

Interactive Decision Support: Advantages of an Incomplete Utility Model

Richard Goodwin
Sesh Murthy

John Rachlin
Rama Akkiraju

IBM T. J. Watson Research Center
Yorktown Heights, NY
{rgoodwin,rachlin,murthy,akkiraju}@watson.ibm.com

Effective production scheduling is at the heart of any efficient manufacturing process and is the key to profitability. Since generating a feasible schedule can be computationally difficult, manufactures use must use computers to generate near optimal schedules. However, using a computer to find a single near optimal schedule requires a precise definition of optimality which must tradeoff the competing interests of production efficiency, customer satisfaction, profitability and product quality. Performing these tradeoffs requires subtle distinctions that are beyond the fidelity of any utility model that could be constructed with reasonable effort. In our experience, scheduling systems work best when they work as assistants to human schedulers, keeping track of complex constraints and presenting the human scheduler with a set of alternatives that highlight the tradeoffs. The computer must allow the human scheduler to examine and manipulate the resulting schedules in order for the human to arrive at the best schedule for execution. To support this effort, the utility model must capture the high level factors that contribute to a good schedule, but it does not, and could not, capture all the subtlety needed to select the best schedule. For manufacturing interests, scheduling objectives fall along four broad dimensions: **Time**, factors that relate to on time delivery, **Quality**, factors related to product quality, **Money**, factors related to profitability and **Disruptions**, factors related to the ease of manufacture. Not incidentally, these four dimensions map to the interests of the customer service representatives, quality engineers, accountants and manufacturing supervisors respectively. Our utility model aggregates factors that contribute to each category, but comparisons between categories are left to the human scheduler who is presented with a set of schedules that represent the Pareto-Optimal frontier. In this paper, we elaborate on our approach to interactive decision-support and illustrate it with examples from the IBM Load Planning System, which uses an asynchronous team of agents to create schedules for shipping manufactured goods to customers.

Introduction

Any utility function used to model preferences over real-world outcomes is necessarily incomplete, given limited resources to create the utility function. Such is the case in scheduling for manufacturing processes where the number of schedules is exponential in the size of the problem and determining the preferred schedule involves subtle tradeoffs between competing objectives. A complete utility function for this domain would specify a partial order over all possible schedules and outcomes. If there are n possible outcomes, then there are $n!$ complete orderings, and far more partial orderings, each of which is a possible utility model. In some cases, where the objective is to minimize a single, easily calculated value, such as cost, the utility model is relatively easy to elicitation and can be represented compactly. However, most problems involve tradeoffs between competing objectives to minimize cost, maximize on time delivery, minimize production disruptions and maximize customer satisfaction. The utility function can be arbitrarily complex and may not have a compact representation. Furthermore, the effort needed to elicit a complete utility model may not be justified by the expected benefit, even assuming that people representing competing interests could eventually agree on a preference ordering over outcomes.

In our approach to computer aided decision making, we recognize that utility models are incomplete and rely on human operators to make the subtle distinctions necessary to resolve tradeoffs. We see decision making in organizations as a process of negotiation that computers can support by supplying relevant information and suggesting good alternative solutions. The purpose of the utility function is to guide search and to identify potentially good solutions for presentation to the human decision maker. The partial utility model must capture the high level factors that contribute to a good schedule, but it does not have to compare dissimilar measures of utility that represent tradeoffs between competing interests. We use a multi-valued utility function that evaluates a schedule along a set of dimensions. For manufacturing interests, we have found that objectives can be evaluated along four broad dimensions: **Time**, factors that relate to on time delivery, **Quality**, factors related to product quality, **Money**, factors related to profitability and **Disruptions**, factors related to ease the of manufacture. Not incidentally, these four measures map to the interests of the customer service representatives, quality engineers, accountants and manufacturing supervisors respectively. The utility model is used to aggregate factors that contribute to each category, but comparisons between categories are left to the human scheduler who is presented with a set of schedules that represent the Pareto-optimal frontier. The scheduler can use these solutions as a basis for evaluating the tradeoffs, negotiating between competing interests and coming to a final decision.

