Negotiation in Multi-Agent System using Partial-Order Schedule

Ritu Sindhu[1], Abdul Wahid[2], G.N.Purohit[3]

[1]Ph.D Scholar, Banasthali University
Rajasthan, India

[2]Department of CS, Gautambudh University
Ghaziabad, India

[3]G.N. Purohit, Dean, Banasthali University
Rajasthan,India

Abstract

In systems composed of multiple autonomous agents, negotiation is a key form of interaction that enables groups of agents to arrive at a mutual agreement regarding some belief, goal or plan, for example. In general, multi-linked negotiation (including both the directly linked and the indirectly linked relationships) describes situations where one agent needs to negotiate with multiple agents about different issues, where the negotiation over one issue influences the negotiations over other issues. The characteristics of the commitment on one issue affect the evaluation of a commitment or the construction of a proposal for another issue. In this paper we built a partial order schedule representation with the help of which we effectively manage interacting negotiation issues.Also we explored how flexibility is an important factor for ordering and managing negotiation issues in a successful negotiation and enables an agent to reason explicitly about the interactions among multiple negotiation issues in order to achieve higher performance.

Keywords: *Multi agent systems, Partial order schedule, ACL, Negotiation.*

1. Introduction

For understanding negotiation clearly we should know that there are three broad topics for research on negotiation ,that serve to organize the issues under consideration. First, negotiation protocols are the set of rules that govern the interaction. This covers, the permissible types of participants (e.g., the negotiators and relevant third parties), the negotiation states (e.g., accepting bids, negotiation closed), the events that cause state transitions (e.g., no more bidders, bid accepted), and the valid actions of the participants in particular states (e.g., which can be sent by whom, to whom and at when).

Second, negotiation objects are the range of issues over which agreement must be reached.

The next level, however, offers flexibility to change the values of the issues in the negotiation object, through counter-proposals, changing the structure of the negotiation object (by adding guarantees, for example), and so on. Finally, the agents' reasoning models provide the decision making apparatus by which participants attempt to achieve their objectives.

For a negotiation to be completed successfully all parties (i.e. agents) must clearly understand the rules of engagement or negotiation protocol. For example, in a simple contract-net protocol, in which a manager issues a call for proposals and waits for a full set of replies (or timeouts), each bidder must be prepared to honour its bid for the duration of the bid's validity. Otherwise, acceptance of the bid will require a second negotiation, which may itself succeed or fail. For example, KQML It is clear that the design of a communications language can restrict or enable different forms of high level reasoning among the agents involved in negotiation. For example, in KQML the sender must decide whether the recipient will respond directly, broker the query, recommends an agent to send the query to, or

recruit an agent who will send response directly to the original agent. The relationships among these negotiation issues can be classified as directly-linked relationship in which first issue affects second issue directly because later issue B is a necessary resource (or a subtask) of former issue, like (such as cost, duration and quality).Secondly indirect-linked relationship in which one issue relates to another issue because they compete for use of a common resource. and at last a facilitates relationship that is the combination of both relationship as described in figure in which the relationship from "Y11" to "Y12" means that the completion of "Y11" will positively affect the execution "y12" by reducing its cost, shortening its process time and/or improving its quality.

