Locally Perceiving Hard Global Constraints in Multi-Agent Scheduling

Joaquim Reis

Dpt. Ciências e Tecnologias de Informação, ISCTE
Avenida das Forças Armadas, 1649-026 Lisboa, Portugal
e-mail: Joaquim.Reis@iscte.pt

Nuno Mamede

Dpt. de Engenharia Informática, IST
Avenida Rovisco Pais, 1049-001 Lisboa, Portugal
e-mail: Nuno.Mamede@acm.org

Henrique O'Neill

Dpt. Ciências e Tecnologias de Informação, ISCTE
Avenida das Forças Armadas, 1649-026 Lisboa, Portugal
e-mail: Henrique.Oneill@iscte.pt

We propose how to model enterprise facilities (like factories, warehouses, etc.) in a multi-product production/distribution network, capacity management at those facilities, and scheduling agents which act as enterprise managers, taking decisions that affect the available capacity. A coordination mechanism through which scheduling agents can locally perceive hard global temporal constraints is also proposed.

Keywords: Supply Chain Management, Scheduling, Multi-Agent Systems

1. Introduction

Coordinated planning of logistics activities has been, in the areas of Management Science and Operations Research (OR), the subject of extensive investigation since the sixties (see [Graves 1993], [Thomas 1996] or [Parunak 1998], for instance). In spite of that, only recently production/distribution scheduling has had more dedicated attention (see [Fox 1993], [Tate 1995], [Rabelo 1996], [Hildum 1997], [Arnold 1997], [Kjenstad 1998], [Klen 1998], [Rabelo 1998], for instance). Production/distribution scheduling is the main theme of this article.

For a general introduction to the classical scheduling problems, including classical/OR approaches see [Blazewicz 1994] or [Brucker 1998]. For Artificial Intelligence (AI) based approaches or modern mixed AI/OR approaches see [Zweben 1994] or [Morton 1993].

This article describes two aspects of an approach to a scheduling problem, in a specific context of supply chain management, usually termed the Extended Enterprise (EE) [O'Neill 1996]. The EE is usually assumed to be a kind of Virtual Organization, or Virtual Enterprise, whether centered around a main enterprise, or not, and where the set of participant agents is relatively stable (for concepts, terminology and classification see [Camarinha-Matos 1999]).

The main features of the scheduling problem in the referred context are:

a) It is a highly dynamic scheduling problem - The scheduling problem can change and the development of a scheduling solution is an on-going process, during which unforeseen events must be accommodated (this is probably more important than the classical search of the optimal scheduling solution);

b) It is a Multi-Agent Cooperative Scheduling (MACS) problem - Each participant agent has its own (local, and incomplete) perspective of the scheduling problem, and can try to satisfy its scheduling preferences, but must be cooperative enough so that it won't invalidate a feasible scheduling solution, if there is one.

The following sections expose two topics which cover the following two aspects of an approach to the scheduling problem referred:

1. A model for MACS - We adopt the AI Multi-Agent Systems (MAS) paradigm (see, for instance, [ICMAS 1996] or [O'Hare 1996]) and propose a group of autonomous agents achieving schedule coordination through communication;

2. A mechanism that allows a group of cooperative scheduling agents to perceive hard global temporal constraints of the scheduling problem by locally exchanging and maintaining a specific, and limited, piece of information. This mechanism allows the agents to avoid exploring scheduling/rescheduling solutions which could appear locally feasible but are not globally feasible. Although it was developed for the context of our EE MACS model, this mechanism can be adapted to any MACS system.

2. A Model of Multi-Agent Cooperative Scheduling

This section summarizes the essential parts of our model. A suitable example will be used, including for the specific aspects to be described. More detail about the model can be found in [Reis 1999a] or [Reis 1999b], and [Reis 1998].

In [Reis 1999a] a two-level model for the EE scheduling environment was proposed including: the *physical level*, respecting to the production/distribution network, and the *virtual*, or *agent*, *level*, respecting to agents which own and manage the physical resources of the physical level. An *EE network*,

$\mathbb{R}$, can be defined by a pair $<\mathbb{RF}, \mathbb{RA}>$, where $\mathbb{RF}$ and $\mathbb{RA}$ represent the physical and the virtual network, respectively.

2.1. The Physical Level

The *physical network* is a multi-product production/distribution network, an acyclical network composed of physical renewable resources, referred to by *physical nodes*. The *physical arcs* of the network link the nodes and point the possible flows of products between pairs of supplier and client nodes. *Capacity nodes* are capable of executing logistic *tasks* (production, transportation, storing), making available a certain set of *output products* for consumption by other nodes, and consuming a certain set of *input products* (or *materials*) made available by other nodes in the network. In this model, both the network and the nodes are multi-product, in the sense that they can deliver more than one product. There are three types of capacity nodes, according to the type of tasks they can execute: the *store node* (S), the *producer* (P) and the *transporter* (T) node. Additionally, *retail nodes*, make available products to the outside, and *raw-material nodes* consume products from the outside of the network. The set of products of the retail nodes defines the set of network *end products* (with *demand* from the outside). Each capacity node has a limited *capacity* for task execution, and maintains an internal capacity state by recording a *schedule* of the available capacity, and the capacity being used by its tasks, along a certain temporal scheduling horizon. The internal capacity state is actually a *scheduling state*, in the sense that records values of capacity along the scheduling horizon, resulting from some scheduling decision.

A physical network, $\mathbb{RF}$, is defined by a pair:

$<V, E>$ ($V=\{\mathbb{V}_1, \ldots, \mathbb{V}_n\}$, $E=\{\ldots, \mathbb{E}_{i,j}, \ldots\}$, with i and j denoting nodes $\mathbb{V}_i \neq \mathbb{V}_j$)

where V is the set of physical nodes and E is the set of physical arcs. Each arc $\mathbb{E}_{i,j}$, defines a supplier-client relationship, and is defined by a triple $<\mathbb{V}_i, \mathbb{V}_j, P_{i,j}>$, where $\mathbb{V}_i$ and $\mathbb{V}_j$ are the supplier and client node, respectively, and $P_{i,j}$ is the set of products supplied by $\mathbb{V}_i$ and consumed by $\mathbb{V}_j$. To keep this explanation simple, the internal details of a physical node, actually complex, won't be described here (this description can be found in [Reis 1998]).

It is assumed that, once a physical network is defined, it won't change. Some additional, and simplifying, assumptions adopted are: a) for each capacity node there are is a single supplier for each input product, and b) the product structure for any end product at any retail node always forms an in-tree structure.[1]

As an example see the physical network represented in Figure 1. This physical network has thirteen capacity nodes, with identifiers v_1 to v_{13}, and two end products, p_1 (delivered at retail nodes v_{14} and v_{15}) and p_2 (delivered at retail nodes v_{15} and v_{16}). Physical arcs are labeled with the identifiers of products transferable between the corresponding pair of physical nodes.[2]

2.2. The Virtual Level

The *virtual network* is a network of *agents*, or *virtual nodes*. These are linked by bi-directional *communication links*, the *virtual arcs*, that connect pairs of agents for the communication necessary to

[1] In the near future, the model is to be extended by relaxing assumption a). This will imply, at the virtual level, additional decisions about alternative suppliers (and, possibly, inter-agent negotiation based on dates and prices for product orders). Assumption b) guarantees having no more than one task per each node involved in any network job (see section 2.4), and allows a simple representation of local scheduling problems (see section 2.5) with just one task (and just one request from a client) per global scheduling problem.