In the rest of this paper, we elaborate on our approach to interactive decision-support and illustrate it with examples from the IBM Load Planning System. We begin by introducing the load planning problem, motivating its importance and describing the competing interests of the parties involved. We then describe our planning system, which uses an asynchronous team of agents (A-Teams) to create schedules for shipping manufactured goods to customers. The A-Team architecture provides a robust, scalable framework that allows the human expert scheduler to guide search, inject partial solutions and pose what if scenarios. These features are essential because they allow the scheduler to conduct negotiations with other people responsible for parts of the manufacturing process and with the manufacture's customers. Through examples, we show how the scheduler can negotiate to change the definition of the scheduling problem and allow for efficient solutions.

Load Planning

Manufactures in the process industries, such a steel and paper producers, need to ship their goods from the mill to their customers. Load planning is the process of creating a plan that assigns each item to be shipped to a particular kind of vehicle, selects a carrier and route for the vehicle and schedules the vehicle's departure from the mill. The manifest for each vehicle must satisfy constraints on weight, volume, the mixing of product types and the mixing of orders from customers that are competitors. Each carrier makes some number of vehicles, of particular types, available to the mill each day and each carrier charges different rates for delivery. The plan must only use the vehicles that are available, while respecting contractual obligations to give a certain volume of loads to each carrier in order to receive volume discounts. In addition, loading docks at the mill and at customers' sites have capacity limits on the number of vehicles, the types of vehicles and the amount of material that can be handled in a given shift. In addition, vehicles may be packed with items going to multiple locations, with some items dropped off on the way to the final destination. The routing of vehicles between destinations for these *drop shipments* affects both the arrival time and the cost. Load planning in this context is more than simple bin packing, but does include bin packing as a sub-problem.

The main objectives of load planning are to ensure on time delivery while minimizing cost. Load planning is of great concern to manufacturers since distribution costs represent a significant percentage of their revenue, with ten percent not being unusual. Reductions of a fraction of one percent in distribution costs translate into millions of dollars of profit over the course of a year. Manufactures also compete for customers primarily on the basis of cost and service, since their

goods are commodity items with almost uniform quality across the industry. Failure to meet delivery dates can cost a manufacture future business and is particularly important for customers that operate in a just-in-time fashion. However, secondary concerns about product quality and disruptions at the mill are still important. Product quality can be affected by the amount of time an item remains in inventory, the suitability of a particular kind of vehicle for transporting it and by the care that a carrier takes when transporting the material. The efficiency of the loading dock is also affected by the schedule. Each switch between loading trucks and loading rail cars disrupts the operations of the loading dock and adds delays, reducing capacity. Excess inventory at the dock reduces efficiency by making it hard to maneuver items into the vehicles.

Problem Definition

In general, the load planning system is given as input the following information:

- An **Orderbook** which defines attributes of each including destination, earliest and latest due date, physical dimensions and quantity.
- A list of available **vehicles and carriers** which defines the options for shipping and gives any contractual volume requirements.
- **Freight rates** which define the cost for shipping goods as a function of the weight and distance, the type of product, the type of vehicle and the carrier.
- **Loading rules** which determine whether a given set of items can legally be loaded into a given vehicle. These rules includes constraints on stacking, weight and product mixing.
- **Distances and transit times**, defined by mode, are used to determine arrival dates, given a ship date and to determine cost, given a freight rate in \$ per mile.
- A **production schedule** defines when items in production will be available to ship.
- An **inventory list** give items that are waiting to be shipped.
- **Dock shipping and receiving constraints** give limits on the number and type of vehicles and on the amount of material that can be handled at the mill and at each customer's receiving dock in a given shift.
- The **current load plan** which lists loads that were planned for shipment. Load planning is an incremental process with a scheduling horizon that may cover a week. Each day the load plan is updated to reflect new orders and changing production schedules. The five day schedule servers as a basis for predicting vehicle need and ordering vehicles from carriers. Substantial changes to the current plan can create problems if vehicles have already been ordered and customers notified to expect delivery on a specific day.