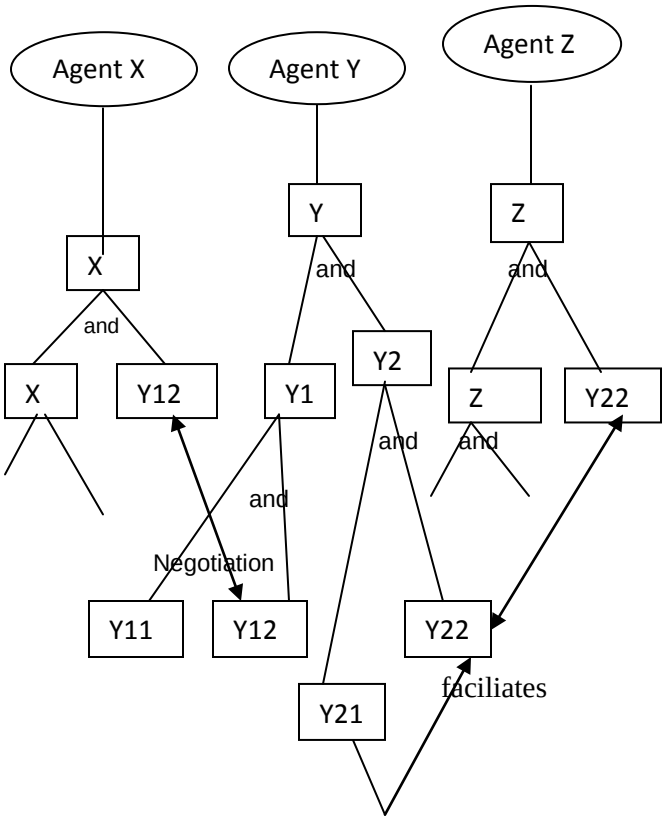


Fig .1 Negotiations between Agents with using facilitates Relationship

In general, multi-linked negotiation (describes situations where one agent needs to negotiate with multiple agents about different issues, where the negotiation over one issue has influence on the negotiation so over other issues.

How can the agent deal with these interrelated negotiations? One approach is to deal with these **negotiations independently ignoring their** interactions.1 If these negotiations are performed concurrently, there could be possible conflicts among the solutions to these negotiations; hence the agent may not be able to find a combined feasible solution that satisfies all constraints without re-negotiation over some already ''settled'' issues. For example, in Figure 1, suppose the Car Producer Agent negotiates with the Consumer Agent and promises to finish Purchase Car by time 20, and concurrently the Car Producer Agent also negotiates with the Transporter Agent about task Deliver Car and gets a contract that task Deliver Car will be finished at time 30, then the Car Producer Agent will find it is impossible for task Par Computer be finished by time 20 given that its subtask Deliver Car will not be finished until time 30.
If done effectively, its permits the agent to minimize the possibility of conflicts among the different negotiations and thus achieve better performance. The multi-linked negotiation problem also an important one because it actually happens in a number of application domains. For example, in a supply chain problem, negotiations go on among more than two agents. The consumer agent negotiates with the producer agent, and the producer agent needs to negotiate with the supplier agents. The negotiations between the producer agent and the supplier agents have a direct influence on the negotiation between the producer agent and the consumer agent.

In general, a Multi-linked negotiation problem occurs when an agent needs to negotiate with multiple other agents about different subjects, and the negotiation over one subject has influence on the negotiations over other subjects. The commitment of resources for one subject affects the evaluation of a commitment or the Construction of a proposal for another subject.

2. Negotiating agents' communication

An agent Communication Language (ACL) is a language with precisely defined syntax, semantics and pragmatics that is the basis of communication between independently designed and developed software agents [8]. Functional agents in a MAS use a common ACL to transfer information, share knowledge and negotiate with each other. Knowledge Query and Manipulation Language (KQML) and the ACL defined by Foundation for Intelligent Physical Agents/ Agent Communication Language (FIPA ACL) are the most widely used and studied ACLs. Each of them offers a minimal set of performatives to describe agent actions and allows users to extend them if the new defined ones conform to the rules of ACL syntax and semantics. In KQML there are no predefined performatives for agent negotiation actions. In FIPA ACL there are some performatives, such as proposal, CFP and so on, for general agent negotiation processes, but they are not sufficient for our purposes. For example, there are no performatives to handle third party negotiation. In this section we present a negotiation performative set designed for MAS dealing with supply chain management.

2.1 Criteria for performative definition and selection

The criteria we used to define negotiation performatives are the following:

1. Compatible with existing performatives. From a practical perspective, to extend either KQML or FIPA ACL performative sets involve a similar process. Since in FIPAACL there is a category containing four negotiation performatives we choose to construct the negotiation performative set based on this subset.

2. Defining new negotiation performatives based on a negotiation protocol. There is no clear means to judge the advantages and disadvantages of a particular extension of a standard.
ACL, just as it is difficult to judge how to add new words and phrases to a language used by human beings.

