Aurora

Intelligent Scheduling For Real-World Problems

Aurora is a sophisticated scheduling system that combines a variety of scheduling techniques, intelligent conflict resolution, and decision support to make scheduling faster and easier. The software's scheduling decisions take into account resource requirements, a variety of constraints, and any pertinent domain knowledge.

Once Aurora has created a schedule, it displays it in a series of graphical displays that allow the user to see the resource allocations and the temporal relationships among the elements. This display also allows the user to edit the schedule directly, intuitively, and easily.

As well as allowing rapid, easy schedule development, Aurora addresses issues disregarded by many scheduling systems.

Aurora focuses on resource requirements and temporal scheduling in combination; most scheduling systems focus on either temporal scheduling or resource selection, not both. Considering these different scheduling aspects in combination is especially important in domains such as class and curriculum planning, where there are a range of resource requirements, the time frame may be rather flexible but there are a number of temporal constraints, and each improvement in the schedule translates to a large pecuniary advantage for the client.

In connection with this, Aurora offers two special resource-related constraints. One is a same-resource constraint, which is associated with one aspect of each element's resource requirements, and guarantees that both of the elements will use the same resource to fulfill that requirement. It is useful when you want to enforce some degree of continuity, such as when you want the same specialist teach classes one and three of a seminar. The second constraint type is a spatial constraint, which allows the user to specify that two elements should be next to each other, or in the same spot. It is associated with spatial scheduling, another feature unique to Aurora. This is useful in everything from selecting lab space that is adjacent to a class's regular classroom, to allocating warehouse space.

Aurora allows extensive customization. Many systems are incapable of taking the client's special needs into account. Consequently, many of the benefits of an automated system are lost, because the user must make adjustments for all of the aspects of the problem that the system could not take into account. Needs and priorities vary widely from one company to another, even within a single domain, and the program needs to be able to reflect that. This customizability also allows the program to take expert domain knowledge into account, because this knowledge can easily be encoded into the heuristics that the system relies upon to make its decisions.

The built-in intelligence results in high-quality schedules, but in all cases the system assumes that the user should have the last word. There are always exceptions, so if the user makes changes (alters the resource allocations, changes the time window, etc.), the system will leave that element where the user placed it, unless the user explicitly indicates that the element should return to normal, floating status.

Features

Schedule Creation & Background

Component Types

Aurora offers three main kinds of scheduling elements: activities, which represent a single task; flows, which represent groups of tasks; and resources. For example, curriculums and courses would be defined as flows; a single class session would be defined as an activity.

In addition to these primary types there are also resource sets, which can be used to group resources, and constraints, which allow the user to create relationships among elements. These last two are described in more detail below.

User-Defined Slots

Often, specific elements - a given kind of activity, for example - need additional information associated with them. Aurora allows the user to add slots to the elements. These slots may be any of a number of types, allowing values ranging from calendars to text to numbers. They allow the user to represent everything from simple comments to an instructor's list of skills, which may later be used in categorizing them into resource sets.

Calendars

Activities and resources often have specific schedules: in a training environment, it might be one six-hour shift a day, five days a week; one two-hour shift a day, three days a week; or two two-hour shift a day, two days a week. In addition to these different schedules, it is often necessary to take holidays into account. A “five day” activity will in reality take very different amounts of time if it is scheduled in late December than if it is scheduled in February; and in a training setting such an activity would probably be delayed until after the holidays. Aurora provides calendars to reflect these complications. The user can define a standard schedule, plus exceptions - either one-time exceptions or yearly holidays.

Hierarchical Relationships

Aurora allows an experienced user to create and edit elements efficiently by taking advantage of a hierarchical relationship among the elements. Editing an element high in the hierarchy automatically edits all of the elements below it, unless the user has previously edited that property directly. This helps avoid the tedium of making the same change many times by hand, and the mistakes that are inevitable in such a situation.

Suppose, for example, that an instructor has just developed a new course. He has given it once, and discovered that it really needed a larger hands-on lab section. By taking advantage of hierarchical relationships, he can add an extra lab section to the base type of the course; this will automatically add an extra lab section to any other instances of that course that have already been created, unless they are locked in (see freezing). Without this feature he would have had to go into each instance of the course and add a lab for each of them - all the more daunting because such an addition would probably require adding various temporal constraints, to make sure that the lab takes place at the correct time.

Resource Requirements

Aurora allows you to associate resource requirements with any activity, flow, or resource. The first reflects the case where a single task requires a resource; the second reflects the case where a full set of tasks needs to use one resource (a class needs an instructor for its duration; a project needs a project manager); the third reflects those occasions when a resource needs another resource to operate properly (a large, specialized piece of lab equipment needs lab space; an engine needs fuel). These requirements may be defined directly in terms of resources, when only one resource can satisfy the requirement; or in terms of resource sets, when any of a group of resources may satisfy the requirement. It also allows you to designate alternate ways in which the set of resources may be satisfied. For example, an advanced course might require an experienced instructor for the duration, or a regular instructor with half-time support from a specialist.