The goal is to produce a set of vehicle manifests, each one assigned to a particular carrier and scheduled for a particular shift at the loading dock. Sequencing of vehicles within a shift is not included because it is easier to do as the vehicles arrive at the dock and the exact location of each item in the staging area is known.

Objectives

Given the problem definition and the definition of a solution, the quality of the solution is determined by how well it satisfies the objectives of the company. We now focus on the problem of determining the objectives of the company and formulating these objectives in a way that can be used by the computer to guide search. In this section, we explain our approach to creating partial utility functions.

For load planning and related manufacturing process scheduling problems, the objectives of the manufacturer fall along four broad dimensions: **Time**, factors that relate to on time delivery, **Quality**, factors related to product quality, **Money**, factors related to profitability and **Disruptions**, factors related to the ease of manufacture. These four categories map to the interests of the customer service representatives, quality engineers, accountants and manufacturing supervisors respectively. Our utility model aggregates factors that contribute to each category, but comparisons between categories are left to the human scheduler who is presented with a set of schedules that represent the Pareto-Optimal frontier, represented by blocks in figure 1.

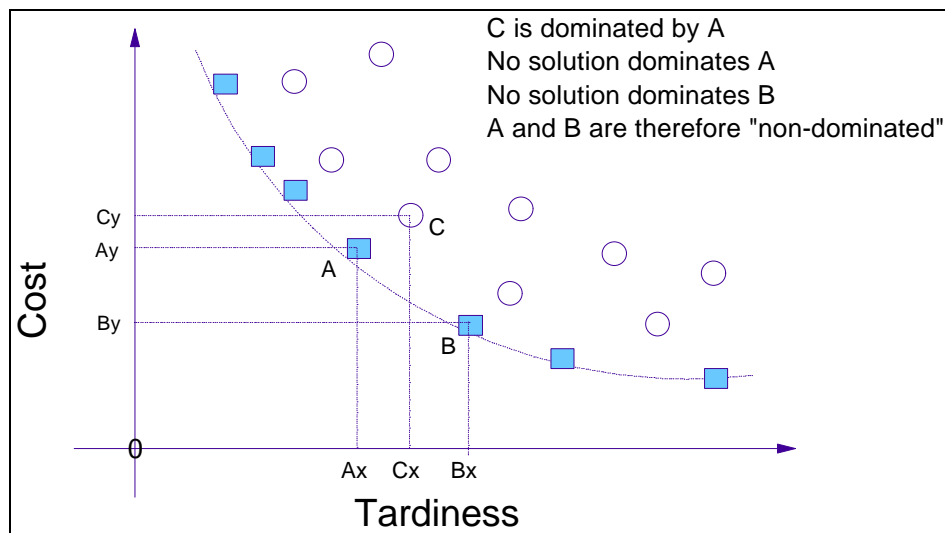


Figure 1: Non-dominated solutions form the Pareto-Optimal frontier.

Aggregating utility factors within each category to create a combined utility for each category is easier than aggregating across categories. Within each category, the utility function compares like measures that reflect the same basic objective and tend to be expressed in comparable units. Each category also maps to the direct interests of a small number of people who have similar personal objectives. Customer service representatives are evaluated on how well customers are satisfied and can give the relative importance of each measure of timeliness. product quality engineers can assess the relative impact that inventory delays and the use of specific vehicles have on product quality. Accountants can give the formula for calculating inventory holding costs and carrier rate structures give transportation costs, both in dollars. Finally, loading dock supervisors are evaluated on the throughput of the loading dock and can give the relative cost of each type of disruption.