2.2 Negotiating agents communication or Negotiating performance for pair-wise negotiation protocol

In a supply chain negotiation process, negotiating agents use an Agent Communication Language (ACL) [5] to bargain with each other. The table below presents the peformatives designed for the negotiating agents based on FIPA ACL [4]. A negotiation protocol, formally described using Color Petri Net (CPN) is also given. Also Performatives for pair-wise negotiation protocol are used when two functional agents negotiate directly. The performatives definitions conform to the FIPA ACL specification. We explain their name and corresponding meaning as follows:

- Accept Proposal: The action of accepting a previously submitted proposal to perform an action
- *CFP:* The action of calling for proposals to perform a given action
- *Proposal:* The action of submitting a proposal to perform a certain action, given certain preconditions
- *Reject-proposal*: The action of rejecting a proposal to perform some acting during a negotiation
- *Terminate*: The action to finish the negotiation process

Initially, one agent starts negotiation by sending a CFP message to the other agent. After several rounds of conversation in which proposes and counter-propose are exchanged, the negotiation between two agents will end when one side accepts (rejects) the other side's proposal or terminates the negotiation process without any further explanation. It is not necessary that the functional agent responds to each message. A functional agent can simply ignore the incoming messages. It is the sender's responsibility to handle the lost message or in cases of lack of

responses. The conversation scenario is described in the following figure:
The pair-wise negotiation protocol simulates a conversation between two persons, in which one Side sends an "ask" and the other side send a "reply." The difference is in the pair-wise negotiation protocol we have to limit the response message types after a functional agent receives incoming messages so that the negotiation process does not become irrelevant to the topic, and at the same time simplify the message handling process. The expected responses or performatives performed by Pair-wise protocol when certain action comes:

- Accept Proposal: Terminate | NONE;
- *CFP :P*roposal | Terminate | NONE;
- *Proposal:* Accept-proposal | Reject-proposal | Terminate | NONE;
- Reject: Terminate | NONE;
- Terminate: NONE.

2.3 Negotiating performance for third party negotiation protocol Performatives

Negotiating performance for third party negotiation protocol Performatives for a third party negotiation protocol are used when functional agents negotiate through the third party (auctioneer). Some performatives defined for the pair-wise negotiation protocol, e.g. accept-proposal, reject-proposal, are still used for this protocol. One new performative, BID, is introduced. The syntax of BID is as follows:
Bid: the action for a bidder to send a corresponding response to an auctioneer
Bid
: sender <word>
: receiver <word>----------auctioneer
: content <expression>----------price for a goods
: Language <word>----------e.g. Knowledge Interchange Format (KIF)
: ontology <word>----------system
: in-reply-to <word>----------auction number
: protocol <word>----------t he default value is English auction.

Initially, one functional agent (seller) starts the negotiation by sending an INFORM message to the auctioneer. This message includes the goods that it wants to sell and the highest desired price (Or the contract that it wants to be bought and the lowest desired price) and the preferred auction protocol. After receiving the message, the auctioneer will broadcast it to potential bidders (assuming the auctioneer knows that information by querying the information agent) and organize an auction according to the requirement the seller submits. After several rounds of conversation, the negotiation process will end with a deal that was reached between seller agent and bidders. It is the auctioneer's responsibility to notify both the seller and bidders of the winner and the losers.

The scenario is described in the following figure:

