The Scheduling of Rail at Union Pacific Railroad

Kathleen Murphy, Elizabeth Ralston, David Friedlander - Brightware, Inc.
Rodney Swab, Paul Steege - Union Pacific Railroad

Brightware Incorporated
350 Ignacio Boulevard
Novato, California 94949
kmurphy@brightware.com, ralston@brightware.com, friedlander@brightware.com

Union Pacific Railroad
1416 Dodge Street
Omaha, Nebraska 68179
reswab@notes.up.com, Paul_Steege@notes.up.com

Abstract

The Union Pacific Railroad (UPRR) has over 31,000 miles of track covering a 24 state region. Planning and scheduling the production, packaging, delivery, and pickup of rail, involved in the maintenance of this network, is a very complex task. Manually scheduling only a subset of the resources required has historically taken several days to accomplish. Moreover, the inability to fully schedule all resources can lead to inefficient resource utilization. This paper describes the Rail Train Scheduler (RTS), designed and developed to capture the expertise of the UPRR scheduler, generate production schedules of all the resources involved, and provide a decision support tool for determining the best mix of resources required. RTS is an expert system that uses constraint satisfaction and domain specific heuristics to produce good, low cost schedules. It has been deployed since January, 1996. UPRR anticipates a savings of about $500,000 per year from the use of RTS.

Introduction

This paper describes a scheduling application, the Rail Train Scheduler (RTS), designed and developed for Union Pacific Railroad (UPRR). The primary motivation for developing the application was to solve a major operational problem relating to the allocation of resources required to provide rail to Union Pacific track construction and repair projects. The current manual process schedules some of the resources (weld plant capacity and rail material) but just assumes adequate availability of the necessary rail handling equipment. It also focuses on the delivery of rail while generally ignoring the pickup of used rail for re-use. It is estimated that a more efficient use of available resources could save UPRR about $500,000 per year.

The solution to this problem was to build an expert system that captures the expertise of the rail scheduling expert, additionally incorporates the scheduling of rail trains and power units and the pickup of used rail, and provides full "what if" capabilities for the user to test various resource mixes. However, the production of these schedules is a very difficult problem, intractable using traditional techniques. The problem was solved using a unique combination of constraint satisfaction techniques and domain specific heuristics. In this way RTS creates 'good fit' schedules automatically in much the same way that the rail train expert does but does so in minutes, rather than days. It also incorporates a more complete scheduling of relevant resources.

As implemented, RTS produces high quality, low cost schedules of all the resources required for the production and delivery of rail to the construction sites and the pickup of used rail from the field. It can also be used as a "what if" decision support tool, allowing the user to experiment with differing amounts of resources to see which combinations make for good schedules.

The Domain

The Union Pacific Railroad has over 31,000 miles of track covering a 24 state region. Every year Union Pacific has a number of track maintenance and construction projects that involve the laying of rail. While most of these projects are planned and scheduled well in advance and spread throughout the year, emergency needs may also arise at any time, e.g. to repair track damaged by floods, derailments, etc. The rail is laid in ¼ mile long segments (strings), which are welded from shorter pieces at dedicated rail weld plants. The rail is then delivered to the construction sites by special rail trains, ideally arriving shortly before a work crew is scheduled to begin laying the rail. Once at the site, the rail is unloaded using special unloading equipment. After a rail train has delivered all of its rail, it can be used to pick up old rail, removed during construction, and return it to the weld plant. This activity requires pickup equipment.

A single UPRR rail scheduling expert schedules the welding of the rail manually, based on the location and delivery dates of the rail requirements, rail inventory, and weld plant capacity. The UPRR maintains enough rail handling equipment (rail trains and unload and pickup units) that this equipment is not scheduled in advance. The supply is assumed to be adequate for any schedules produced. Pickups of rail in the field, for return to the weld plant and re-use, are generally not scheduled. The scheduling expert produces an initial schedule for all rail projects planned for the year and then produces revised schedules on at least a weekly basis, taking into account changing conditions, emergency requirements, etc.

The production and delivery of this welded rail requires several resources, as mentioned above: weld plant capacity, rail inventory, rail trains, and unload and pickup units (referred to collectively as power units). Each track construction project is assigned to a specific weld plant, which will supply all the welded rail required for that project. Raw material for the welded rail, specific rail trains, and specific power units are also all dedicated to a particular weld plant. This allows the production and delivery of rail from a single weld plant to be scheduled independently of any other plants, although resources and/or requirements may be manually shifted from one weld plant to another to remedy resource imbalances. UPRR currently has two weld plants, although it may also lease additional plants as necessary. Also, a certain amount of weld plant overtime can be scheduled to meet welded rail requirements. The rail raw material is welded into ¼ mile strings and fed directly on to a rail train at the weld plant. The rail must be welded and loaded in the reverse order that it is to be delivered and unloaded.