[2] Note that nodes identified by v_{14} to v_{16} and v_{17} to v_{20}, here called retail nodes and raw-material nodes, respectively, are fictitious nodes used just for modeling purposes. These ficticious nodes just represent "ports" to the outside in the model and wouldn't correspond to any site in a real production/distribution network. The "real", and so-called retail nodes and raw-material nodes in the literature (see [Williams 1981], for instance), would correspond, in Figure 1, to nodes labelled v_1, v_2 and v_3 (for "real" retail nodes) and v_8, v_{11}, v_{12} and v_{13} (for "real" raw-material nodes).

coordination. Each physical node is owned and managed by one agent and each agent owns and manages one node, and the client-supplier relationships among nodes are extended to the agents. Depending on the node it owns, each agent can be either a *capacity agent*, a *retail agent* or a *raw-material agent*. Each capacity agent can receive requests and rescheduling requests from their clients, and send requests and rescheduling requests to its suppliers. Requests are orders of a certain amount of a product for a certain date, and are limited to the set of output products of the agent node, for the case of the requests received, and to the set of input products of the agent node, for the case of the requests sent. A rescheduling request refers to a previously accepted request and states the need to agree on an alternative date, motivated by temporal or capacity constraints.

For the satisfaction of a received request, each capacity agent must create and schedule a task in its node. Individual agent scheduling decisions are made according to *scheduling preferences*, which can be, for instance, scheduling a task the earliest as possible or the latest as possible. An agent can always revise a scheduling decision, if it collides with global (network) temporal limits, when facing a rescheduling request from a supplier or a client, and when available capacity allows.

In the EE framework there is a *supervision unit*, which corresponds to a team of all members involved and has an integration role, including the proposal of plans in a medium/long-term perspective, in face of demand forecasts for the end products of the network (see [O'Neill 1996]). Specifically for the short-term activity of scheduling, the supervision unit is maintained, in the model, in the form of a special, and unique, virtual node, the *supervision agent*. This agent has the only role of introducing new scheduling problems, has no physical node associated and does not intervene in the activity of the remaining agents (apart from introducing new work in the system). This setting cannot be considered strictly hierarchical, because agents do not depend on the supervision agent for taking their individual scheduling decisions. In the near future we plan to extend the model, to accommodate network planning activity and the influence of medium/long-term network planning on the individual agents short-term scheduling activity.

A virtual network, $\mathbb{RA}$, for a given physical network, $\mathbb{RF}$, is defined by a pair:

$<\mathbb{G}_0, \mathcal{GV}>$ $\quad (\mathcal{GV}=\{\ldots,<\mathbb{G}_i,\mathbb{V}_i>,\ldots\})$

where $\mathbb{G}_0$ is the supervision agent, and $\mathcal{GV}$ is a set of agent-node pairs, $<\mathbb{G}_i,\mathbb{V}_i>$, defining a one-to-one association between each agent $\mathbb{G}_i$, and a node, $\mathbb{V}_i$, of $\mathbb{RF}$.

For the physical network illustrated in Figure 1, there is the complementary agent network represented in Figure 2, with the supervision agent identified by g_0, and with agent g_1 managing node v_1, agent g_2 managing node v_2, etc..

The following sub-sections focus in structures used in the individual agent activity (both intra-agent and inter-agent oriented) for scheduling, which link the two levels of the model.

2.3. Requests, Events and Tasks

Coordination is to be achieved through communication between pairs of client-supplier agents. Each agent will only interact with suppliers and clients, by sending and receiving certain types of *messages* in the context of agent *conversations*. For a product *local request* (*i.e.*, an inter-agent order, from a client agent to a supplier agent), a message typically contains product, quantity, time and node information. Product *global requests* (*i.e.*, orders from or to the outside of the EE network) contain the same kind of information, but are processed only by the supervision and retail and raw-material agents.

Events, denoted in general by e, are elementary records of information containing product, quantity, time and place information. For the intended multi-agent scheduling purposes, everything in the multi-product production/distribution environment can be described through events. For instance, a product request, a task start or a task end, have associated product, quantity, time and place information, and they can be represented through the event concept. Formally, an event, e, can be defined as a triple of the form:

$\langle\{\ldots,\langle p_i,q_i\rangle,\ldots\},t,id\rangle$ with $p_i \in \mathbb{P}$, $id \in \mathit{id}$

where the first element of the triple is a set of finite product-quantity pairs — p_i denotes a product ($\mathbb{P}$ is the set of all products in the network) and q_i the quantity of product p_i (any number) —, t is a temporal instant (an integer) and id is a (physical or virtual) node identifier (id is the corresponding set of identifiers).

Events used to represent product requests have just one product-quantity pair, the quantity is a positive number, the time value is the due-date proposed and the node information identifies the supplier agent. For instance, in the EE network illustrated in Figure 1 and Figure 2, a request message from g_5 to g_7, ordering q_x units of product p_2 for due-date t_x, would contain an event like $\langle\{\langle p_2,q_x\rangle\},t_x,g_7\rangle$. The values in the event convey the essential pieces of information of *what* (product p_2), *how much* (quantity q_x), *when* (time t_x), and *where* in the network (agent g_7). Request events are denoted by $\mathbb{d}$. A local request from an agent g_i to an agent g_k, that originated from the i_rth global request (from the outside of the network) to retail agent g_r is denoted by $\mathbb{d}^{i,k}_{i_r,r}$.

Tasks are also described through the event concept. A task is defined as a pair of events $\langle e_s,e_e\rangle$, where e_s is the task *start event* and e_e is the task *end event*. Tasks are denoted by $\mathbb{O}$. A task to satisfy a local request $\mathbb{d}^{i,k}_{i_r,r}$ of agent g_k is denoted by $\mathbb{O}^{k}_{i_r,r}$ and gives rise to a number of $F^{k}_{i_r,r}$ local requests to suppliers.[3] See illustration in Figure 3, where, over a time line, $\mathbb{O}^{k}_{i_r,r}$ represents a typical (producer or transporter) agent task of an agent g_k, $\mathbb{d}^{i,k}_{i_r,r}$ the associated request from downstream (received from a client agent, g_i) and $\mathbb{d}^{k,j}_{i_r,r}$ one of the associated requests to upstream (sent to supplier agents g_j, with $j=1,\ldots,F^{k}_{i_r,r}$).

2.4. Network Jobs

A *network job*, denoted by $\mathbb{RT}$, is a graph made with the set of all agent tasks that originated from the same product global request. Figure 4 represents the network job $\mathbb{RT}_{1,14}$, which would develop from a global request, $\mathbb{d}^{0,14}_{1,14}$, of product p_1 at retail agent g_{14} of the EE network example. This development would occur after the propagation of inter-agent requests upstream the network represented by solid arcs in Figure 5 (additional information contained in this figure will be explained later on).

The arrows in a network job graph mirror the supplier-client relationships among the agent nodes, and indicate temporal precedences between pairs of agent tasks. It is important to note that, as tasks are private to the agents (and so, each agent doesn't know about other agents tasks), a network job is an abstraction (and so, there is no such thing as a network job, from an individual agent perspective). But, as it will be shown in next sub-section, for each agent task, each agent internally retains, in some form, the temporal constraints corresponding to the task.

Temporal slacks inserted in a schedule can be very important for schedule flexibility (see [Hildum 1994], for instance). Additionally represented in Figure 3 are the temporal slacks inserted in a network schedule from the perspective of a capacity agent g_k. *Internal slacks* are inserted locally and by the initiative of the agent g_k; *external slacks* result from similar behavior in all agents of the network. The *internal*

[3] Although tasks are private to agents, we must refer to them in the context of the EE network. Also, it will be $F^{k}_{i_r,r} \geq 1$, for any capacity agent, g_k, in general. For the particular cases of the store and the transporter agents (*i.e.*, capacity agents managing S or T type nodes) it is always $F^{k}_{i_r,r} = 1$.

downstream slack and the *internal upstream slacks* are denoted by symbols `fij` and `fim`, respectively, and the *external downstream slack* and *external upstream slacks* are denoted by symbols `FEJ` and `FEM`, respectively.

