ANALYSIS OF ISSUES FOR PROJECT SCHEDULING BY MULTIPLE, DISPERSED SCHEDULERS (DISTRIBUTED SCHEDULING) AND REQUIREMENTS FOR MANUAL PROTOCOLS AND COMPUTER-BASED SUPPORT

Final Report

NASA/ASEE Summer Faculty Fellowship Program--1991

Johnson Space Center

Prepared By:	Stephen F. Richards, Ph.D.
Academic Rank:	Associate Professor
University & Department	Ambassador College Computer Information Systems Department Big Sandy, Texas 75755
NASA/JSC	
Directorate:	Information Systems
Division:	Information Technologies
Branch:	Software Technology
JSC Colleague:	Chris Culbert
Date Submitted:	November 18, 1991
Contract Number:	NGT-44-001-800

ABSTRACT

Although computerized operations have significant gains realized in many areas, one area, scheduling, has enjoyed few benefits from automation. The traditional methods of industrial engineering and operations research have not proven robust enough to handle the complexities associated with the scheduling of realistic problems. To address this need, NASA has developed COMPASS (COMPuter Aided Scheduling System), a sophisticated, interactive scheduling tool that is in wide-spread use within NASA and the contractor community. Like most existing tools, however, COMPASS addresses only single- user applications. COMPASS therefore provides no explicit support for the large class of problems in which several people, perhaps at various locations, build separate schedules that share a common pool of resources.

This research examines the issue of distributed scheduling, as applied to application domains characterized by the partial ordering of tasks, limited resources, and time restrictions. The focus of this research is on identifying issues related to distributed scheduling, locating applicable problem domains within NASA, and suggesting areas for on-going research. The issues that this research identifies are goals, rescheduling requirements, database support, the need for communication and coordination among individual schedulers, the potential for expert system support for scheduling, and the possibility of integrating artificially intelligent schedulers into a network of human schedulers.

INTRODUCTION

Scheduling is the process of assigning resources and times to each activity of a plan (or plans) while ensuring that each constraint is obeyed. Optimization criteria can determine the relative desirability of two alternate schedules. Although scheduling problems are often simple to visualize and express, scheduling is an NP-complete problem, so attempts to apply mathematical programming to scheduling have met with very limited success [2, 6]. In fact, programming approaches have been limited to very narrow problem domains, especially that of the job-shop, in which jobs must be assigned to various machines.

This research focuses on the class of scheduling problem in which:

1) activities have precedence relationships (one activity must not begin until another activity has completed);
2) resources are limited;
3) objectives or optimization criteria exist that may be used to rank competing schedules; and
4) the time frame in which to complete all activities (or as many activities as possible) is limited.

This class of problem differs from the job-shop problem domain in that a job-shop problem assumes an infinite time line in which all activities may complete. In a job-shop problem, all activities are scheduled regardless of the total time required. In contrast, in this research resources may be over-subscribed, so that even the optimum schedule might not accommodate all desired activities within the time limitations. Thus, provision must be made for selecting between competing activities (or sets of related activities) where insufficient time exists for the completion of all activities.

To assist users in developing viable schedules NASA has developed COMPASS (COMPuter Aided Scheduling System) [3], a computer-based tool that interactively schedules activities in a user-specified order. COMPASS provides graphical tools for displaying activities, resource availability, and schedules. In COMPASS, activities are partially ordered and require resources. Activity attributes supported by COMPASS include priority, required resources, duration, earliest permissible start time, latest permissible end time, and state conditions. COMPASS has enjoyed wide-spread acceptance within NASA and the contractor community.

NASA has recently proposed enhancing COMPASS to support multi-user or distributed scheduling problems. This research has focused on the issues raised by distributed scheduling and on requirements for computerized support of this problem domain. The next section of this report defines distributed scheduling and addresses these issues.

DISTRIBUTED SCHEDULING

Distributed scheduling consists of those scheduling problems involving several schedulers, each of whom is responsible for scheduling activities that are somehow interrelated. Typically, the activities will share a common resource pool, but the activities also may have precedence requirements, or one activity may establish a state that another activity requires, etc. Distributed scheduling problem domains of particular interest to NASA include the scheduling of astronomical satellite experiments, personnel training, and space mission activities.