Resource Sets

Aurora allows you to group resources arbitrarily into resource sets. Elements can then request these sets as part of their resource requirements, reflecting the idea that any properly qualified resource could perform a corresponding task. This also gives the user the freedom to group one resource differently in different situations. An experienced instructor, for example, might be grouped into both course development and teaching sets. This makes sure that all resources live up to their potential, and that the schedule takes advantage of all of their capabilities, not just one or two.

Constraints

Aurora offers a wide variety of constraints, allowing the user to define the scheduling problem as generally as possible, so that Aurora will automatically satisfy the specified relationships. Although a human could carefully define task 1 so that it occurred before task 2, it is far easier to create a constraint specifying that task 1 should finish before 2 begins; this makes it so that if the user needs to move one element, the other moves automatically to respect the constraint.

Constraint options include temporal, resource, and spatial constraints. Temporal constraints specify what the temporal relationship of two elements should be. A temporal constraint might be used to make sure that a more advanced course in a curriculum was scheduled later than the introductory course in a given series. Resource constraints indicate that two elements should use the same resource. If the beginning and advanced course in a specific class series should be taught by the same instructor, for the sake of continuity, the user would use a same-resource constraint

to reflect that. Spatial constraints may indicate that two elements should (or should not) be next to each other, or in the same location.

Scheduling

Freezing

Aurora allows the user to specify some elements as “frozen”, indicating they should not be moved or automatically altered, even in conflict resolution. This gives the user additional control over what is scheduled. They can freeze either a time frame, or a specific element.

This is useful for things such as preventing part of the schedule from changing once interested parties (e.g. students attending a course series) have been notified.

Default Scheduling

Aurora offers a default quality and prioritization scheme that has proven itself across a variety of domains. This functions well in most cases, and is useful for those situations where it is undesirable or impossible to construct scheduling heuristics customized to the domain at hand. Stottler Henke developed this algorithm while working on a scheduling project with NASA. It considers all possible resource allocations before it begins scheduling, allowing it to pinpoint possible trouble spots - the resources with the most elements vying for their potential use. It then works around these bottle necks, scheduling the elements away from them as much as possible, and leaving only those elements that must use them.

Display Features

Editing from the Schedule Display

When editing an existing schedule, it is inconvenient to flip back and forth from the display to the editing tab. Aurora avoids this problem by allowing the user to make all the editing changes possible from the editing tab from the schedule itself. You can drag elements in the resource allocation displays, changing the resource to which they are allocated; the quantity they require of a given resource; and the time window for the task. You can also double-click on an element, to bring up the editing dialog box from which you can alter the times precisely, add constraints, etc. In addition to editing schedule information, you can alter the appearance of the element - both its color and label are in the user’s full control.

Creating Elements from the Display

Aurora allows the user to create both activities and resources from the schedule display itself. This is especially valuable for filler activities such as vacations and down-time, which often need to be defined in relation to the existing schedule - a tedious process if the user must create it from the edit tab. It is also useful because the elements show on the display as soon as they are created, unlike elements created from the edit tab - the latter must be scheduled before they are displayed.

Markup

Aurora offers a variety of graphical markup with which the user can annotate the graphical schedule results. This markup is saved with the file and will show on both printouts and in the exported GIF image.

Conflict Manager Regulation

The conflict manager is the scheduling component that attempts to resolve conflicts during the scheduling process. Aurora includes a graphical control of the conflict manager, which the user can use to regulate how much conflict management takes place during scheduling. When starting a schedule, the user might turn it down or off, because he knows that there will be many unresolvable conflicts (in a situation with perennial over-allocation), or want a coarse layout within a few seconds, so he can see the shape of the schedule. When nearing the end of a schedule, the user might turn it up high, because he wants the schedule to be as good as possible, even if it takes a couple minutes to run.

Conflict Viewing

Any resources shown on the schedule display show any conflicts in red, so the user can see and deal with them. However, there may be instances where he wants a global view of all conflicts in the schedule. The user can see these using a conflict display window. The conflicts are broken down first by resource, and then by time frame.

Ignoring Conflicts

There are occasions when there is a conflict, but the user knows it is not a “real” conflict - they know that a vendor’s delivery is going to be late, or that

two classes can, in this situation, use the same lab intermittently. In such a situation, the user can specify that a conflict be “ignored”. It will no longer be displayed as a conflict, and the conflict manager will not try to resolve it.

Output

Reports

Aurora allows you to export comma-delimited reports about resource usage, which can then be opened in Excel or another spreadsheet program. The default report indicates what is using the selected resource group through time.

Export to GIF

Aurora can export any of its schedule displays to a GIF image the size of the schedule display itself, allowing the user to include the schedule in presentations, post it, or otherwise distribute it.

Printing

Aurora can print any of its schedule displays (resource allocation displays and temporal displays). It uses the currently viewed time frame to determine print bounds, resulting in an intuitive output paradigm that is flexible enough to show a variety of views.