Track construction projects may have requirements for more than one type and/or weight of rail. For example, standard rail material is used on straight stretches of track, while head hardened rail is used for curves. The head hardened rail is much more expensive and has a purchasing lead time that is twice as long as that of standard rail. The quantities of welded rail required vary widely both by project and, within a project, by rail type and weight. Rail for one project may fill several rail trains, while in other cases the requirements for several projects will all fit on to a single train. Rail train capacity varies from train to train. Currently, trains have anywhere from 40 to 54 pockets, where each pocket holds a single string. Power units travel only on rail trains, so the required equipment must be available at the weld plant before a loaded train can leave.

In the manual scheduling process, the rail scheduling expert groups rail requirements into train loads, based on both geographical and temporal proximity of the projects. He orders the weld requirements in each load and determines when each load is to be welded. He schedules plant overtime as necessary to meet the project requirements. He also considers rail material availability in producing the schedule. He does not schedule the rail trains or power units beyond specifying the train to be used for the next load to be welded.

A good rail train schedule is defined by a set of prioritized goals (ordered from highest to lowest priority):

- Deliver all welded rail requirements on time.
- Maintain uninterrupted weld line operation/ minimize overtime.
- Minimize the total number of miles traveled to deliver the rail.
- Pickup up rail in the field and bring it to the weld plant.
- Free up excess trains/power units for leasing.

In general, the goal of minimizing distance is achieved by minimizing the number of loads required. Thus the scheduling expert strives to create full loads. He analyzes the geography of rail requirements and groups requirements into loads such that the projects to which the rail is to be delivered are close together or lie along a single path from the rail plant.

UPRR would like to reduce the costs associated with providing rail to the construction projects by minimizing the resources required to support a schedule. In other words, they would like to reduce the amount of rail inventory on hand at any given time, to have no more weld plants (or weld plant shifts) in operation than are necessary to support current rail needs, to have an adequate, but not excessive, supply of rail handling equipment that is used efficiently, and to maximize the amount of used rail that is picked up and returned to the weld plants for reuse. However, determining the correct mix of resources required to produce a good, low cost schedule is a difficult problem, as will be discussed in the next section.

The Operational Problem

Resources - weld plant capacity, rail inventory, trains, and power equipment - interact in complex ways which affect the quality and cost of the schedules that can be produced. This section discusses the important tradeoffs and interactions of the rail train scheduling process.

Typically what happens when resources are not reasonably balanced is that it becomes difficult to group requirements into full loads. Once there are small loads, the schedule requires more rail trains to deliver the rail. When more trains are required, more power units are needed as well. If it turns out that there are not enough trains in the system to support the schedule, then the weld line must be shut down to wait for a train to return to the weld plant to be loaded. This interruption of welding may then necessitate overtime later to meet required delivery dates. The result may be a high cost cycle of weld line shutdowns alternating with overtime to minimize late deliveries. In this type of cycle late deliveries are likely to occur. Moreover, when there is a shortage of rail trains, pickups of rail in the field for delivery to the weld plants can not be scheduled without intensifying the problem. Figure 1 illustrates this high cost situation, with the solid arrows indicating definite consequences and the dashed arrows indicating possible consequences.

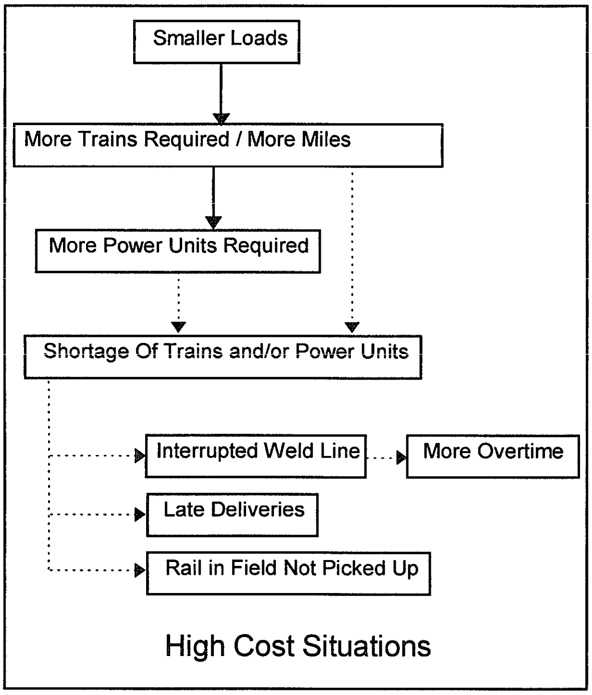

Figure 1

Several different sets of circumstances can lead to the situation just described. In order for requirements to be grouped into full loads, some requirements must be welded earlier than is necessary for on time delivery. If rail inventory is tightly constrained, it won't support rail being welded early. If the weld plant runs out of one type of rail, requirements for other types must be welded early to avoid shutting down the weld line. Projects with requirements for both the type of rail that is out of stock and other rail that is available are then likely to have those requirements delivered on separate trains, rather than grouped on a single load.