In such problems, the size and complexity of the scheduling task and the limited abilities, skills, knowledge, and resources of any individual make the distribution of the

scheduling task a natural and necessary means of developing the required schedule. By distributing the work, each scheduler can concentrate on a manageable volume of work in a narrow domain. However, the lack of a single point of control increases the complexity of the overall scheduling problem (as a result of the necessary communication overhead) and raises several issues regarding the interactions of multiple schedulers and the integration of their individual schedules.

The most basic issue is that of goals. What measure of goodness is most appropriate in a distributed environment? How do the optimization criteria for a distributed scheduling problem differ from those for a non-distributed problem? Variables commonly used for scheduling problems include [4, 5]:

-- Completion time: the time at which processing of the last activity completes.
-- Flow-time: the total time that activities spend in the shop.
-- Lateness: the difference between the completion time of an activity and some pre-specified due date associated with that activity.
-- Tardiness: equal to lateness when lateness is positive, otherwise equal to zero.

Schedule evaluation criteria typically involve minimizing or maximizing the mean, total, minimum, or maximum or one or more of these variables. In a standard job-shop problem, these criteria are assumed to be universally agreed upon. However, even in such a standard, non-distributed scheduling environment, the various tasks to be scheduled may belong to several different customers (perhaps represented by members of the marketing staff), each of whom would prefer that his or her tasks be given high priority. Thus, even in a non-distributed setting conflicting goals may exist. When conflict exists, the scheduler must have some means of determining a set of priorities to be applied to the scheduling task. The scheduler may be flexible in his or her choice of priorities, adjusting them to the needs of the moment. For example, the scheduler might attempt to mollify a major customer who has been slighted by giving preference to that customer's work. Regardless of the conflicting demands, however, the optimization requirements are formulated as a single point of control and this procedure can succeed because the single scheduler (or team of schedulers) who develops the optimization criteria also controls the entire resource pool.

In a distributed schedule, however, individual schedulers must share resources, so one scheduler optimizing his or her schedule may restrict another scheduler's options, resulting in a suboptimal global schedule. The issue of a global measure of goodness becomes more important in distributed scheduling than in individual, interactive scheduling. This is true because an individual scheduler can accept a schedule even without a specific measure of goodness; the schedule may balance several conflicting needs fairly well and "just look good." A distributed schedule, in contrast, must "look good" through several sets of eyes. When a team of schedulers must continue to work together on future projects, perceptions of inequity or misplaced priorities can engender resentments that will poison these on-going relationships. Thus, some mechanism for balancing both local and global optimization must be provided. The protocol used by the schedulers to coordinate their activities must support optimization techniques that are perceived as both equitable and efficient.

Selecting a desirable scheduling protocol requires balancing several possibly conflicting goals, including the following [1]:

1. The protocol should encourage the development of high quality schedules that score well when evaluated by either the global optimization criteria or the optimization criteria of individual schedulers. Schedules should also be resilient to unexpected changes.
2. The protocol should be easy to use. Features enhancing ease of use include ease of

learning; minimum complexity; informative to the user of the state of activities, resources, etc.; and natural representation or concepts. Yet, the process should be sufficiently rich in features and notation to encompass a wide range of scheduling problems.

3. The overhead, such as communications requirements, should be kept to a minimum.
4. The time required to develop schedules should be short, especially in highly dynamic environments.
5. Any computerized support should have a short response time. This requires that optimization techniques be computationally simple.

Several sample scheduling protocols are listed below. For each alternative the issues of division of resources, communication, cooperation, and optimality are discussed.