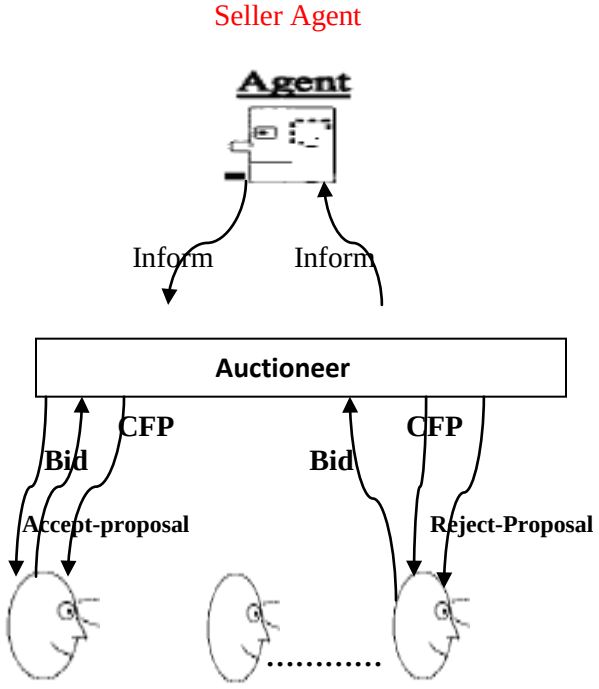


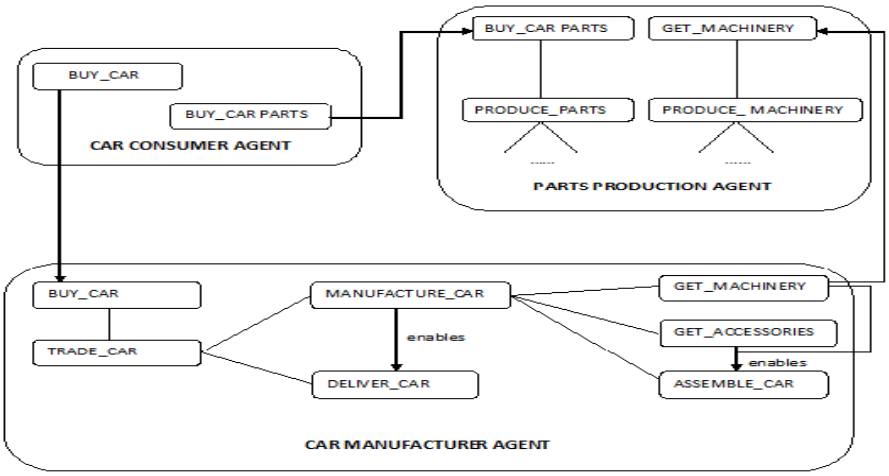
Fig2. Scenario between Agent and Third Party

3. The Agent Scenario using SCM

Consider the following example shown in Figure 1, which is a simplified supply chain containing five agents. The consumer (dealer) agent represents the environment that generates tasks

to be completed by the other four agents. The manufacturing agent, the production agent, the purchase agent and finally the inventory agent. When a new task is generated by the consumer Agent, it indicates how much it is worth and its deadline.

When the Car Producer Agent (manufacturing agent) receives a task Purchase Car from the Consumer(Dealer) Agent, it also needs to sub-contract parts of the task Get Hardware and Deliver car to the Hardware Producer Agent (production agent+ purchase agent) and the Transporter Agent(inventory agent)respectively. The following three negotiations are interrelated: the negotiation .



THREE AGENTS

between the Car Producer Agent and the Consumer Agent(dealer) on task Purchase car, the negotiation between the Car Producer Agent and the Hardware Producer Agent(production agent+ purchase agent) on task Get Hardware, and the negotiation between Car Producer Agent and the Transporter Agent(inventory agent) on task Deliver Car.

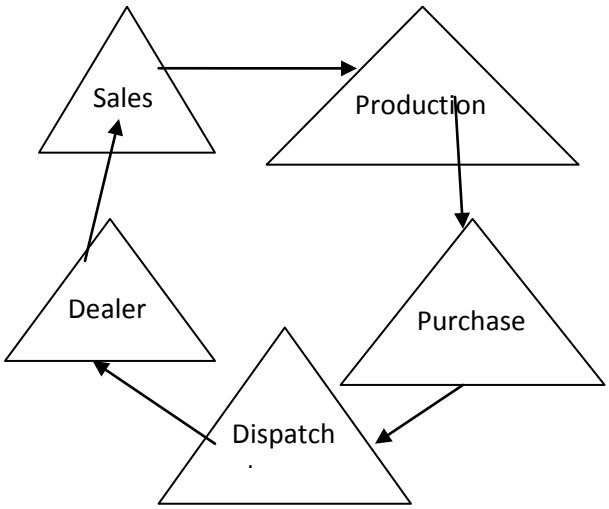


Fig3.Supply chain example

There are three agents in Figure 3:

1. **Car Consumer Agent** generates three types of new tasks: Purchase Car task for Car Production Agent, Get Machinery task for Parts

Production Agent, and Deliver car for Transporter Agent.

2. **Car Manufacturer Agent** receives the Purchase Car task from Car Consumer Agent, and needs to decide if it should accept this task and, if it does, what the promised finish time of the task should be. Figure shows the local plan for producing computers; it includes a non-local task Get Machinery that requires negotiation with Parts Production Agent. It also includes a non-local task Deliver Car that requires negotiation with Transporter Agent.