Another set of circumstances leading to small loads involves weld plant capacity. When a weld plant is running at close to total capacity (including all available overtime hours), there is not much flexibility available to schedule requirements early to maximize full loads. This is because the highest priority is put on timely delivery. A requirement will not be scheduled to be welded early if that will cause another requirement to be welded and delivered to a project late. If capacity is constrained the schedule may require a larger number of smaller loads, which is a costly situation.

A final example of what happens when resources are not in balance involves the relative number of trains and unload units available at a weld plant. A shortage of unload units will increase the number of rail trains required to support the schedule. This is because trains will be required to remain at the weld plant after they have been loaded with welded rail, waiting for an unload unit to return with another train. If the shortage of unload units is great enough, this situation can result in an actual shortage of rail trains, with the consequences described above.

Clearly the best way to minimize costs is not to remove all constraints on resources. Rather, these constraints need to be well balanced. This balancing is a complex and delicate process. If all of the costs, benefits, and interactions between resources involved in the production and delivery of rail could be analyzed and explicitly modeled, and known at the time a schedule was produced, then the quality of a single schedule could be quickly calculated. In this case it would be relatively easy to assure the appropriate balance of resources to produce an optimum schedule. However, no such comprehensive, explicit model exists.

In fact, there are a number of factors which affect the quality and cost of a schedule but are clearly difficult to quantify. One of these is the cost of late delivery of rail. Delivering rail late to a project can cause disruptions in the construction schedules, which certainly is a cost to the railroad as a whole. There is no model for this cost of late deliveries, but the scheduling expert (and the RTS software) attempts to avoid late deliveries if at all possible, incurring other costs (increased travel and/or overtime) if necessary to avoid them.

Another factor affecting the cost of a schedule is the value of picking up used rail in the field and returning it to the weld plants for re-use. There are clearly costs and benefits involved in picking up this rail. If two schedules are otherwise the same, the one which includes more rail pickups is presumably better. This benefit has not been quantified.

Because there is no explicit cost model for either producing an optimum mix of resources or for determining the relative costs of various schedules, the only way to effectively assess the consequences of different mixes of resources is to produce schedules based on different combinations of resources, evaluating how fully the available resources are used and how well the resulting schedules meet the prioritized schedule goals. However, the manual process of scheduling the rail is very time consuming and includes neither the scheduling of rail trains and power equipment nor the pickup of rail in the field. Clearly, a fast, accurate and consistent method for the complete scheduling of all resources is highly desirable.

The Operational Solution

The development of the Rail Train Scheduler was motivated by a desire to have a method for quickly and consistently producing both real production schedules that make good use of all available resources and hypothetical schedules that show the exact effects of adding or taking away resources. The Rail Train Scheduler (RTS) automates the full scheduling of rail production and delivery, including the scheduling of rail trains and power equipment. It also has features that allow the user to modify the availability of resources and immediately view the resulting schedule and its resource utilization.

RTS has a fully developed "what if" capability. Both resources and requirements can be modified, added, deleted, or assigned to a different weld plant. Figure 2 shows the window provided for definition of new trains and modification of parameters of previously defined trains. Rail inventory quantities can be changed. Open purchase orders can be modified, created, or deleted. Usage of weld plants, rail trains, and power equipment can be restricted to specified periods of time (Figure 3 shows the window provided to do this).

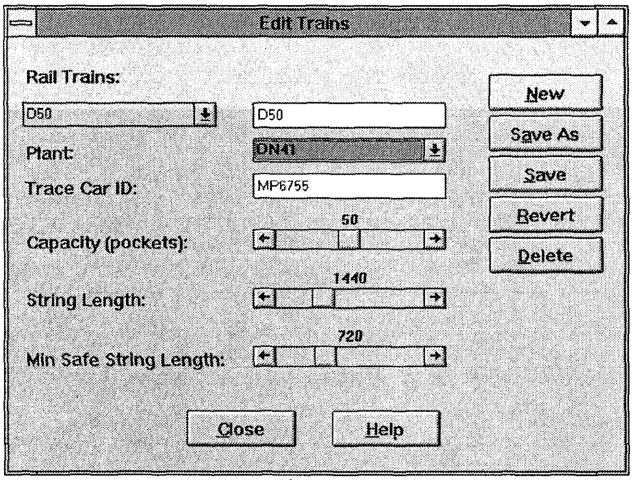

Figure 2

In addition, parameter settings which control the relative priorities of the goals of the scheduler can also be changed by the user. For example, it may be decided that even if overtime is not required to meet delivery dates, it may sometimes be desirable to add overtime and weld early in order to create full loads. The relative priorities of maximizing full loads and avoiding unnecessary overtime can be adjusted in RTS by changing the value of a parameter that specifies that a requirement can be welded N number of days early to create a full load, even if this results in overtime being required to meet the delivery dates of one or more subsequent loads.