1. Schedule tasks by priority. This protocol requires that all tasks be known and prioritized in advance and then be scheduled in priority sequence. This is really non-distributed scheduling, except that we have several schedulers responsible for collecting tasks and we may provide improved computer support to enable the individual schedulers to track their own set of tasks by viewing only their portion of the schedule. This protocol also requires some mechanism for assigning priorities to tasks (e.g., a central authority or a voting scheme). The potential for high quality projects is the same as for non-distributed scheduling. Although an advantage of this method is that participants will perceive it as fair, following this method strictly does not allow for compromises, such as scheduling two medium priority, low resource intensive tasks instead of one high priority, high resource intensive task.
2. First come, first served. In this approach all schedulers are equal and none has priority over the others. Resources are not assigned to individual schedulers, but may be reserved by any scheduler. The state of the global schedule must be continuously available to all schedulers. No cooperation among schedulers is required. Optimization is poor, because there is no attempt to balance the needs of multiple schedulers. There is a tendency among schedulers to reserve resources early, even before they know their full requirements. This tendency to hoard can result in the allocation of resources to low priority tasks.
3. Divide resources among schedulers in advance. This protocol permanently allocates resources to specific schedulers who can use them as they choose. No communication or cooperation among schedulers is required. Schedulers need not even know the global schedule. This approach is impractical when there is a potential state conflict between tasks (e.g., when two schedulers independently schedule a treadmill experiment and aa microgravity experiment that requires no vibration). This approach may also yield poor schedules when one scheduler assigns resources to low priority tasks or leaves resources unused that could be used by another scheduler. In this approach the quality of the resulting schedule is limited by the quality of the initial allocation of resources.
4. Divide resources among schedulers in advance but permit borrowing. This approach differs from the previous one by permitting schedulers to negotiate among themselves to improve their schedules. There is still no need for global optimization criteria. The status and bargaining power of individual schedulers is determined by the initial allocation of resources. Communication needs consist of a knowledge of resources available to other users.
5. Sharing of intentions among schedulers. In this protocol schedulers review their intentions with their peers and receive feedback before reserving resources and

committing to a particular schedule. While this approach has the potential for producing high quality schedules through the sharing of knowledge and expertise, it also imposes a heavy communication burden among schedulers that can negate much of the benefit resulting from distributing the scheduling task. This approach is also fragile in that its success depends on the voluntary cooperation of each scheduler. Where this cooperation fails, this protocol can degenerate into a first come, first served system.

6. Simultaneous iterative scheduling. In this protocol each scheduler devises a schedule and shares it with others. Schedulers identify and resolve conflicts by some agreed upon method. If unscheduled tasks and unallocated resources remain, another round of scheduling follows. In this approach all schedulers must be ready to schedule simultaneously. Also, each participant must be provided some incentive to cooperate with the others in resolving conflicts. The global schedule must be available to all schedulers.
7. Consecutive iterative scheduling. In this protocol the schedulers are divided into two or more groups that alternately devise schedules. This approach is useful when one group creates resources required by another. For example, a university administration develops a schedule of classes, the students then submit their individual schedule requests, and the administration, after analyzing the requests, adds sections to some classes and deletes sections from others. The students then request changes to their schedules. In principle this cycle can continue for many iterations. This approach requires some incentive to cooperate and requires that each scheduler knows the global schedule and the state of available resources.

Any attempt to develop a universal scheduling methodology is doomed to failure because of the enormous diversity of scheduling domains. The variety of tasks, resources, constraints, and environments is virtually unlimited. The protocols listed above are not applicable to all domains but must be selected based on the characteristics of the specific domain of interest.

A second issue identified by this research is the requirement for revising a schedule, also termed rescheduling [2]. Several factors can trigger a need to reschedule. A resource can become unavailable, making the current schedule unfeasible; a task can take longer than expected; or a user can change his or her requirements so as to impose a conflict, exhaust a resource needed by a later task, or delete an enabling task that creates a resource or state needed subsequently. In addition, rescheduling is desirable, although not required, whenever an opportunity arises to improve the schedule by adding previously unscheduled tasks or resequencing already scheduled tasks. This can happen, for example, when new resources become available or when a task completes early. Differences between scheduling and rescheduling include: 1) rescheduling takes place in the context of an existing schedule that we may wish to disturb as little as possible; 2) rescheduling must consider work in progress; 3) rescheduling often must occur quickly, in contrast to the initial scheduling which may be performed in a more leisurely manner; and 4) someone other than the original scheduler may perform the rescheduling. An important issue for rescheduling in a distributed scheduling environment is the need to reduce communication requirements among schedulers to facilitate quick rescheduling. Since this may require a return to centralized scheduling, the rescheduler must have the appropriate information to make beneficial changes.