3**. Parts Production Agent** receives two types of tasks: Get Machinery from Car Manufacturer Agent and Buy Car Parts from Car Consumer Agent. It needs to decide whether to accept a new task and what is its promised finish time.

4. **Other Agents**

There are two other agents also involved in the process:

a) **Transporter Agent**
Its task is to deliver car from Car Manufacturer Agent to Car Consumer Agent.

b) **Trader Agent**
Its task is to establish a trade between Car Consumer Agent and Car Manufacturer Agent for buying a new car.

We first define two generalized terms to make the following description easier. In the following description, we will use the term contractor agent to refer to the agent who performs the task for another agent and gets rewarded for the successful completion of the task; and contractee agent to refer to the agent who has a task that needs to be performed by another agent and pays a reward to the other agent. The contractor agent and the contractee agent negotiate about a task and a contract is signed (a commitment is built and confirmed) if an agreement is reached during the negotiation. In this work, the negotiation process between agents is based on an extended contract net model.

- Contractee agent announces a task by sending out a proposal.
- Contractor agent receives this proposal, evaluates it, responds to it in one of three ways: by accepting it, by simply rejecting it, or by rejecting it but at the same time making a counter-proposal.
- Contractee agent evaluates the responses, it either chooses to confirm an accepted proposal, or chooses to accept a counter-proposal.
- Contractee agent awards the task to the chosen contractor agent based on the commitment (the mutually accepted
- Upon proposal or counter-proposal) which is confirmed by both agents; the negotiation process then ends successfully. If a mutually agreed proposal/counter-proposal cannot be found, the negotiation process fails.

This process can be extended to a multi-step process by introducing an extended series of alternative proposals and counter-proposals. However, in this paper, we only focus on the two-step (proposal, counter-proposal) negotiation process. A proposal which announces that a task (t) needs to be performed includes the following attributes:

1. **earliest start time (est):** the earliest start time of task t; task t cannot be started before time est.
2. **Deadline (dl):** the latest finish time of the task; the task needs to be finished before the deadline dl.
1. **Minimum quality requirement (minq):** the task needs to be finished with a quality achievement no less than minq.
2. **Regular reward (r):** if the task is finished as the contract requested, the contractor agent will get reward r.
3. **Early finish reward rate (e):** if the contractor agent can finish the task by the time (ft) as it promised in the contract, it will get the extra early finish reward.(e*r*(dl-ft),r).
4. **Decommitment penalty rate (p)**: if the contractor agent cannot perform the

task as promised in the contract (i.e. the task could not be finished by the promised finish time), it pays a decommitment penalty (p * r) to the contractee agent. Similarly, if the contractee agent needs to cancel the contract after it has been confirmed, it also needs to pay a decommitment penalty (p* r) to the contractor agent.

Suppose Car Manufacturer Agent has received the following two tasks in the same scheduling time window.

task name : Purchase Car A
arrival time : 10
earliest start time : 15(arrival time +estimated negotiation time (7))
deadline : 50
reward : r = 10
decommitment penalty : p = 0.8
early finish reward rate: e = 0.01

task name: Purchase Car B
arrival time: 12
earliest start time: 17 (arrival time +estimated negotiation time (5))
deadline: 80
reward: r = 10
decommitment penalty rate: p = 0.9
early finish reward rate: e = 0

The agent's local scheduler reasons about these two new tasks according to the above information: their earliest start times, deadline, estimated process times and the rewards. It then generates the following agenda which includes the following tasks:

The Car Manufacturer Agent checks the local plans for these tasks as shown and finds there are five negotiations:

1. Negotiate with Car Consumer Agent about the promised finish time of Purchase Car_ A.

2. Negotiate with Car Consumer Agent about the promised finish time of Purchase Car _B.

3. Negotiate with Parts Production Agent about whether it can accept the task Get _Machinery_ A and if it accepts this task, what is the promised finish time.

4. Negotiate with Parts Production Agent about the task Get _Machinery _B, with the same concerns as above.

5. Negotiate with Transporter Agent about whether it can accept the task Deliver Car A, and if it accepts this task, what is the earliest start time and what the promised finish time is.