Once RTS has produced a schedule, the user can see the resources it uses. The user can examine a running inventory of each type of rail for each weld plant, the overtime schedule for each plant, and rail train utilization (Figure 4).

In designing and developing RTS, there was no technical problem involved in providing the user with these capabilities. However, developing an appropriate algorithm for producing good schedules was a significant technical challenge. This technical problem is described in the next section.

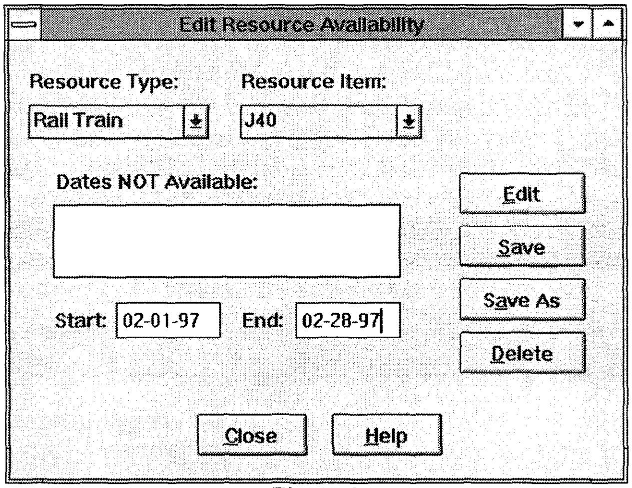

Figure 3

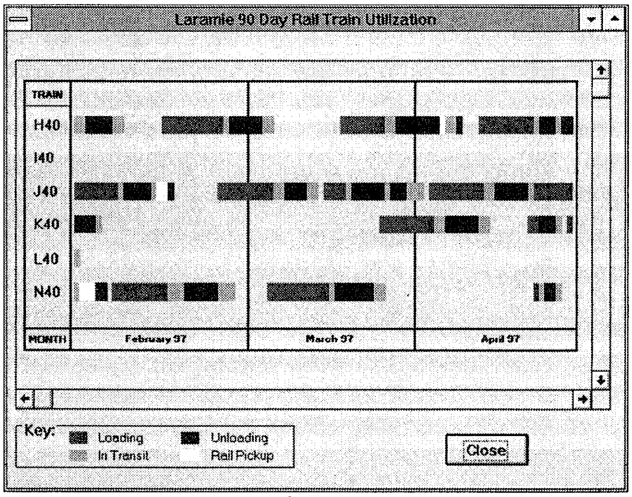

Figure 4

The Technical Problem

The scheduling problem central to the RTS application is a complex problem that contains elements of two known NP-Complete problems, the "Traveling Salesman Problem" and the "Bin Packing Problem". The scheduling problem displays many of the characteristic features of any NP-Complete optimization problem:

1. There is no specific procedure to follow to create the least cost schedule.
2. The number of possible schedules increases exponentially with the number of variables involved in schedule generation. The variables involved include different combinations of the following:
   - What type and quantity of rail, for which project(s), to include in each load.
   - How to order the rail projects within each load
   - What order to schedule the loads in.
   - Which train and power unit(s) to assign to each load
   - How much overtime to schedule for each weld plant and when to schedule it.
   - If and when to assign pickup(s) of rail for the weld plant to a load and how much of which rail to pick up.
3. There are no criteria for recognizing the best schedule in isolation from all other possible schedules.

Because of the exponential increase in the number of possible schedules as the number of variables to be taken into account increases, it's not possible to generate and evaluate all possible schedules. There is no feasible way to consistently generate and recognize the "best" schedule. One has to settle for a very good schedule. Moreover, unlike the classic NP-complete problem, there is no explicit cost model available to determine the cost of a schedule. This makes comparing the relative quality of various schedules a complex and subjective process, involving an assessment of how well the prioritized goals are met and how fully and efficiently resources are utilized.