A third issue is that of database support for distributed scheduling. A distributed scheduling system requires many of the features of a distributed database management system. The system must merge separate databases of tasks, resources, constraints, and

assignments into a single image while retaining the ability to display for individual schedulers only those portions of the database under their control. However, since each scheduler has a different view of the world (with different time scales, measures of goodness, types of constraints, etc.), the system must support different user languages and communicate with each scheduler in a natural and helpful way. As our software tools, such as COMPASS, address more diverse and complex problem domains, we will require a more comprehensive database language for describing scheduling problems.

A fourth issue is that of communication and coordination among schedulers. At present, many of the NASA schedulers who impact one another by their work do not even know one another, much less communicate regularly. The people who schedule the Hubble telescope and its ground facilities are assigned specific resources that they schedule independently of each other. One research question concerns how much of this lack of communication is the result of historical developments and how much is intentional on the part of schedulers. Do they fail to communicate because they do not perceive a need to communicate, or because they feel communication is too time consuming, or because they fear loss of control of their environment, or is there some other reason? If independent scheduling is a human preferred approach, then it will be important to determine why this is true, how we can encourage people to cooperate, and how we can enhance cooperation while reducing communication. The mechanisms for communication and coordination (the languages, database support, and interaction procedures) appear to be a critical aspect of distributed scheduling by human agents.

A fifth issue involves the introduction of expert system support for scheduling. Optimization heuristics have been envisioned for individual scheduling support; some of this support is already available on COMPASS. Support for distributed scheduling would focus on communication and negotiation. An expert system that monitored the actions of all schedulers could infer when one scheduler needed to know of the actions of another. This would reduce communications requirements. Also, an expert system could search for instances in which it would be mutually advantageous for two schedulers to trade resources or to reschedule certain tasks.

Finally, a sixth issue asks how we can introduce artificially intelligent schedulers into the system. The scheduling of certain domains, such as power generation, may be suitable for AI approaches. It would be natural to introduce such agents into human scheduling systems. How can artificial and human agents best interact? This issue awaits the development of suitable AI schedulers for individual scheduling domains.

CONCLUSIONS

This research has begun an investigation of the issues of distributed scheduling. This report has identified and discussed several issues. These issues will impact both the computerized support for distributed scheduling that is envisioned to appear in future versions of COMPASS as well as the manual procedures that schedulers will use for cooperating with one another.

During this academic year this researcher will remain in contact with NASA to identify members of the NASA community who might benefit from distributed scheduling and who are willing to participate in experiments during the coming summer. During the summer we will investigate the effects of different optimization criteria and procedures for communication and coordination.

REFERENCES

1. Fox, Mark S., "Constraint-Directed Search: A Case Study of Job Shop Scheduling," Ph.D. dissertation, Carnegie-Mellon University, 1983.

2. Fox, Mark S. and Zweben, Monte, "Knowledge Based Scheduling," Tutorial MA2, presented at the Ninth National Conference on Artificial Intelligence, July 15, 1991.

3. McDonnell Douglas Space Systems Co., "COMPASS 2.0 User's Manual", for NASA/Johnson Space Center Software Technology Branch, 1991.

4.. Ow, Peng Si, "Heuristic Knowledge and Search for Scheduling," Ph.D. dissertation, Carnegie-Mellon University, 1984.

5. Salvador, Michael S., "Scheduling and Sequencing," in HANDBOOK OF OPERATIONS RESEARCH, Joseph J. Moder and Salah E. Elmaghraby (Editors), Van Nostrand Reinhold, New York, 1978, pp. 268-300.

6. Ullman, J. D. "NP-Complete Scheduling Problems," Journal of Computer and System Sciences, Vol. 10, 1975, pp. 384-393.