Within each category, there are multiple measures of utility. Some measures are standard across the industry, while others are used because a particular company finds them easy to measure or believes that they better reflect the true objective. For example, tardiness can be measured as the number of orders that are late. Given the total number of orders, this measure can be translated into the percentage of orders that are late, a typical measure used in the industry. However, this measure does not indicate how late the orders are and using it to optimize schedules would create a preference for making an order one month late rather than making two orders one day late each.

It also ignores the size of the order and the importance of the customer. Given the option, it is probably better to make a 5 ton order late rather than a 50 ton order. Some mills also rank customers, with high volume customers getting higher priority for getting goods there on time. To account for these preferences, companies may weight the number of tardy orders by the number of days late, the size of the order and the importance of the customer. But an individual measure tends not to capture the desired objective. For example, it might be better to have one 50 ton order 1 day late than to have two 25 ton orders one day late. Both have the same number of ton-days late, but the second one disappoints two customers while the first one disappoints only one. The best utility function may use a combination of tardiness measures.

In creating scheduling systems, we have included industry standard measures of utility and have augmented them with measures used by the mills we have visited and with by measures that complete patterns. For example, we have encountered mills that use the number of high priority orders late, the number of ton-days late and the number of late orders. To this we have added the full complement of measures that weight late orders by the number of days, the weight, and the priority of the customer, in any combination. The full set of measures for each category are given in below. Within each category, the IBM Load Planning System allows the customer to aggregate these measures using a linear weighted sum. This simple utility representation has the advantage that it is easy to understand and tune. Give the formula, schedulers can understand the reasons that one schedule was ranked higher than another. This improves their understanding of how the system works and, consequently, their confidence in its operation. Finally, we recognize that even non-linear utility functions can be approximated by linear functions, over a small range of values that one sees in normal operations.

- **Time:**
 - ◇ Number of orders early or late.
 - ◇ Weighted by the number of days late or early.
 - ◇ Weighted by the weight of the order.
 - ◇ Weighted by the priority of the customer
 - ◇ Number of orders after the drop dead date, the last date the customer will accept delivery.
 - ◇ Weighted by the number of days late or early.
 - ◇ Weighted by the weight of the order.
 - ◇ Weighted by the priority of the customer
- **Quality:** Quality is a measure of product quality and perceived product quality.
 - ◇ Number of loads shipped by a mode other than that preferred by the customer
 - ◇ Number of loads shipped by a carrier other than that preferred by the customer
 - ◇ Number of loads shipped in a vehicle other than that preferred by the customer
 - ◇ Excess loads arriving at the customer dock per day.
 - ◇ Excess weight arriving at the customer dock per day.
 - ◇ Number of items in inventory for more than X days.
 - ◇ Weight of items in inventory for more than X days.
 - ◇ Number of loads that mix products
 - ◇ Number of loads that mix orders.
- **Money:** Factors which impact costs and revenues are included in this category, including:
 - ◇ Transportation cost.
 - ◇ Average transportation cost per ton

- ◇ Total drop-off charges
- ◇ Inventory carrying costs
- **Disruptions:** This category addresses operational concerns on the manufacturing floor, or at the loading docks, or while the goods are in transit such as:
 - ◇ Number of drop shipments
 - ◇ Number of transitions between the loading of different modes (i.e. Truck and train)
 - ◇ Number of loading dock capacity violations

Planning Approach

Rather than going directly from inputs to solutions, the IBM load planning system divides the problem up into three steps: group formation, load formation and load sequencing.

Group formation: Items are partitioned into groups of items that are going to the same, or similar destination, at about the same time and that can legally be loaded together into the same types of vehicles. This is the divide step of a divide and conquer approach. It reduces the problem size and eliminates attempts to create loads with items that can not be loaded together in any vehicle. The rules for partitioning are not rigid and improvement algorithms can move items between groups, coalesce groups and split groups when looking for better solutions.