Alternative Approaches to Solving the Technical Problem

A number of different techniques have been developed in operations research and artificial intelligence to find good (although not optimal) solutions to NP-Complete type problems in a reasonable amount of time. Among these are:

- Operations Research modeling
- Genetic Algorithms
- Artificial Intelligence heuristic techniques
  - General purpose
  - Domain specific

Operations Research Modeling

The operations research approach models very specific types of problems. A typical approach is to use a linear programming model. Each model consists of a set of linear equations based on the constraints, costs, and benefits of that particular type of problem. A transshipment model was initially considered to be a candidate for use in developing RTS. However, further analysis showed two fundamental problems. First, while many of the characteristics of the rail train scheduling problem fit the model, there were additional constraints and characteristics that couldn't be accounted for by this approach. Second, the key to the operations research approach is to develop a complete set of equations modeling all of the relevant cost factors to derive the least cost schedule. Since many of the actual costs and benefits are not readily available or quantifiable and some cannot be known until after the fact, this approach proved to be unworkable.

Genetic Algorithms

One AI approach that was considered briefly for RTS was the use of genetic algorithms. This technique is based on genetics and natural selection as they occur in the physical world. A large population of potential solutions is initially created randomly. Then a fitness function is applied to each member of the population to select a subset of the best solutions to be used to create the next generation of potential solutions. Eventually, over many generations, one or more very good solutions will evolve. This technique has produced good results with the Traveling Salesman Problem. However, the success of its application is dependent upon two conditions. First, candidate solutions need to be represented concisely as vectors of symbols or numbers. Second, there must be an accurate and efficient fitness function to determine the relative "goodness" of each potential solution. Neither of these conditions is readily met for the production and scheduling of rail, so the approach was not pursued.

Artificial Intelligence Heuristic Techniques

A number of AI heuristic search techniques have been developed for use when searching large numbers of possible solutions for a good one. There are two basic approaches:

- Techniques that will frequently, but not always, find the best solution quickly
- Techniques that always find a good solution in a reasonable amount of time

What these techniques all have in common is that they improve the efficiency of the search at the expense of completeness. They do this by pruning out large chunks of the search space.

One of the most effective general purpose techniques, that combines elements of both basic approaches, is the A* search algorithm. Like the other approaches mentioned, this algorithm was rejected because of the absence of a good cost function.

On the other hand, another general purpose technique, constraint satisfaction, was found to be very useful in implementing RTS. With this technique, large numbers of potential solutions are either rejected or never generated because they fail to satisfy one or more of the constraints of the problem. RTS also makes heavy use of domain specific heuristics to find good schedules. The next section describes how this was done.

The Technical Solution

As indicated above, the approach taken in implementing RTS combines both constraint satisfaction and domain specific heuristics. In general, the overall approach can be described as follows:

- Satisfy all scheduling constraints.
- Break the scheduling process into independent components that can be performed sequentially.
- Within these components, use domain specific heuristics in much the same way as the domain expert to guide decisions to produce a single good schedule, rather than generating and comparing large numbers of alternative schedules.

RTS is designed to satisfy a number of constraints in producing a schedule. First, the scheduler must schedule all requirements for welded rail and all pickups of rail to be delivered to another project or to a different weld plant. Second, it must satisfy constraints relating to the availability of resources. In order for a welded rail requirement to be scheduled, there must be sufficient inventory of the specified type and weight available when welding is projected to begin. There must also be sufficient capacity at the weld plant to weld the required rail as scheduled. Trains and power units can be assigned to loads only if they are available, based on estimates of the time required to load and unload rail and to travel. The procedures for generating the schedule are designed to avoid producing a schedule that violates these constraints.

Within the constraints described above, the scheduler uses domain-specific heuristics to produce a schedule that maximizes timely delivery of requirements, while promoting level scheduling of hours at the weld plants. It also attempts to minimize the number of trains which are used to deliver rail and to minimize the total distance traveled by these trains. Most of the steps involved in producing a schedule are to some degree interdependent: rail availability depends on the order in which rail requirements are welded and the weld plant schedule; the ordering and grouping of requirements into loads depends on rail inventory, plant capacity, and train assignment; train assignment depends weld plant schedule and the composition of the loads; etc. However, it was determined that the process could be broken into several independent, sequential steps and still produce a good, if not necessarily optimal schedule. These steps are:

1. Check for inventory and plant capacity shortfalls and generate a tentative weld plant schedule - that is, hours per week that the welding line will operate.
2. Create a geographic network of the weld plant and all rail delivery and pickup locations.
3. Construct rail delivery train loads based on geographic and temporal proximity.
4. Order the loads and assign weld completion dates, rail trains, and power units.
5. Add rail pickups to the load schedule when possible.

It is beyond the scope of this paper to describe fully the heuristics employed. The remainder of this section describes the primary heuristics related to the building of loads in a single procedural pass.