The relative positioning of the interval of the task $\mathbb{O}^{k}_{i_r,r}$ and the intervals $\mathbb{h}^{k,j}_{i_r,r}$ depends on the scheduling preferences exerted by agent g_k, and is described by internal slacks `fij` and `fim` (which should both be non-negative), defined as follows:

$\mathtt{fij}^{k}_{i_r,r} = \mathtt{TIME}(\mathbb{d}^{i,k}_{i_r,r}) - \mathtt{TE}(\mathbb{O}^{k}_{i_r,r})$ and

$\mathtt{fim}^{k,j}_{i_r,r} = \mathtt{TS}(\mathbb{O}^{k}_{i_r,r}) - \mathtt{TIME}(\mathbb{d}^{k,j}_{i_r,r})$

where `TS` and `TE` return the start and end time of a given task, and `TIME` returns the time value of a given request.

In the context of a process $\mathbb{RT}_{i_r,r}$, an agent g_k can reschedule its task, $\mathbb{O}^{k}_{i_r,r}$, without sending rescheduling requests to the client, or to the supplier agents, as long as the task interval falls inside the most restrictive $\mathbb{h}^{k,j}_{i_r,r}$ interval. The intervals $\mathbb{H}^{k,j}_{i_r,r}$ are similar to the intervals $\mathbb{h}^{k,j}_{i_r,r}$, but include additional (non-negative) duration corresponding to the sum of a pair of values of external slacks. The most restrictive $\mathbb{H}^{k,j}_{i_r,r}$ interval has, however, a more important meaning: its extreme time points are the hard limits for scheduling task $\mathbb{O}^{k}_{i_r,r}$ without violation of two global temporal constraints: the network due-date, $\mathtt{DD}_{i_r,r}$, and the network release date, $\mathtt{RD}_{i_r,r}$, *i.e.*, they mean the *latest finish time* and the *earliest start time* for the task.

2.5. Scheduling Problems

Whenever a new product request appears, an agent will create a new *scheduling problem*, which will then be joined to the agent list of current scheduling problems (see next sub-section). In particular for a capacity agent, the description of an actual scheduling problem accommodates the respective solution and is to be dynamically updated by the agent whenever rescheduling occurs. Note that the *scheduling problem* is an elementary scheduling problem. The "real" scheduling problem of the agent is composed of these elementary problems, and is described by the list of current scheduling problems. Whenever there is a new order, the scheduling problem of the agent changes to accommodate for the new elementary problem (a similar approach is frequently applied to dynamic scheduling problems, see [Blazewicz 1994], chapter 8.). As an EE network is a MAS there can be multiple perspectives of the same scheduling problem.

For the local perspective of a capacity agent g_k, a scheduling problem, $\mathbb{PE}^{k}_{i_r,r}$, is described by the tuple:

$<\mathtt{DD}^{k}_{i_r,r}, \mathbb{RD}^{k,}_{i_r,r}, \mathbb{d}^{i,k}_{i_r,r}, \mathbb{d}^{k,}_{i_r,r}, \mathbb{O}^{k}_{i_r,r}>$

where $\mathbb{d}^{i,k}_{i_r,r}$ is the request from a client, $\mathbb{O}^{k}_{i_r,r}$ is the associated agent task scheduled, $\mathbb{d}^{k,}_{i_r,r}$ is the set of the associated ($\mathbb{d}^{k,j}_{i_r,r}$) requests to suppliers, $\mathtt{DD}^{k}_{i_r,r}$ is the hard due-date and $\mathbb{RD}^{k,}_{i_r,r}$ is the set of hard release dates (both for task $\mathbb{O}^{k}_{i_r,r}$ of agent g_k). Local due-date and release dates, $\mathtt{dd}^{k}_{i_r,r}$ and $\mathtt{rd}^{k,j}_{i_r,r}$ (see Figure 3), are represented, in $\mathbb{PE}^{k}_{i_r,r}$, through the time values of the request $\mathbb{d}^{i,k}_{i_r,r}$ and of the requests in $\mathbb{d}^{k,}_{i_r,r}$, respectively. These dates implicitly represent the intervals $\mathbb{h}^{k,j}_{i_r,r}$. The intervals $\mathbb{H}^{k,j}_{i_r,r}$ are also

implicitly represented, through $DD^{k}_{i_r,r}$ and the $RD^{k,j}_{i_r,r}$ ($RD^{k,j}_{i_r,r} \in \mathcal{RD}^{k,}_{i_r,r}$). These latter dates play an important role in the second topic of this paper and, in a later section, it will be described how an individual agent in the network maintains up-to-date information about them.

For the global perspective of the supervision agent, g_0, the description of a scheduling problem includes the initial global data of the problem, which is a global request, $\mathcal{d}^{0,r}_{i_r,r}$, and the global due-date and release date, $RD_{i_r,r}$ and $DD_{i_r,r}$.

2.6. Agent Architecture

The architecture sketched in Figure 6 groups the important components of a capacity agent. The component labeled *interface* contains input and output mailboxes where messages received from, and messages to send to, other agents (clients and suppliers) are temporarily stored. The identifiers of the client and the supplier agents, as well as the identifiers of the products consumable (by clients) and the products deliverable (by suppliers) in the EE network are also stored (permanently) in this component.

The component labeled *internal state* maintains: a) an *interaction* (or *communicational*) *state*, by maintaining the inter-agent conversations presently active with other (client or supplier) agents, b) the list of current *scheduling problems*, and c) a *capacity state*, which includes, for the node managed by the agent, the capacity available and the capacity used by the tasks scheduled, for the scheduling temporal horizon. The high level protocol for inter-agent communication/cooperation assumes one conversation per request exchanged in a supplier-client pair, and allows communication of requests (from the client, and respecting to an appearing network job) and communication of rescheduling requests (from the client or from the supplier, and respecting to a previously established network job). In particular for rescheduling, there can be series of stages where a pair of agents can reach an agreement on a new (inter-agent) due-date. This can be considered a limited form of negotiation (however, this is not a contract-net protocol, like the one presented in [Smith 1988]). See [Reis 1999a] and [Reis 1999b], for more details about interaction states and the protocol, and [Reis 1998] for details about capacity states, including capacity management.

The component labeled *scheduling behavior* is responsible for the way the agent governs its action. This includes taking scheduling decisions attending to individual scheduling preferences and interacting with the other agents, based on the temporal constraints and the capacity and interaction states. In its simplest form, this component can be a set of dispatch rules used for capacity allocation to tasks, or it can involve a more sophisticated scheduling system, with planning and reasoning capabilities. In the first case we will have a more reactive agent and, in the latter case, an agent of a more cognitive kind.

In [Reis 1999a] internal structures were proposed for describing the interaction state, that allow decoupling of the steps of request reception, task scheduling and requests sending to suppliers, for each of the scheduling problems. The decoupling can allow the agent to focus its attention on the scheduling problems in a more fine-grained and prioritized way, and so, allowing a more *micro-opportunistic* (see [Sadeh 1994]) kind of scheduling agent.

3. Getting a Local Perspective of Hard Temporal Global Constraints

This section describes, first, how each capacity agent, g_k, in our MACS system maintains the hard temporal limits for each of its tasks, $\mathcal{O}^{k}_{i_r,r}$ (*i.e.*, the $DD^{k}_{i_r,r}$ and the $RD^{k,j}_{i_r,r}$ dates in each agent scheduling problem, $\mathcal{PE}^{k}_{i_r,r}$). Then it is shown how this information is used to avoid the exploration of inconsistent solutions from a temporal scheduling perspective.

