A Beam Search Heuristic for Multi-Mode Single Resource Constrained Project Scheduling

Chuda Basnet
Department of Management Systems
The University of Waikato
Private Bag 3105
Hamilton
chuda@waikato.ac.nz

Guochun Tang
School of Management
Shanghai Second Polytechnic University
50Yuan Ming Yuan Road
Shanghai 200002, China
gtang@public1.sta.net.cn

Tadashi Yamaguchi
Faculty of Information Media
Hokkaido Information University
Nishinopporo 59-2, Ebetsu-shi
Hokkaido, Japan
tad.yama@do-johodai.ac.jp

Abstract

In this paper, the beam search technique is applied to resource constrained project scheduling where there is a single renewable resource to consider. Such projects occur frequently in practice: such as use of labour in construction projects, or constraints on number of programmers to carry out a software project. Usually the manpower needs are estimated in units such as work-hours or work-days. The multi-mode consists essentially of how many people can be employed to finish an activity. This problem is also called the discrete time/resource trade-off problem. The beam search employs a truncated breadth-first search tree, where all the choices are evaluated at each node, but only a limited number of the choices are selected at each level for further search. A number of evaluation techniques are presented for employment within the beam search heuristic. Computational results are presented.

1 Introduction

In this paper, the focus of discussion is resource constrained project scheduling where there is a single renewable resource to consider. Such projects occur frequently in practice: such as use of labour in construction projects, or constraints on number of programmers to carry out a software project. Usually the manpower needs are estimated in units such as work-hours or work-days. The multi-mode consists essentially of how

many people can be employed to finish an activity. For example, if it takes 72 work-hours to finish an activity, one mode may be with 6 workers, who will finish the work in 12 hours. However if 10 workers take up the project, it may take 8 hours. There may be minimum and maximum limits on the number of workers to be assigned. Thus the main concern is with the work content W_i for each activity i, and if r_{im} and d_{im} are the resource requirements and durations for some particular mode m, $r_{im}\, d_{im} \geq W_i$: only efficient modes are considered, where higher resource requirements result in lower durations of activities. No pre-emption is allowed. This problem is also called the discrete time/resource trade-off problem. The beam search technique is applied to generate a makespan minimising schedule for such projects.

2 Literature Review

Resource constrained scheduling has been a subject of research for a number of years now. A number of reviews of this literature has also been published [3, 7, 8]. So it is not the intention here to provide an exhaustive literature review. Thus only selected references are discussed here, which are relevant to the presentation that follows.

Boctor [1] compares a number of heuristics for multi-mode resource constrained projects. These heuristics break the multi-mode resource constrained scheduling problem into two parts: selection of an activity and choosing a mode for that activity. Priority rules such as SLK prioritise the activities so that an activity with the highest priority can be chosen. For example, at each decision point in SLK the slacks of all the eligible activities are recalculated, and the activity with the smallest slack is chosen. Similarly the rule SFM chooses the mode with the shortest feasible duration. Boctor [1] tested combinations of activity priority rules and mode selection rules on randomly generated sets of problems. The combination SLK-SFM performed best, but the combinations RWK-SFM, CAN-SFM, and LFT-SFM were very closely behind.

In the SLK rule, at each decision point the activity with the smallest slack is chosen, slack calculation being based on the shortest possible duration for the not-yet-scheduled activities and the selected duration for already-scheduled activities [1]. The LFT rule is based on the same durations as SLK except the latest finish time is calculated, and the activity with the minimum latest finish time is selected. In applying the RWK rule, the remaining work for each candidate activity is calculated as the sum of the shortest possible durations of the activity and all its successors, and the activity with the maximum remaining work is selected. When using the CAN rule, one calculates the number of subsequent candidates, a subsequent candidate to an activity x being defined as an activity which becomes or remains schedulable if x is scheduled to start at the considered period. The activity with the maximum number of subsequent candidates is selected.

A branch and bound search for the single resource/time tradeoff problem has been presented by Demeulemeester *et al*. [4] . This algorithm is basically an extension of the authors' earlier work [6] on the multi-resource version of the problem, and can optimally solve problems with up to 30 activities. The procedure employs a look-up table to see if a search node with the same activities have been searched before, and it employs a lower bound which is based both on the remaining critical path length of the eligible activities and on the ratio of the total remaining work load to the resource limit. This lower bound is particularly useful for a truncated search such as employed here, as it is able to pick up nodes with potentially minimal makespans.

Chang *et al*. [2] used beam search in job scheduling of flexible manufacturing systems. This problem is identical to jobshop scheduling with a makespan minimization objective, except that alternate machines are allowed for operations. In their scheme the search is done chronologically, and at each epoch where there is a choice, the choice is evaluated by using the shortest processing time (SPT) rule to complete the schedule. The makespan thus found is used as a guide to retaining the choice or to drop it. This search scheme outperformed commonly used dispatching rules. This success provided the motivation for the research presented here.

3 Problem Statement

Indices:

$i,j = 1 \ \ldots N$ is the index of activities on node. The activities are ordered such that a higher index does not precede a lower index. 1 and N are indices of dummy activities that have no resource requirements or durations and signify the start and end of the project.

$t = 1 \ \ldots T$ index of time periods.

$m = 1 \ldots M_i$ index of modes.

Parameters:

d_{im} = Duration of activity i in mode m $(1 \le m \le M_i)$.

r_{im} = Resource requirement of activity i, in mode m.