It is not possible to handle the weld plant capacity and rail inventory constraints completely independent of the other aspects of the scheduling. However, it is possible deal with these constraints in three distinct steps. Before any loads are created or trains assigned, the program checks projected inventory (based on purchase orders) and determines if there are any requirements that cannot be welded soon enough for on time delivery due to inventory shortfalls. If so, these requirements are assigned an earliest weld date, based on inventory availability, that is used in both the scheduling of overtime and the construction of loads. RTS then calculates the total number of welding hours necessary to produce the required rail for the project year, and produces an overtime schedule for the weld plant. The goal is to schedule overtime so that all rail can be welded and delivered on time, while running the weld plant at least 40 hours each week. The algorithm adds any required overtime as needed to meet delivery requirements. Then, during load construction, whenever adding a requirement to a load will necessitate the requirement being welded early, inventory and plant capacity are checked first. This means checking to see if scheduling the requirement early will prevent another requirement from being welded on time or will cause an inventory shortfall for a requirement with an earlier due date. If the requirement can be scheduled early without violating inventory and plant capacity constraints, the requirement is added and the weld plant schedule is adjusted accordingly. A similar process is invoked when weld completion dates are assigned.

In order to minimize distance traveled, the geography of scheduled rail projects is represented in such a way that requirements for nearby projects will be grouped together and deliveries to a number of projects along a single path can be made by the same train. Rail projects can occur at any point in the UPRR track network. Each project location is likely to have several different routes between it, the assigned weld plant, and the other project locations (hence the Traveling Salesman element of the problem). The heuristic applied to consistently achieve short routes between these locations involves simplifying this network to a single path between each project location and the weld plant. This is achieved by constructing a geographic network of rail delivery and pickup sites for each weld plant. The weld plant is the center of the network, with the shortest distance paths to the other locations radiating out from there (see figure 5). While this representation is not guaranteed to produce the absolute minimal distance schedule, it will consistently produce short routes for each train load without having to consider alternative routings.

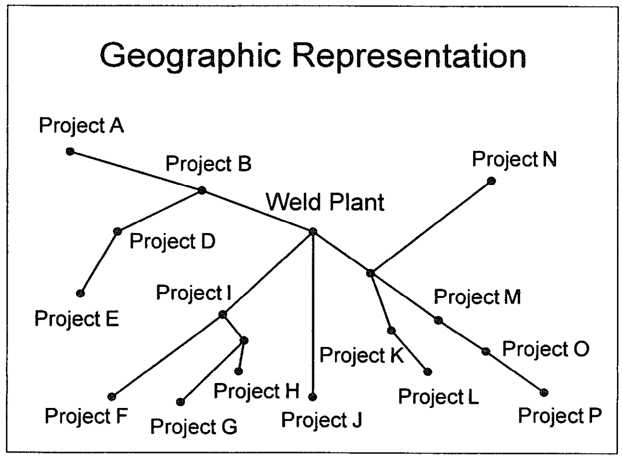

Figure 5

Loads are constructed by traversing this geographic network, moving from the outermost points in towards the weld plant. Requirements are grouped into loads if they lie on the same path and have onsite dates (or earliest weld dates, in the case of inventory shortfalls) that lie within a user defined timeframe. Partial loads are combined at forks in the tree, if all the requirements are needed within the same timeframe. The order of the requirements within each load is defined by the order of the requirements on the path, insuring the shortest travel distance. The heuristic to minimize distance also involves minimizing the total number of loads required, by combining requirements into full loads whenever possible. At the same time, the heuristics are implemented in such a way that several other constraints are met. These include the constraints to not exceed available rail inventory and plant capacity, the time proximity constraint, and the constraint that a single train will not deliver to projects in "different" directions from the weld plant.

RTS must handle rail trains of different capacities. Because the loads in a schedule are constructed prior to and independent of the assignment of actual trains, there is a question as to when a load is complete, with no room left for any additional rail. The heuristic for handling this involves determining the most common (modal) capacity of all the trains assigned to a weld plant. This capacity is then used globally as the upper limit for the size of most loads. It is exceeded only when increasing the load size to the capacity of a larger available train would eliminate the necessity for assigning part of a requirement to an additional load. Since the loads are constructed prior to train assignment, availability of large capacity trains is handled by setting aside reserved times on these trains for large capacity loads as the loads are constructed.