3.1. Getting and Maintaining Local Hard Temporal Limits

When a new scheduling problem is launched by the supervision agent, agents propagate requests upstream the network, starting in a given retail agent. In the example of Figure 5, this is shown by the solid arcs with arrows, labeled with $\mathbb{d}$ symbols. First, the supervision agent sends a request to retail agent g_{14}. Then requests propagate from g_{14} to g_1, from g_1 to g_4, from g_4 to g_7, from g_7 to g_8 and g_9, from g_9 to g_{11} and g_{12}, and then from g_8, g_{11} and g_{12} to raw-material agents g_{17}, g_{18} and g_{19}, respectively. Finally, the supervision agent receives requests from the raw-material agents.

The retail agent and the raw-material agents receive, additionally, from the supervision agent, the global due-date ($DD_{i_r,r}$), and the global release date ($RD_{i_r,r}$), respectively. Also, each request message[4] sent by client g_i to agent g_k, conveying a request $\mathbb{d}^{i,k}_{i_r,r}$, must accommodate, additionally, the value of the external downstream slack for g_k, $FEJ^{k}_{i_r,r}$. Similarly, the acceptance message[4] sent by each supplier agent g_j to agent g_k, responding to request $\mathbb{d}^{k,j}_{i_r,r}$ made before by g_k, will accommodate the external upstream slack (with g_j) for g_k, $FEM^{k,j}_{i_r,r}$. Agent g_k does the following:

a) Computes the date $DD^{k}_{i_r,r} = TIME(\mathbb{d}^{i,k}_{i_r,r}) + FEJ^{k}_{i_r,r}$;

b) Computes the dates $RD^{k,j}_{i_r,r} = TIME(\mathbb{d}^{k,j}_{i_r,r}) - FEM^{k,j}_{i_r,r}$.

Whenever g_k is involved in rescheduling,[4] the values of the temporal slacks mentioned above can change, but the dates computed in a) and b) remain fixed (assuming the global due-date and release date, $RD_{i_r,r}$ and $DD_{i_r,r}$, supplied by the supervision agent won't change). For task $\mathbb{O}^{k}_{i_r,r}$ of agent g_k, $DD^{k}_{i_r,r}$ and $\underset{j=1,\ldots,F^{k}_{i_r,r}}{MAX}(RD^{k,j}_{i_r,r})$ are the right and left temporal hard limits, respectively. Every agent involved in network job $\mathbb{RT}_{i_r,r}$ must be able to compute those limits. So, agent g_k itself must also perform transmission of temporal slacks to suppliers and to the client. For this reason, the agent must compute and send:

a) The *downstream-upstream slack*, $FJM^{k,j}_{i_r,r} = FJ^{k}_{i_r,r} + fim^{k,j}_{i_r,r}$, to supplier g_j;

b) The *upstream-downstream slack*, $FMJ^{k,i}_{i_r,r} = FM^{k}_{i_r,r} + fij^{k}_{i_r,r}$, to client g_i,

where $FJ^{k}_{i_r,r}$ is the *downstream slack*, and $FM^{k}_{i_r,r}$ is the *upstream slack*, defined by (see also Figure 3):

$$FJ^{k}_{i_r,r} = FEJ^{k}_{i_r,r} + fij^{k}_{i_r,r},$$

$$FM^{k}_{i_r,r} = \underset{j=1,\ldots,F^{k}_{i_r,r}}{MIN}(FM^{k,j}_{i_r,r}), \quad \text{and}$$

$$FM^{k,j}_{i_r,r} = FEM^{k,j}_{i_r,r} + fim^{k,j}_{i_r,r}.$$

Note that the values transmitted by an agent, $FJM^{k,j}_{i_r,r}$ and $FMJ^{k,i}_{i_r,r}$, correspond to sums of temporal slacks inserted in the network schedule by agents downstream and upstream the network. So, although

[4] See the *supply request conversation* in [Reis 1999a].

they sum private information of the agent and several other agents, those values cannot be considered private information of any agent involved, in particular.

By convention, retail and raw-material agents deal only with global requests from, and to, the outside of the network,[5] they don't introduce internal slacks, and they just can perceive FEJ and FEM slacks in a network job. For the retail agent g_r, the FEJ slack is defined as:

$$FEJ^{r}_{i_r,r} = DD_{i_r,r} - TIME(dl^{0,r}_{i_r,r})$$

where $dl^{0,r}_{i_r,r}$ is the global request from outside. g_r will transmit the value of $FEJ^{r}_{i_r,r}$ as the value of the FJM slack to its supplier. For a raw-material agent involved in the network job $\mathbb{RT}_{i_r,r}$, g_m, the FEM slack is defined as:

$$FEM^{m}_{i_r,r} = TIME(dl^{m,0}_{i_r,r}) - RD_{i_r,r}$$

where $dl^{m,0}_{i_r,r}$ is the global request to outside. g_m will transmit the value of $FEM^{m}_{i_r,r}$ as the value of the FMJ slack to its client. Figure 5 illustrates, together with requests, the propagation of the slacks FJM (solid arcs), as well as the propagation of the slacks FMJ (dashed arcs).

Additionally, each capacity agent g_k can compute the *total slack*, $FT^{k}_{i_r,r}$, through:

$$FT^{k}_{i_r,r} = FJ^{k}_{i_r,r} + FM^{k}_{i_r,r}$$

The value of $FT^{k}_{i_r,r}$, although computed locally, can be used by the agents to estimate the *global temporal flexibility* of the scheduling problem: a higher value of the slack $FT^{k}_{i_r,r}$ points a less constrained problem, with the possibility of a higher number of feasible solutions,[6] and vice-versa. In general, the same value of $FT^{k}_{i_r,r}$ will show up for several agents involved in the same network job. The tasks of those agents will form one or more *quasi-critical paths*, in the case of $FT^{k}_{i_r,r} > 0$, or one or more *critical paths*, in the case of $FT^{k}_{i_r,r} = 0$. These two situations are illustrated for our network job example $\mathbb{RT}_{1,14}$ (see Figure 4, with interactions amongst network agents in Figure 5), in the results of the spreadsheet simulations shown in Figure 7 and Figure 8.

For the first case (Figure 7), the global scheduling problem has the following initial data: $TIME(dl^{0,14}_{1,14}) = 22$, $DD_{1,14} = 23$ and $RD_{1,14} = 0$ (the dd, DD and RD values of g_0, respectively, in the table labeled "input"). The resulting schedule is shown for the given values of the internal slacks (fij and fim) and durations (d) for each capacity agent tasks. In the area labeled "schedule evaluation", agents g_{11}, g_9, g_7, g_4 and g_1 are involved in a quasi critical path, and perceive the same total slack value of 11 (this can be confirmed in the "network data" below, in the figure).

In the second case (Figure 8) all temporal slack was removed by imposing zero values for all internal slacks, and changing the initial data to $TIME(dl^{0,14}_{1,14}) = 22$, $DD_{1,14} = 22$ and $RD_{1,14} = 10$. A critical path now appears, involving agents g_{11}, g_9, g_7, g_4 and g_1 (each one perceiving total slack values of 0).

[5] In a in a network job, these global requests can be seen as ficticious tasks with zero duration "assigned" to the retail and raw-material agents.

[6] The word "possibility" is used, because the agents must still account, additionally, for their capacity constraints.

3.2. Avoiding Inconsistent Solutions

Any feasible solution to a scheduling problem must respect capacity and temporal constraints. Even if the amount of capacity dedicated to each task in a network job is very high, both the temporal precedences and the hard temporal limits expressed by the RD and DD dates must be respected. This can be expressed, for each capacity agent, by the following constraints:

a) $FT^{k}_{i_r,r} \geq 0$,

b) $FJ^{k}_{i_r,r} \geq 0$,

c) $FM^{k}_{i_r,r} \geq 0$,

d) $fij^{k}_{i_r,r} \geq 0$, and

e) $fim^{k,j}_{i_r,r} \geq 0$.