These five negotiations are all related. The potential relationships among multiple negotiation issues can be classified as two types.

4. Partial order schedule

A partial-order schedule is the basic reasoning tool that we use for multiple related negotiations. Here we present the formalizationof the partial-order schedule and use an example to explain how it works for a multi-linked negotiation Figure 5 shows the partial ordered schedule from the example in Figure 4.

A poset consists of a set together with a binary relation that indicates that, for certain pairs of elements in the set, one of the elements precedes the other. These relations are called partial orders to reflect the fact that not every pair of elements of a poset need be related: for some pairs, it may be that neither element precedes the other in the poset. Thus, partial orders generalize the more familiar total orders, in which every pair is related. A finite poset can be visualized through its Hasse diagram, which depicts the ordering relation between certain pairs of elements and allows one to reconstruct the whole partial order structure.A partial order is a binary relation "≤" over a set P which is reflexive, antisymmetric, and transitive, i.e., for

all a, b, and c in P, we have that: In mathematics, especially order theory, a partially ordered set (or poset) formalizes the intuitive concept of an ordering, sequencing, or arrangement of the elements of a set.

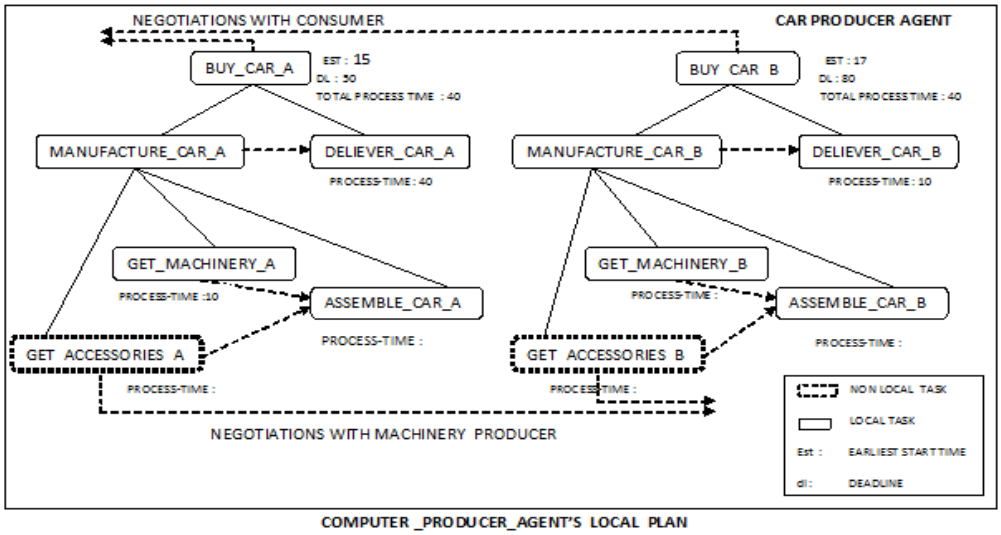


Fig: 4 **Car Producer Agent's Local Plan**

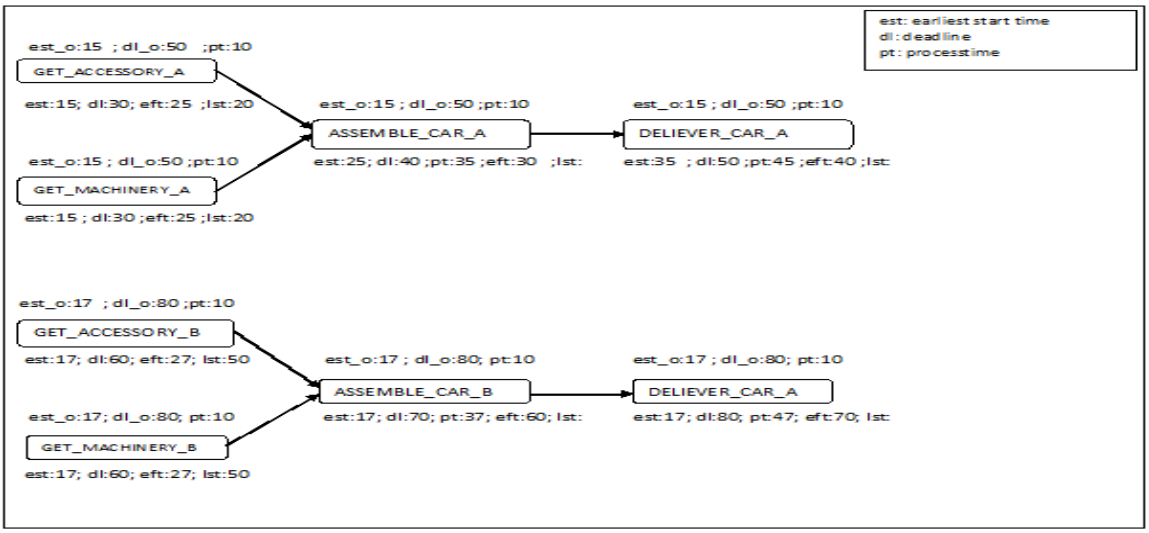


Fig: 5 The **Partial-Order Schedule of the Car Manufacturer Agent**

a ≤ a (reflexivity);
if a ≤ b and b ≤ a then a = b (antisymmetric);
if a ≤ b and b ≤ c then a ≤ c (transitivity).

In other words, a partial order is an antisymmetric preorder. In other words we can say that a Partial-Order Schedule represents a group of tasks with specified precedence relationship among them using a directed acyclic graph: G = (V;E). V = {u}, each vertex in V represents a task. E = {< u,v >/ u,v}. belongs E). Each edge (u, v) in E denotes the precedence relationship between task u and task v (P (u; v)), that is task u has to be finished before task v can start.

Some terms are used to explain how negotiation is done using partial order schedule.

Task (t) is represented as a node in the graph; it is the basic element of the schedule. A task (t) needs a certain amount of process time (t.process time). A task can be a local task or a nonlocal task: a local task is performed locally (i.e., the "Get Accessories A "task) and a nonlocal task (i.e. the "Get Machinery A" task) is performed elsewhere and hence does not consume local process time.