Application Development and Test

RTS was designed and implemented in ART*Enterprise®, interfaced with Oracle. It runs as a single user application under Windows, on a personal computer. This machine is connected across a LAN to a UNIX server, running Oracle. ART*Enterprise was selected because it is a powerful tool that includes, in a single, easy to use development platform, all of the capabilities required to implement the project. Among these are a screen painter and graphics tool kit for implementing sophisticated user interfaces, an interface to Oracle and other common databases, and the capability to integrate an object oriented approach with both rules and procedural code. All of this functionality was crucial to the implementation of RTS. Moreover, UPRR had satisfactory experience with the ART® family of products on other AI projects. Oracle was already in place as the UPRR standard for accessing the data required by RTS.

RTS was developed over a single calendar year and required about 3.0 man-years of effort. The completed application was tested by automatically generating schedules, based actual rail requirements, for a number of alternative scenarios that included varying the amount and availability of resources, such as rail trains and power units. The computer generated weld plant schedules were compared to manual schedules created by the scheduling expert. It was verified that the schedules produced were substantially the same. Where the schedules differed, the variance was most often due to either the limited availability of resources not scheduled in the manual process or the lack of accurate geographical data for certain rail requirements in RTS. The RTS generated scheduling of the rest of the resources could not be directly compared with manual output, since these resources are not scheduled as part of the manual process. However, the output was inspected by the expert for accuracy and reasonableness in meeting the prioritized goals of the scheduling process. The resource utilization displays were inspected by the developers and the expert and were verified to accurately reflect the actual resource utilization of the associated schedules. RTS passed user acceptance tests to the satisfaction of both the expert and his management organization and was deployed in January, 1996.

Application Use and Payoff

The scheduling of weld plants and rail trains at Union Pacific is the responsibility of a single domain expert who is the only initial user of the RTS application. The system is intended to be run on a daily basis, in order to automatically update weekly schedules based on actual requirements and available resources, and to incorporate emergency requirements into the schedule, both with little user intervention. RTS is also intended to be used as a decision support tool to aid in intelligently planning the acquisition of resources such as rail, trains, and power units, and assessing the need for a second shift at the weld plants. Both the amount and timing of resource acquisition can be estimated with the software. The rail train operations involve hundreds of trips per year by rail trains carrying hundreds of miles of rail. Anticipated benefits of $518,000 per year will be realized from timely response to emergency rail requirements, from being able to intelligently reduce the amount of resources needed to support the rail construction projects, and from scheduling resources only as necessary to support good schedules. There will also be a number of intangible benefits: preserving the expertise of the single UPRR rail scheduler and significantly shortening the time to produce schedules.

The initial usage of RTS proved to be a sufficient test for UPRR to validate the schedules that are automatically generated and to verify that the intended usage and the benefits expected to be derived from full use of the system are reasonable. It demonstrated that RTS will automatically produce good schedules, requiring little or no modification, in minutes. After some operational experience with the deployed system, however, it also became apparent that certain enhancements to the software would be required before there would be regular use of the system and achievement of significant benefits.

The poor quality of the rail requirement data that serves as input to RTS formed a significant barrier to regular use of the system. Cleaning up this data before running the scheduler was a very manual and time consuming process that had to be repeated each time new rail requirements were downloaded from the mainframe. Operational changes that will improve data integrity and timeliness may take several years to effect. Consequently, it became apparent that the requirements editing process needed to be automated as much as possible. Initial experience with RTS also served to emphasize the desirability of two other enhancements: an expanded capability to define the initial conditions of the scheduling process and the ability to arbitrarily define the timeframe of the schedule to be generated.

Application Enhancements and Maintenance

The enhancements identified during the initial deployment of RTS have been designed and developed during the past year. The most significant of these has been the requirements editor. The rail requirements are maintained in a mainframe system that is not integrated with RTS. Changes to requirements in RTS are not reflected in changes to the mainframe requirements. Moreover, due to operational restrictions, the rail scheduling expert is not allowed to change the data in the mainframe to correspond to the corrections he makes in RTS. Consequently, in the initial release of RTS whenever the latest requirements were downloaded all previous edits were lost.

RTS was designed to be run from the most recently downloaded requirements. The user could also edit the schedule produced by a previous run, and then run the scheduler from requirements derived from the edited schedule. However, there was no capability to edit the requirements directly, prior to running the scheduler. To make corrections to the downloaded requirements, the user would run RTS with the unedited requirements, edit the resulting schedule, and then rerun the scheduler using requirements based on the edited schedule. This was a very time consuming process, especially since requirements are downloaded nightly.

The enhanced system has separate editors for requirements and schedules. There is also a reconciliation process to merge the edited requirements with downloads from the mainframe, including a small knowledge base to determine:

- whether there is a *corresponding* requirement in the download for each of the edited requirements,
- which value should be used if there is a disagreement between downloaded and edited values,
- when the user should be queried to make the final decision.