During a rescheduling activity involving the agent, constraints d) or e) can be violated. However, this will be just a temporary situation, as the agent will then reschedule its task for an earlier time, in the first case, or for a later time, in the second case (and, if necessary, send rescheduling requests to the "opposite side", *i.e.*, to suppliers, or to the client, respectively).

By analyzing its local values of $FT^{k}_{i_r,r}$, $FJ^{k}_{i_r,r}$ and $FM^{k}_{i_r,r}$, an agent can conclude the following (and should act accordingly):

1. If $FT^{k}_{i_r,r} < 0$, the scheduling problem is overconstrained, and has no solution. In face of that, the agent must give up finding a feasible solution by canceling the corresponding product request. This avoids infinite propagation cycles, with rescheduling requests back and forth through the network, as it will lead to the unscheduling of every task of the $\mathbb{RT}_{i_r,r}$ network job.[7]

2. If either $FM^{k}_{i_r,r} < 0$ or $FJ^{k}_{i_r,r} < 0$, but still $FT^{k}_{i_r,r} \geq 0$, this points to a network schedule shifted to the left (too much internal slacks inserted downstream), or to the right (too much internal slacks inserted upstream) with respect to its global $RD_{i_r,r}$ and $DD_{i_r,r}$ limits.[8] It is worth then, for the agents to involve themselves in rescheduling activity, and they should do so, since there are still feasible solutions.[9]

By using these rules the agents of a EE network act cooperatively in scheduling and rescheduling. This is so because:

a) Each agent won't keep on searching for apparently feasible local solutions that are not part of a global feasible solution; furthermore, the agent will try to avoid other agents to do so.

b) Each agent will try to adjust its current local scheduling solutions (possibly giving up some individual scheduling preferences) so that the hard global temporal constraints will be respected.

---

[7] This is an extreme case, since it leads to the cancellation of the global request. In the model, temporal limits with the outside of the network are considered non-negotiable. However, the supervision agent can re-introduce the problem, re-defined with less constraining $DD_{i_r,r}$ and $RD_{i_r,r}$ dates.

[8] The same conditions can appear if the global scheduling problem is ill-formed, like $TIME(d^{0,r}_{i_r,r})$ being too far away from the $DD_{i_r,r}$ date, but this is just an abnormal case.

[9] At least from a purely temporal perspective (additionally, capacity constraints must also be respected).

4. Conclusion and Future Work

A model for a Multi-Agent Cooperative Scheduling (MACS) system for an Extended Enterprise (EE) context was described. At the level of the physical facilities the model emphasizes explicit representation of the production/distribution network resources, resource capacities and logistics tasks. At the level of the agents involved the model emphasizes coordination through inter-agent communication and a simple architecture for scheduling agents. The model also proposes an explicit representation of scheduling problems (depending on the individual agent perspective) which is suitable for a highly dynamic scheduling environment, and integrates scheduling with agent interaction.

A coordination mechanism was described for governing the action of the agents of a MACS system, in a purely temporal scheduling perspective. By exchanging values of specific inter-task temporal slacks, the agents are able to compute certain cutoff values, which point the limits of local consistent scheduling solutions, and can be used by the agents to act cooperatively. Note that the values exchanged cannot be considered private information of any agent involved, in particular.

The ideas exposed in this paper were tested in partial simulations like the one exposed in Figure 7 and Figure 8. A computational MACS system (still under construction, using the object oriented computer language CLOS) joins all pieces of our model. Part of this system is being used to test and improve the ideas exposed and to proceed with exploratory work in MACS problem solving.

In the near future, our work will include:

a) Work on coordination mechanisms, similar to the temporal slack mechanism presented, but resource/capacity related and work on individual agent scheduling behavior leading to a more fine-grained (and micro-opportunistic) kind of agent mental model;

b) Reactive scheduling capabilities, to accommodate schedules under execution and deal with real-time execution constraints. At present time, the model does not deal with real-time execution;

c) Introduce medium/long-term planning activities, which will bias the agents short-term scheduling activity (*e.g.*, storing products to cover a forecasted demand during a certain period). At present time, an EE network is solely driven by orders, as in a make-to-order environment;

d) Inter-agent negotiation, including multiple clients/suppliers, non-cooperative agents (*e.g.*, that don't give up its scheduling preferences), and negotiable temporal limits with the outside of the network;

e) Adapt the physical level of the model for just-in-time and repetitive manufacturing contexts;

f) Comparing the results of our work with others.

5. Acknowledgments

The authors thank to the anonymous reviewers for their comments and recommendations.

6. References

Arnold 1997 — Arnold, Jörg; Dudenhausen, Hans-Martin; Halmosi, Hans, 1997, *Production Planning and Control within Supply Chains*, Fraunhofer Institute Manufacturing Engineering and Automation, Stuttgart, Germany (ESPRIT project 20544, X-CITTIC).

Blazewicz 1994 — Blazewicz, J.;Ecker, K.H.;Schmidt, G.;Weglarz, J., 1994, *Scheduling in Computer and Manufacturing Systems*, Springer Verlag, 1994

Brucker 1998 — Brucker, Peter, 1998, *Scheduling Algorithms*, Springer-Verlag, Berlin, 1998.

Camarinha-Matos 1999 — Camarinha-Matos, Luís M.; Afsarmanesh, Hamideh, 1999, *The Virtual Enterprise Concept*, In Infrastructures for Virtual Enterprises, Networking Industrial Enterprises, Luís M. Camarinha-Matos, Hamideh Afsarmanesh (eds.), Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999, 3-14.

Fox 1993 — Fox, Mark S.; Chionglo, John F.; Barbuceanu, Mihai, 1993, *The Integrated Supply Chain Management System*, Department of Industrial Engineering, University of Toronto, Canada (from the Web site http://www.ie.utoronto.ca/EIL/iscm-descr.html).

Graves 1993 — Graves, S.C.; Kan, A.H.G. Rinnooy; Zipkin, P.H., (Eds.), *Logistics of Production and Inventory*, Handbooks in Operations Research and Management Science, Volume 4, North-Holland, Amsterdam, 1993.

Hildum 1994 — Hildum, David W., 1994, *Flexibility in a Knowledge-Based System for Solving Dynamic Resource-Constrained Scheduling Problems*, PhD. Thesis/Technical Report 94-77, Multi-Agent Systems Laboratory, Department of Computer Science, University of Massachussetts at Amherst, USA, (file HILDUM-T.PS, from the Web site http://centaurus.cs.umass.edu/)

Hildum 1997 — Hildum, David W.; Sadeh, Norman M.; Laliberty, Thomas J.; McA'Nulty, John; Smith, Stephen F.; Kjenstad, Dag, 1996, *Blackboard Agents for Mixed-Initiative Management of Integrated Process-Planning/Production-Scheduling Solutions across the Supply Chain*, Proceedings of the AAAI-97, AAAI Press/The MIT Press, Menlo Park, California, USA 1000-1005.

ICMAS 1996 — ICMAS 1996, 1996, *Proceedings of the Second International Conference on Multi-Agent Systems (Contents)*, ICMAS-96, AAAI Press, Menlo Park, California, 1996.

Kjenstad 1998 — Kjenstad, Dag, 1998, *Coordinated Supply Chain Scheduling*, PhD. Thesis, Norwegian University of Science and Technology - NTNU, Department of Production and Quality Engineering, Trondheim, Norway, 1998.

Klen 1998 — Klen, A.P.; Spinosa, L.M.; Rabelo, R.J.; Ferreira, A. C., 1998, *Integrated Logistics Management Support System: An Advanced Coordination Functionality for the Virtual Environment*, Proceedings of the 5th International Workshop on Intelligent Manufacturing Systems, IMS'98, Gramado, Brasil, November 1998.