Load formation: Items are assigned to vehicles in ways that create legal manifests. A complete solutions loads all items onto vehicles. One of the principle heuristics for achieving efficiency is to maximize vehicle utilization in order to reduce the number of vehicles needed, there by reducing costs, reducing the demand on loading dock capacity and making best use of the available vehicles. All of these help to achieve on time delivery as well. However, increasing vehicle utilization by combining orders going to similar destinations can lead to longer transit times and increased drop off charges that negatively impact both cost and timeliness. Combining orders also means that the vehicle can not leave until the last order is ready for shipment. Finally, there is a basic tradeoff between speed and cost. Shipping by rail tends to be less expensive, but slower. Trucks are faster and can go directly to more destinations, but tend to be more expensive. Delaying an order to combine it with another to achieve good vehicle utilization may require a change from rail to truck in order to meet delivery dates and result in a net increase in cost.

Load sequencing: Given a set of legal loads, load sequencing assigns these loads to specific docks and shifts, taking into account the dock capacity and vehicle availability. Each load is available to ship when the last item in the load has been produced. A simple heuristic for creating initial solutions is to assign each vehicle to the shift when it is first available for shipment. This may violate vehicle availability constraints and dock capacity constraints which force the load's ship date to be pushed out. Delaying a shipment for want of a vehicle or dock space may make the load late and require a change from train to truck in order to deliver the shipment on time.

Asynchronous Teams

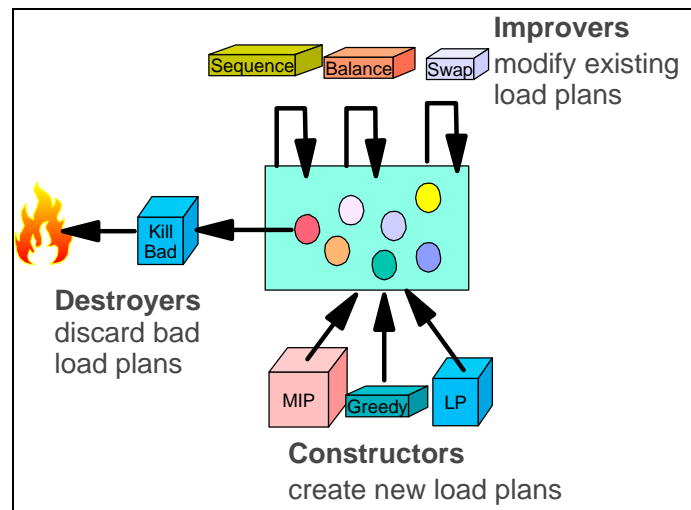


Figure 2. An A-Team is a software agent architecture that encapsulates different problem solving approaches as autonomous intelligent agents.

An Asynchronous Team or A-Team [TSM94], [Souza93], is an agent-based architecture well suited to multi-objective design problems. In this architecture, independent agents evolve a population of solutions towards the Pareto-Optimal frontier. As shown in Figure 1, agents (shown as blocks) communicate only through a population of candidate solutions. There are three basic roles played by agents:

1. **Constructors** create new solutions and add them to a population.
2. **Improvers** take a copy of one or more existing solutions, and modify them to produce a new solution which is then added to the population.
3. **Destroyers** keep the size of the design population in check by removing bad or redundant solutions.

The collection of agents runs as an anytime algorithm, with each agent embodying a different approach to improving the population of solutions. Often the approaches are heuristic. For example, constructors often use greedy algorithms that focus on a single objective. Using a set of greedy constructors, each of which focuses on a different objective, tends to seed the population with a diverse variety of solutions. From this beginning, improver agents can look for opportunities to hill climb solutions, along one or more dimensions, towards the Pareto-Optimal frontier. Adding domain specific heuristics, such as moving late orders to trucks from rail cars to reduce tardiness, allows the system to take advantage of knowledge to guide the search for better solutions. However, agents are not limited to running simple heuristic algorithms. In the load planning system, for example, we use linear and mixed-integer programming algorithms to generate initial solutions and to improve existing solutions.