Pretasks of task t is a set of tasks that need to be finished before task t can start: Pre (t) = {s/s, € V □ < s,t> € E}, task t can start only after all tasks in Pre(t) have been finished. For example, the pre tasks of task "Install Accessories A" includes task "Get Machinery A" and task "Get Accessories A".
The Posttasks of task t is a set of tasks that only can start after task t has been finished: Post(t) = {r|r € V □ < t, r> € E}.For example, the post tasks of task "Assemble Car A" includes task "Deliver Car A".A task t has constraints of earliest-start-time (t.est) and deadline(t.dl).
The earliest-start-time of task t (t.est) is determined by the earliest-finish-time of its pre-tasks (eft[Pre(t)]) and its outside earliest-start-time constraint(t.est_o)6 t.est =max(eft[Pre(t)]; t.est_o);**The earliest-finish-time of a task t (t.ef t)** is defined as : t.ef t = t.est + t.process _time.
The earliest-finish-time of a set of tasks V (eft[V]) is defined as the earliest possible time to finish every task in the set V, it depends on the earliest-start-time and the duration of each task. For example, in Figure 6, outside-earliest-start-time constraint for task "Assemble Car A" is 10 (same as its super task 'Purchase Car A"), the earliest-finish-time for its pretasks is 20 (assume "Get Machinery A" could finish at its earliest possible time), then the earliest-start time for task "Install Accessories A" is 20.
The deadline of task t (t.dl) is determined by the latest-start time of its post tasks (lst[Post(t)]) and its outside-deadline-constraint(t.dl_ o):t:dl = min(lst[Post(t)], t.dl).
The latest-start-time of a task t (lst(t)) is defined as: t.lst = t.dl - t.process time.
The latest-start-time of a set of tasks V (lst[V]) is defined as the latest time for the tasks in this set to start without any task missing its deadline, it depends on the deadline and the duration of each task.
The Flexibility of Task t represents the freedom to move the task around in this schedule.
F(t) = (t.dl-t.est-ti.process_time)/ti.process_time.
For example, F(Get Accessories A) = (50-10-10)/10 = 3.
The Flexibility of a Schedule S measures the overall freedom of this schedule; it is the sum of the flexibility of each activity weighted by its process time of the process time of the schedule. The flexibility of the task with a longer process time has a bigger influence on the flexibility of the schedule.
F(S) ∑t€s F(t) *t.process time/(∑i ti.process_time).