Morton 1993 — Morton, Thomas E.; Pentico, David W., 1993, *Heuristic Scheduling Systems, with Applications to Production Systems and Project Management*, John Wiley & Sons, Inc., USA, 1993.

O'Hare 1996 — O'Hare, G.M.P.; Jennings, N.R., 1996, *Foundations of Distributed Artificial Intelligence*, John Wiley & Sons, Inc., 1996, New York, USA.

O'Neill 1996 — O'Neill, H.; Sackett, P., 1996, *The Extended Enterprise Reference Framework*, Balanced Automation Systems II, Luís M. Camarinha-Matos and Hamideh Afsarmanesh (Eds.), 1996, Chapman & Hall, London, UK, pp 401-412

Parunak 1998 — Parunak, H. Van Dyke; VanderBok, Ray, 1998, *Modeling the Extended Supply Network*, Industrial Technology Institute (file ISAS98.PDF, from the Web site http://www.erim.org/cec/sched/schedpubs.htm).

Rabelo 1996 — Rabelo, R.J.; Camarinha-Matos, L.M., 1996, *Towards Agile Scheduling in Extended Enterprise*, Balanced Automation Systems II, Luís M. Camarinha-Matos and Hamideh Afsarmanesh (Eds.), 1996, Chapman & Hall, London, UK, pp 413-422

Rabelo 1998 — Rabelo, Ricardo J.; Camarinha-Matos, Luis M.; Afsarmanesh, Hamideh, 1998, *Multiagent Perspectives to Agile Scheduling*, Basys'98 – IEEE / IFIP International Conference on Balanced Automation Systems, Prague, Czech Republic, 26-28/08/98.

Reis 1998 — Reis, J.; Mamede, N.; O'Neill, H., 1998, *Ontologia para um Modelo de Planeamento e Controlo na Empresa Estendida*, Proceedings of the IBERAMIA'98, Ibero-American Conference on Artificial Intelligence, Lisbon, Portugal, October 5-9, 1998, Helder Coelho (Ed.), Edições Colibri, Lisbon, Portugal, pp. 43-54 (in portugese).

Reis 1999a — Reis, J.; Mamede, N.; O'Neill, H., 1999, *Agent Communication for Scheduling in the Extended Enterprise*, In Infrastructures for Virtual Enterprises, Proceedings of the IFIP TC5 WG5.3/PRODNET

Working Conference on Infrastructures for Virtual Enterprises (PRO-VE'99), Porto, Portugal, October 27-28, 1999, Luís M. Camarinha-Matos, Hamideh Afsarmanesh (eds.), Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999, pp 353-364.

Reis 1999b — Reis, J.; Mamede, N., 1999, *What's in a Node, Nodes and Agents in Logistic Networks*, Proceedings of the ICEIS'99, 1st International Conference on Enterprise Information Systems, Setúbal, Portugal, Mars 27-30, 1999, Joaquim Filipe and José Cordeiro (Eds.), pp. 285-291.

Sadeh 1994 — Sadeh, Norman, 1994, *Micro-Oportunistic Scheduling: The Micro-Boss Factory Scheduler*, Intelligent Scheduling, Morgan Kaufman, 1994, Cap. 4

Smith 1988 — Smith, Reid G., 1988, *The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver*, Readings in Distributed Artificial Intelligence, Bond, Alan H.; Les Gasser (eds.), Morgan Kaufman Publishers, Inc., San Mateo, California, 1988, 357-366.

Tate 1995 — Tate, Austin; Drabble, Brian; Dalton, Jeff, 1995, *An Engineer's Approach to the Application of Knowledge Based Planning and Scheduling Techniques to Logistics - The O-Plan Project*, Artificial Intelligence Applications Institute, The University of Edinburgh, 14 July 1995.

Thomas 1996 — Thomas, Douglas J.; Griffin, Paul M., 1996, *Coordinated Supply Chain Management*, European Journal of Operational Research, 94 (1996) 1-15.

Williams 1981 — Williams, Jack F., 1981, *Heuristic Techniques for Simultaneous Scheduling of Production and Distribution in Multi-Echelon Structures: Theory and Empirical Comparisons*, Management Science, Vol. 27, No. 3, March 1981, 336-352.

Zweben 1994 — Zweben, Monte; Fox, Mark S., 1994, *Intelligent Scheduling*, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1994.

7. Figures

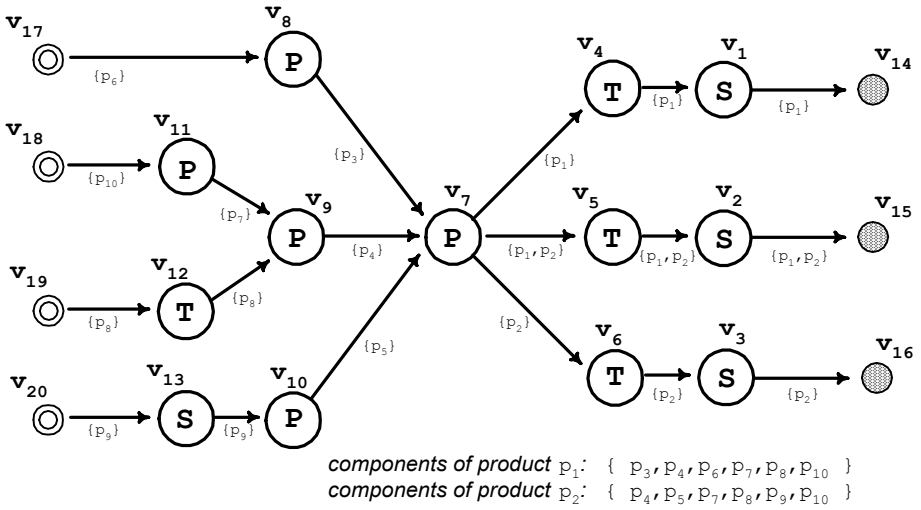

Figure 1- Physical network example.

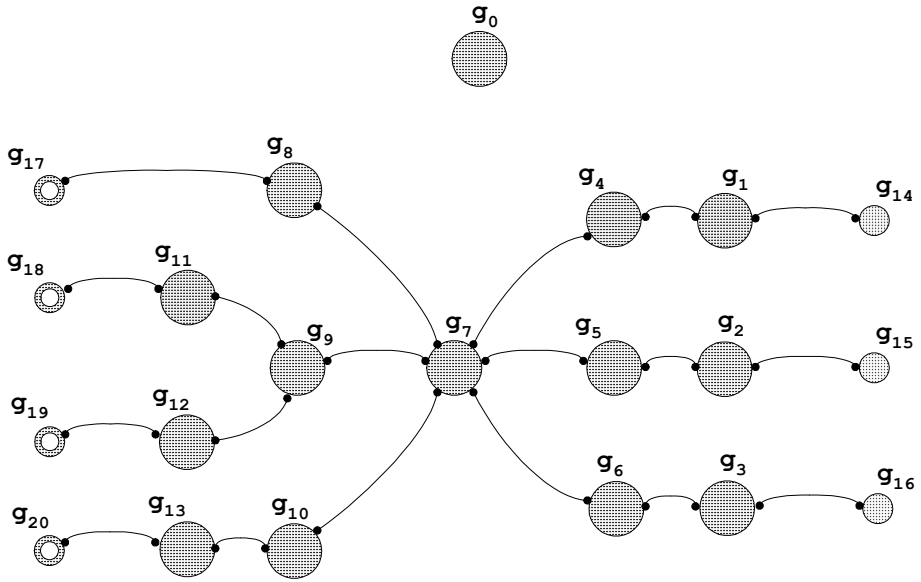

Figure 2- Agent network example (for simplicity, virtual arcs between the supervision agent and the other agents are not represented).