The A-Team architecture does not define the content of the agents, but only their possible roles and the fact that they communicate solely through a shared population of solutions. This allows us to make use of the research in optimization and search which has focused on developing algorithms for solving particular problems. An A-Team can embody many such algorithms simultaneously. However, agents are more than simply function calls to iterative improvement algorithms. They decide for themselves *what* to work on, *when* to work, *how often* to work, and

in a parallel distributed environment, even *where* to work. The intelligence for controlling their individual behavior lies solely within the agent itself rather than some central controller

In addition to load planning, A-Teams have been used to solve nonlinear algebraic equations [TPM83], solve traveling salesman problems [TS92], configure task-specific robots [Murthy92], designing high-rise buildings [Quadrel91], control electric networks in real-time [TR93], and diagnosing faults in power systems [CT93]. A more detailed account of how A-Teams support cooperative approaches to scheduling may be found in [MRAW97]. [HLH97] demonstrates how combining multiple algorithms together produces solutions which are preferable to the solutions generated by individual algorithms operating alone,.

Interactive Scheduling

To illustrate the interactive nature of the IBM load planning system, we include two brief scenarios that give typical interactions between the scheduler and the system. In the first scenario, the scheduler uses the system to create a schedule. We show how the scheduler can inject extra knowledge to guide search and how the scheduler can modify the problem to allow for good solutions. In the second scenario, we show how the scheduling system can be used to answer what if questions and support negotiations.

Typically, schedulers begin each day by updating the load planning schedule to reflect changes in the production schedule, in orders, in loading dock capacity and in vehicle availability. The scheduler starts the load planning system and it loads the new information from the mill's logistics system. The existing schedule is automatically updated to remove orders that no-longer exist and is re-evaluated to show how well it solves the revised load planning problem. The scheduler can then have the system generate a set of alternative solutions. The existing solution is used to seed the A-Team population and the A-Team agents are run to generate a new set of solutions. The scheduler is presented with the set of non-dominated solutions along the Pareto-Optimal frontier. Suppose the scheduler decides to focus on two low cost solutions, and notices that one reduces tardiness by placing three small high priority orders in trucks. In order to reduce cost and increase vehicle utilization, the solution also fills each truck with part of another low priority order going to the same location. The scheduler notices that the destination for one of the high priority orders is on the way to the destination for another. He combines the two orders in one truck to form a drop shipment and discards the loading for the items used to fill the rest of the two trucks. He then re-runs the optimizer and it packs the unloaded items into partially filled rail cars. The third truck was scheduled to get an order to a customer by Friday. From the items in the order, the scheduler suspects that it is a inventory replenishment order used to maintain supply safety levels but not scheduled for immediate use. The priority is high only because it is going to a major customer. Sending the order by rail will get it there by Monday, three days later, but the next business day. The scheduler knows that the customer runs the plant 7 days a week, but only logs items into inventory on weekdays. The scheduler decides to call the customer to confirm this suspicion and ask if a delay is acceptable. The customer agrees and the scheduler changes the due date to Monday. Rerunning the optimizer produces a solution that efficiently packs all the items into rail cars.

In the above scenario, the scheduler has interjected knowledge to improve the solution and to change the problem. In many cases, the schedulers have years of experience and can recognize opportunities for improving a solution that our heuristic search methods might have missed. Sometimes these opportunities represent new heuristics that are applicable to a particular mill or customer but are not appropriate for inclusion in a generic load planning product. The interactive nature of the scheduling system allows the schedulers to make use of their knowledge without requiring changes to the scheduling software. This interaction also provides opportunities for the scheduling system to learn from the scheduler. Unfortunately, we have not had the time to take advantage of this opportunity yet. In addition, by highlighting the tradeoff between the two low cost solutions, the scheduling software identified a situation where the scheduler could negotiate with a customer to change the problem and reduce costs.

