processes.

4. Conclusion

Although many group decision support systems have been tested in large-scale environments (e.g., negotiations between large interest groups), which require complex techniques for communication (e.g., Kersten, 1985), the problem of identifying individual decision makers with a minimal overhead of addressing information has been remained untouched. The problems of inefficiency (when addressing different types of decision makers) and the need for adaptive mechanisms has become evident when information sharing had to be supported (Huber, 1984).

In this paper, a new approach to identify decision makers involved in large-scale group decision processes has been presented. It is based on a representation scheme which has been derived from semantic networks. It contains different kinds of nodes and different organizational group levels for large-scale group decisions. The major reasons for the effectiveness of the addressing scheme can be summarized as follows:

(1) Its use in real-time decision making is suggested because it leads to *reliable results*, which are assumed in time-critical applications (Adelman, 1984).

(2) Hierarchical group levels support the concept of *representatives* which is commonly used for large-scale group decision making.

(3) Although hierarchical levels are introduced (to enable representatives), *direct contact to all members* of the group according to the decision model is maintained, which means democracy is enabled by keeping a great amount of decision makers.

(4) More efficient group decisions *enhance the degree of solution-legitimacy* because the decision can in principle be based on a large amount of decision makers and relevant information holder.

(5) *Different organizational models* of decision making (e.g., Huber, 1981) are directly supported.

Although the addressing scheme does not support the way of how decisions are found, it is a first step for adequate modeling large-scale group and real-time decision making. Time-critical decision making is still a topic of further research. According to the environment of problem situations (constant changes, unpredictable behaviour) further research activities must be concentrated on knowledge representation techniques for multi-agent activities (e.g., Galliers, 1988). Furthermore, it has to be taken into account, that an individual can play more than one role simultaneously (according to the decision algorithm and the requirements for other decision makers).

References

Adelman, L.: Real-Time Computer Support for Decision Analysis in a Group Setting: Another Class of Decision Support Systems, in: Interfaces, Vol. 14, No. 2, pp. 75-83, March-April 1984.

Bonczek, R.H.; Holsapple, C.W.; Whinston, A.B.: Computer-Based Support of Organizational Decision Making, in: Decision Science, Vol. 10, pp. 268-291, 1979.

DeSanctis, G.; Gallupe, R.B.: A Foundation for the Study of Group Decision Support Systems, in: Management Science, Vol. 33, No.5, pp. 589-609, May 1987.

Galliers, J.R.: A Strategic Framework for Multi-Agent Cooperative Dialogue, in: Proceedings ECAI '88, ed.: Kodratoff, Y., pp. 415-420, GI, Munich, August 1-5, 1988.

Huber, G. (1981): The Nature of Organizational Decision Making and the Design of Decision Support Systems, in: Management Information Systems Quarterly, Vol. 5, No. 2, pp. 1-10, June 1981.

Huber, G. (1984): Issues in the Design of Group Decision Support Systems, in: Management Information Systems Quarterly, Vol. 8, No. 3, pp. 195-204, September 1984.

Kersten, G.B.: NEGO-Group Decision Support System, in: Information & Management, Vol. 8, pp. 237-246, 1985.

Malone, T.W.; Grant, K.R.; Turbak, F.A.; Brobst, St.A.; Cohen, M.D.: Intelligent Information-Sharing Systems, in: Communications of the ACM, Vol. 30, No.5, pp. 390-402, May 1987.

Turoff, M.; Hiltz, St.R.: Computer Support for Group Versus Individual Decisions, in: IEEE Transactions on Communications, Vol. COM-30, No. 1, pp. 82-91, January 1982.