5. Conclusion

In this abstract we have described an approach to modeling the supply chain management problem in the real business environment using software agents. We use the concept of negotiating agent to model the self-interested entities in the market place. The system framework we designed allows negotiating agents join, stay or leave the system freely. The

basic ideas and methods to attack the aspects of negotiating agent negotiation behaviors including the communication and problem solving parts have been given and studied. To deal with the multiple related negotiation issues, the agent needs to analyze the relationships among these negotiation issues and find what the influence of one issue on the others is. So for this the agent builds a partial-order schedule generated by the agent's local scheduler, so that the agent knows what these tasks are and how they are related to each other. The agent sorts its current negotiation issues according to their importance, their flexibilities or the difficulties of negotiation processes8, and finds the influence of the previous issue on the later issues Also we explored how flexibility is an important factor for ordering and managing negotiation issues in a successful negotiation so as to achieve higher performance. Negotiation performatives for pair-wise and third party protocol have been designed.

6. References

[1] Decker, K., Lesser, V. R. Quantitative Modeling of ComplexEnvironments. In International Journal of Intelligent Systemsin Accounting, Finance and Management. Special Issue onMathematical and Computational Models and Characteristicsof Agent Behavior., Volume 2, pp. 215-234, 1993.

[2] Deshmukh, A. V., Talavage, J. J., and Barash, M. M.Complexity in Manufacturing Systems: Part 1 - Analysis ofStatic Complexity IIE Transactions, vol 30, number 7, pp.645-655, 1998.

[3] Horling, Bryan, Lesser, Victor, Vincent, Regis. Multi-AgentSystem Simulation Framework. In 16th IMACS WorldCongress 2000 on Scientific Computation, AppliedMathematics and Simulation, EPFL, Lausanne, Switzerland,August 2000.

[4] Pritsker, A.A.B. GERT Networks (Graphical Evaluation andReview Technique) The Production Engineer, October 1968

[5] Sandholm, T. and Lesser, V. 1996. Advantages of a LeveledCommitment Contracting Protocol. Thirteenth NationalConference on Artificial Intelligence (AAAI-96), pp.126-133, Portland, OR, .

[6] Sandholm, T. 1996. Negotiation among Self-InterestedComputationally Limited Agents. Ph.D. Dissertation,University of Massachusetts at Amherst, Department ofComputer Science.

[7] Sandip Sen and Edmund H. Durfee A Formal Study ofDistributed Meeting Scheduling Group Decision andNegotiation, volume 7, pages 265-289, 1998.

[8] Wagner, Thomas and Lesser, Victor. Relating QuantifiedMotivations for Organizationally Situated Agents. InIntelligent Agents VI: Agent Theories, Architectures, and
Languages, Springer

[9] Vincent, R.; Horling, B.; Lesser, V. An Agent Infrastructureto Build and Evaluate Multi-Agent Systems: The Java AgentFramework and Multi-Agent System Simulator. In LectureNotes in Artificial Intelligence: Infrastructure for Agents,Multi-Agent Systems, and Scalable Multi-Agent Systems.Volume 1887, Wagner & Rana (eds.), Springer, pp. 102127,2000

[10] Zhang, Xiaoqin and Lesser, Victor. Multi-LinkedNegotiation in Multi-Agent Systems. Technicial Report ofComputer Science Department, UMass., TR-2002-02.

[11]Xiaoqin Zhang, Victor Lesser" Multi-Linked Negotiation in Multi-Agent Systems" AAMAS'02, July 15-19, 2002, Bologna, Italy.

First Author(Ritu Sindhu): Ph.D Scholar, Banasthali University, Rajasthan. Completed her B.Tech (CSE) from U.P.T.U, Lukhnow, M.Tech (CSE) from Banasthali University, Rajasthan.

Second Author(Abdul Wahid): Presently working as a A.professor in Computer science department in Gautambudh University, greater noida India. Completed his MCA, M.Tech. and Ph.D. in Computer Science from Jamia Millia Islamia (Central University), Delhi..

Third Author(G.N.Purohit): Dean, Banasthali University, Rajasthan