In the second scenario, the scheduler receives a call from a customer service representative who wants to know why a customer's order will arrive two days late and why it was not put on a truck to get it there sooner. Using the load planning system, the scheduler can add a constraint that forces the order to go by truck and re-run the optimizer. Simply moving the items to a truck would not give the impacts of the change on other parts of the schedule and would not allow for optimizations, like adding extra items to the truck in order to reduce overall cost. Suppose that the load planning system returns two solutions. The first solution gets the order there on time by shipping it today in a truck that is not well suited to the type of product. This truck was used because no trucks of the right type are available today. Using this type of truck risks damage to the product. The second solution ships the order tomorrow in a truck of the appropriate type and gets it there one day late. Sending the order by truck costs an extra \$1000. The scheduler can then use these solutions as a basis for informing and negotiating with the customer service representative. If the representative insists on shipping the product tomorrow to save a day, the customer service representative's name is recorded as authorizing the \$1000 expenditure. Providing facilities for track and authorizing decisions helps improve accountability.

Conclusions

In this paper, we have presented our approach to computer aided decision making, which recognizes that utility models are incomplete and relies on human operators to make the subtle distinctions necessary to resolve tradeoffs. We see decision making in organizations as a process of negotiation that computers can support by supplying relevant information and by suggesting good alternative solutions. Partial utility functions that evaluate a solution along a number of incomparable dimensions provide the information needed for guiding computer automated search, while giving solutions that highlight tradeoff for human schedulers. The use of a standard set of dimensions: time, quality, money and disruptions, and aggregating utility measures within each dimension using a linear weighted sum simplifies the elicitation procedure for deployment in real systems. Combining our incomplete utility model with an interactive A-Team based optimizer has resulted in robust scheduling systems that improve the efficiency and operating profits of our customers.

References

- [Biermann93] Biermann, C.; 1993. *Essentials of Pulping and Papermaking*, San Diego: Academic Press.
- [HLH97] Huberman, B., Lukose R., Hogg T.; 1997. An Economics Approach to Hard Computational Problems. *Science* **275** (pp 51-54).

- [Murthy92] Murthy S.; 1992. *Synergy in Cooperating Agents: Designing Manipulators from Task Specifications*, Ph.D. Dissertation, Carnegie Mellon University.
- [MRAW97] Murthy S., Rachlin J., Akkiraju R., and Wu F.; 1997. *Agent-based cooperative scheduling*, 1997 Workshop on Constraints and Agents (Technical Report WS-97-05). Menlo Park : AAAI Press.
- [Quadrel91] Quadrel R.; 1991. *Asynchronous Design Environment: Architecture and Behavior*, Ph.D. Dissertation, Carnegie Mellon University.
- [Souza93] Souza P de.; 1993. *Asynchronous Organizations for Multi-Algorithm Problems*, Ph.D. Dissertation, Carnegie Mellon University.
- [TPM83] Talukdar S.N., Pyo S.S., Mehrotra R.; 1983. *Distributed Processors for Numerically Intense Problems*, (Final Report for EPRI Project, RP 1764-3).
- [TR93] Talukdar S.N., Ramesh V.C.; 1993. *Cooperative Methods for Security Planning*, 4th International Conference on Expert Systems Application to Power Systems, Melbourne, Australia, Jan 4-8.
- [TS92] Talukdar S.N., Souza P.S. de; 1992. *Scale Efficient Organizations*, Proceedings of the 1992 IEEE International Conference on Systems, Man, and Cybernetics, Chicago, Illinois, Oct. 18-21.
- [TSM94] Talukdar S.N., Souza P. de, Murthy S.; 1993. *Organizations for Computer-Based Agents*, Engineering Intelligent Systems, (1:2).