Edited requirements now contain additional information that is used by the knowledge base and the user to make data integration decisions. This information includes the source and modification status of the requirement, i.e., whether it came from the mainframe or was added by the user, and whether it was modified from the mainframe value. It also includes the last two downloaded values for each of the editable fields.

The knowledge base for reconciling the data is composed of a number of rules. Some of these rules are general, such as:

If an edited requirement has status *modified* **and** a corresponding mainframe requirement is found **and** the mainframe requirement has changed from its previous value **and** the edited requirement now matches the mainframe requirement,
Then set the status of the edited requirement to *unchanged.*

If an edited requirement has status *modified* **and** a corresponding mainframe requirement is found **and** the mainframe requirement has changed from its previous value **and** the edited requirement does not match the current mainframe requirement,
Then ask the user whether to use the edited value and leave the status as *modified*, or use the mainframe value and set the status to *unchanged.*

Other rules are application specific, such as:

If an edited requirement does not have status *added* **and** the corresponding mainframe requirement was removed because it was *completed* **and** the delivered rail was equal to or greater than the total amount of the edited requirement,
Then change the status of the edited requirement to *deleted.*

If an edited requirement has status *modified* or *unchanged* **and** the corresponding mainframe requirement was removed because it was *completed* **and** the delivered rail was less than the total amount of the edited requirement,
Then ask the user whether to change the status of the edited requirement to *deleted* or *added.*

The initial enhancements to RTS have been developed by Brightware, Inc., with support from Union Pacific staff. The enhanced system was deployed in March of 1997, just as this paper went to press. At this point, the UP staff is assuming responsibility for the maintenance and any further enhancements to RTS.

Conclusion

This project encompasses many complex and interdependent tasks. Decomposition of the overall task into more manageable subtasks led to the development of an inventive solution. This solution incorporates heuristics derived from the expertise of the UP rail scheduler. Because of this, RTS produces weld plant schedules which are substantially the same as those produced by the expert. A benefit of this approach is that the schedules can be quite easily understood by their users. Another benefit is that the schedules produced are not nearly as sensitive to minor changes in input as is sometimes the case with a more traditional search-space approach.

RTS will become a valuable tool with the modifications now being deployed, but numerous opportunities still remain. Since daily use of the system is expected, performance will need to be assessed and tuned. In addition to a general performance tuning of the code, it is anticipated that performance can be improved via a hardware upgrade, a consolidation of database accesses, or an elective reduction of the scheduling timeframe.

RTS was not designed to address one major problem related to rail construction projects - the cascading of used rail from one project to another. Used mainline rail is often quite adequate for re-use as rail on less traveled routes or sidings. Presently, both the decision to convey rail from one project to another and the selection of which projects would best provide rail to which other projects rest outside of RTS. Incorporating the functionality to make these decisions into RTS could result in even better use of resources and further cost reductions.

References

Heng, G. K.; Seng, G. K..; Whye, L. C.; Hwa, T. P.; and Kiang, T. E. 1995. Scheduling of Marine Resources in the Port of Singapore Authority: A Total Approach. In Proceedings of the Seventh Annual Innovative Applications of Artificial Intelligence, 62-69. Menlo Park, CA: AAAI Press.

Hopcroft, J. E. and Ullman J. D. 1979. *Introduction to Automata Theory, Languages, and Computation*, 320-365. Reading, Mass.: Addison Wesley.

Husbands, P. and Mill, F. 1991. Simulated Co-Evolution as the Mechanism for Emergent Planning and Scheduling. In Proceedings of the Fourth International Conference on Genetic Algorithms, eds. R. Belew and L. B. Booker, 264-270. San Diego, CA: University of California.

Lee, J. K.; Lee, K. J.; Hong, J. S.; Kim, W.; Kim, E. Y.; Choi, S. Y.; Kim, H. D.; Yang, O. R. and Choi, H. R. 1995. DAS: Intelligent Scheduling Systems for Shipbuilding. In Proceedings of the Seventh Annual Innovative Applications of Artificial Intelligence, 90-106. Menlo Park, CA: AAAI Press.

Linnainmaa, S.; Savola, J.; and Jokinen, O. 1995. EPO: A knowledge based system for wood procurement management. In Proceedings of the Seventh Annual Innovative Applications of Artificial Intelligence, 107-113. Menlo Park, CA: AAAI Press.

Rowe, J.; Jewers, K.; Sivayogen, J.; Codd, A. and Alcock, A. 1996. Intelligent Retail Logistics Scheduling. In *AI Magazine* 17(4): 31-40.

Syswerda, G. 1991. Schedule Optimization Using Genetic Algorithms. *Handbook of Genetic Algorithms*, ed. L. Davis, 332-349. New York: Van Nostrand Reinhold.