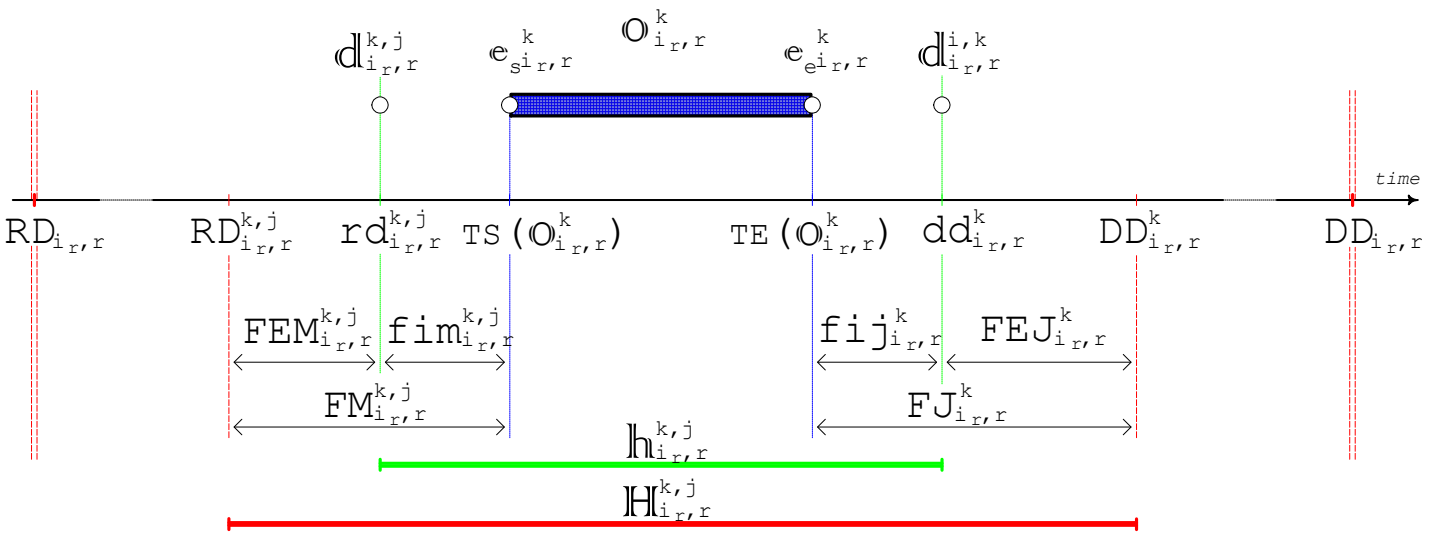

Figure 3- Events and temporal information associated to task, $O_{i_r,r}^{k}$.

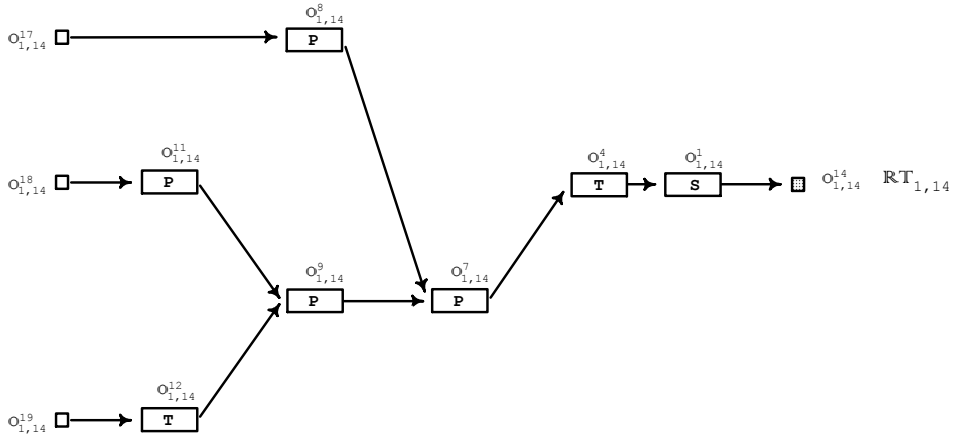

Figure 4- A network job.

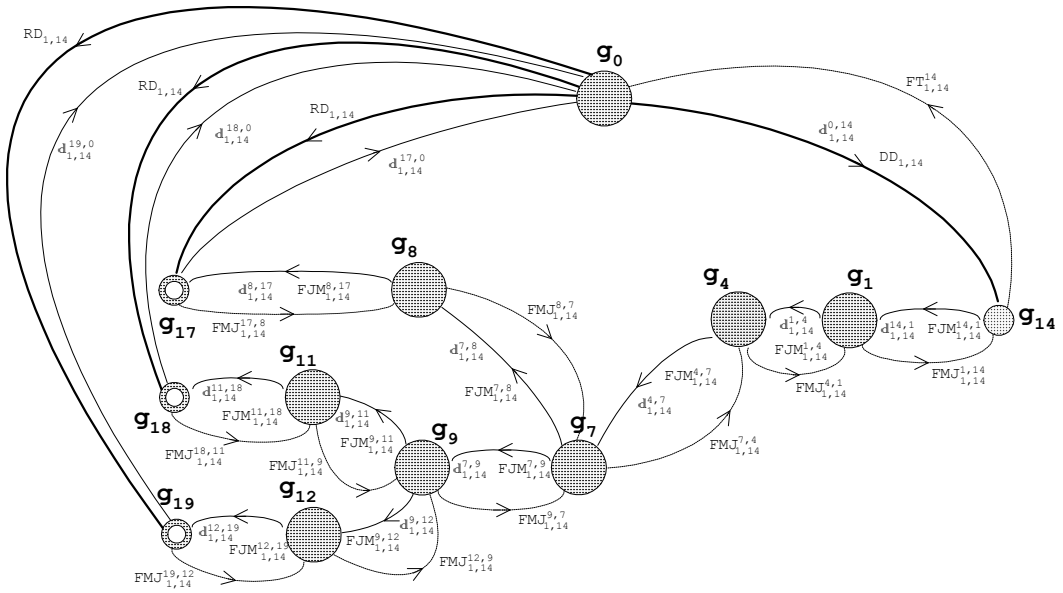

Figure 5- Inter-agent communication example.

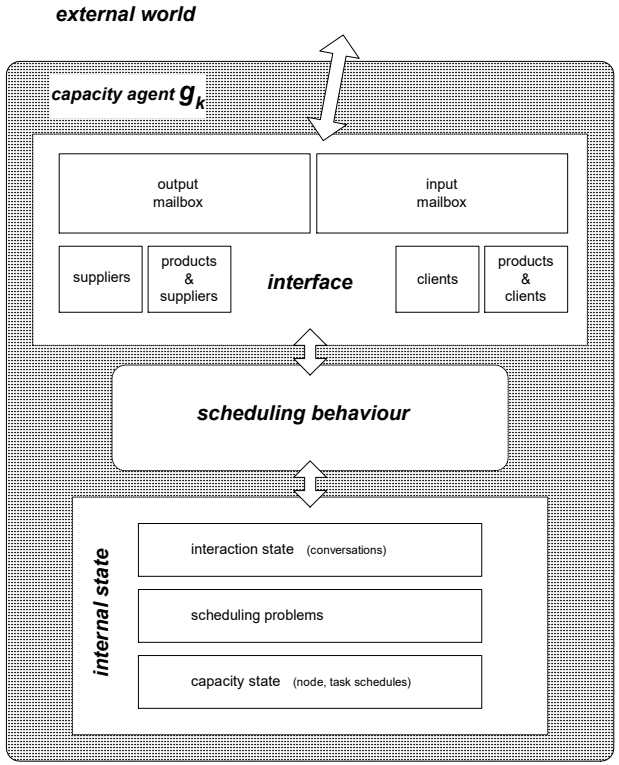

Figure 6- Architecture for a capacity agent.

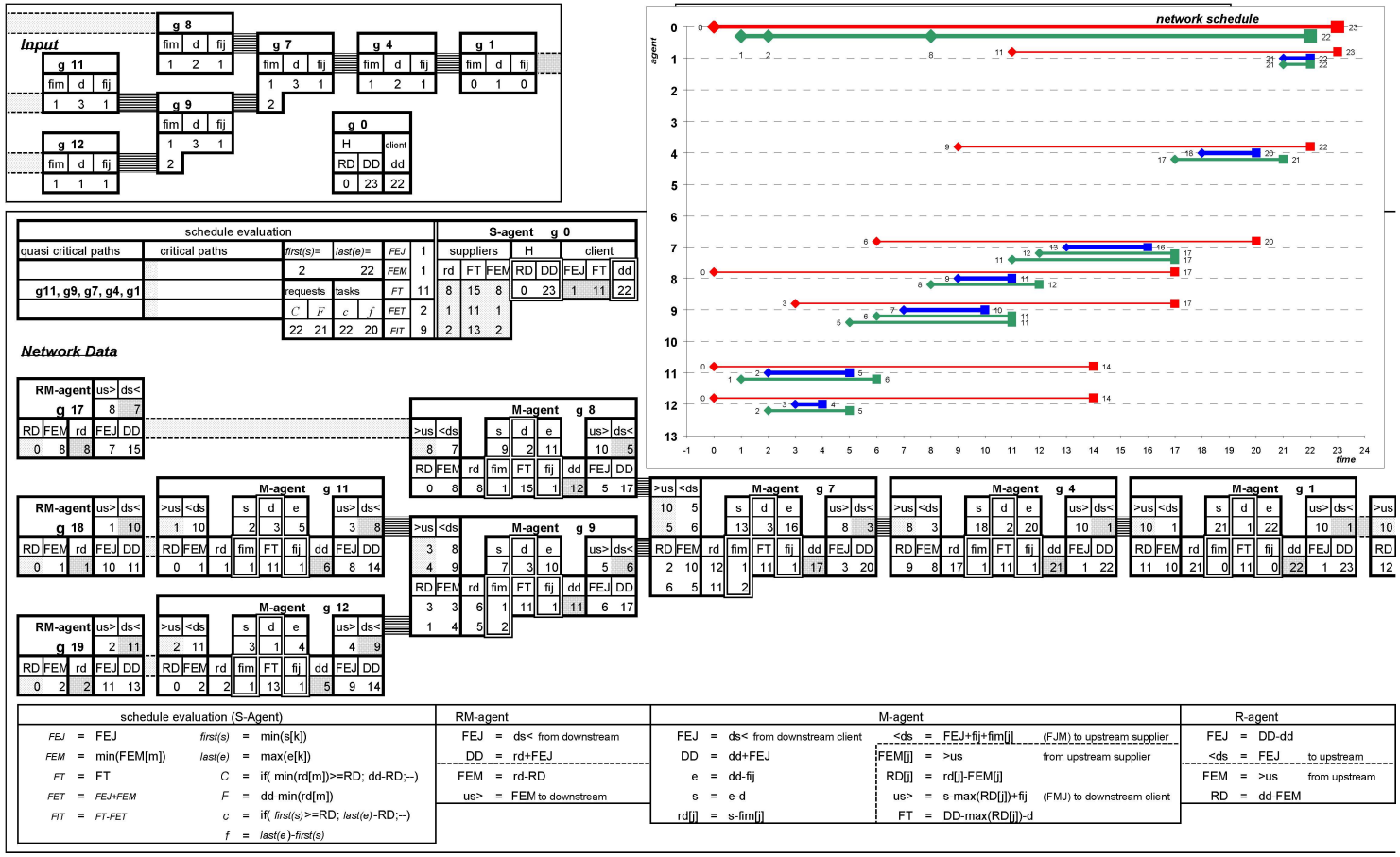

Figure 7- Network job example simulation (with quasi critical paths).

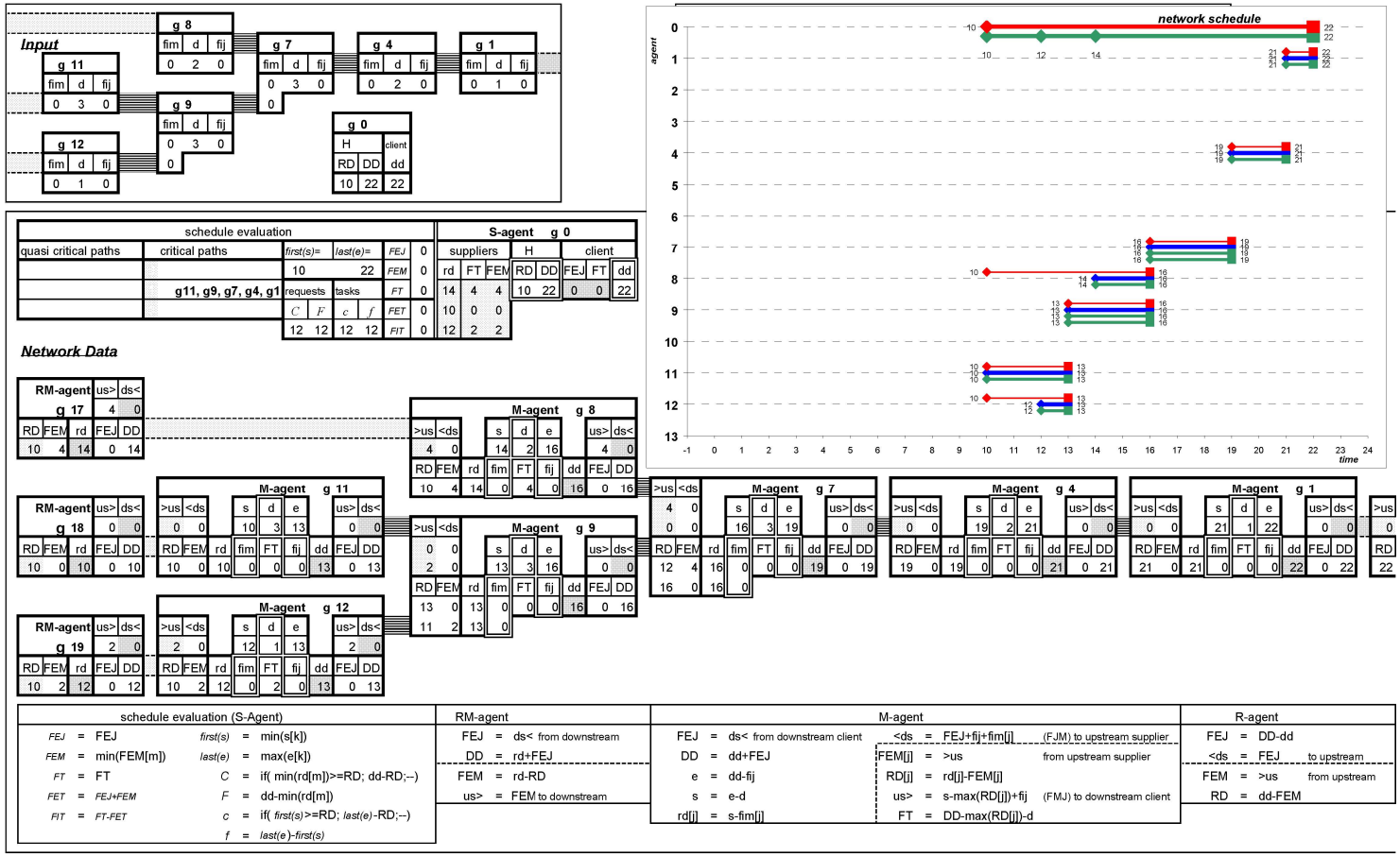

Figure 8- Network job example simulation (with critical paths)