R = Available resource quantity.

W_i = Work content of activity i $(r_{im}d_{im} \ge W_i)$.

$i \rightarrow j$ = Activity i precedes activity j.

Decision variables:

f_i = Finish time of activity i.

S_t = The set of activities that are active in time t, = $\{i \mid f_i - d_i < t \le f_i\}$.

x_{im} = 1 if activity i is scheduled in mode m,

0 otherwise.

With this notation, the problem may be formulated as follows:

$$\min \ f_n$$

$$subject \ \ to$$

$$f_i \ge f_j + \sum_m d_{im} x_{im}, \ \ j \rightarrow i$$

$$\sum_{S_t} \sum_m r_{im} x_{im} \le R, \ \ t = 1,2,.... f_n$$

$$\sum_m x_{im} = 1$$

$$x_{im} \in (0,1)$$

The objective function is the finishing time of the last (dummy) activity, thus the formulation seeks to minimise the makespan. The first constraint enforces the precedence of activities. The second constraint stipulates that resources cannot be used more than they are available. The third constraint ensures that only one mode can be selected for each activity.

4 A Beam Search

This problem is strongly NP-hard. Thus optimum solutions of this problem are hard to find. Demuelemeester *et al*. [4] have presented a branch and bound search approach for this problem. Even with good lower bounds, and dominance rules that prune the search tree considerably, the exhaustive search tree can grow very large, and the maximum number of activities in their paper is 30. On the other hand there are many heuristics that can schedule multi-mode resource constrained projects in a single-pass, myopic fashion. Obviously, these heuristics are very fast, and do present satisficing but not optimal solutions. A middle ground between these two extremes of exhaustive search and single pass heuristics is to prune the search tree very harshly, allowing only a very limited number of nodes to be sprouted. The beam search is such a technique. In this approach the number of nodes allowed at any time is set to a limit, called the beam width. This approach is commonly used in the artificial intelligence area, where the search tree can often be very large.

Figure 1 shows the search tree in a beam search with beam width 2, meaning only two nodes are allowed to sprout at any one stage. The first node 1 sprouts three nodes 2, 3, and 4. These three nodes are evaluated, and node 2 is not considered very promising and dropped from further consideration. The nodes 3 and 4 together sprout the nodes 5, 6, 7, 8, and 9. After evaluation, only nodes 7 and 8 survive at this stage. Similarly nodes 12 and 13 are deleted after evaluation at the third stage.

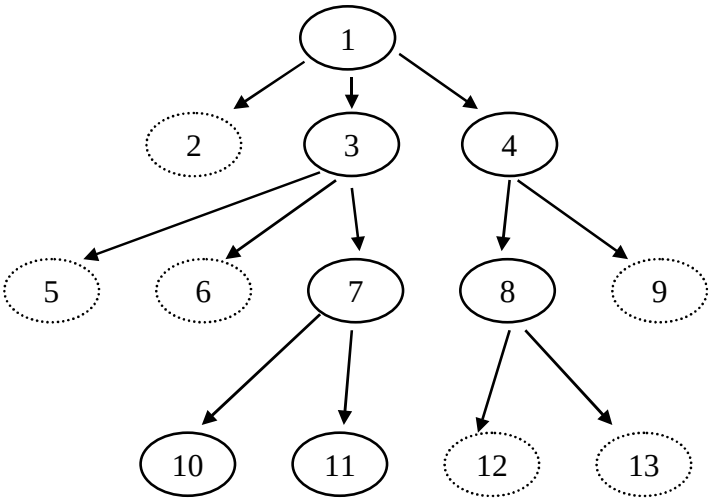

Figure 1. Scheme of a beam search

Beam search is a breadth first search, since all the nodes are evaluated at each stage before going any deeper in the search tree. The success of a beam search depends on the evaluation function that is employed at each node. The requirement here is different from the lower bounds in a branch and bound search, since an inadequate lower bound does not result in a non-optimum solution. Because of the exhaustive search in the branch and bound, the optimum solution is bound to emerge. The success of the lower bound is only in reducing the search. However, an inadequate evaluation function in a beam search may prune potentially desirable branches of a search tree forever, and will result in unsatisfactory solutions.

4.1 The Evaluation Function

A search tree is employed, where the search proceeds in a chronological fashion. Each node represents a point in time in the schedule where an activity is finished. At that time all the activities eligible to be scheduled at that time are determined, and each feasible activity-mode combination is a node to be sprouted from that node. To evaluate such a sprouted node, the following evaluation functions were experimented with in this research.

4.2 Demeulemeester Lower Bound (DLB)

Demeulemeester *et al*. [4] have employed a lower bound for their branch and bound search, which provides a strong lower bound. This lower bound (DLB) may be used for an indication of the quality of a particular search node. DLB is constructed both from the precedence consideration of eligible activities. The extended precedence-based lower bound is:

$$lb_0^{ext} = \max \{\max_{i \in E'} x_i, \max_{i \in S_i} \{\min (x_i, y_i)\}\}$$

In the above, E' is the set of activities that are eligible for scheduling at the next decision point t', and $S_{t'}$ is the set of activities still in progress at that time (t'). For each of these activities i:

x_i denotes $t' + RCP_{i1}$, where RCP_{i1} is the remaining critical path length of the activity computed under the assumption that all the succeeding activities are scheduled in the shortest duration modes.

y_i denotes $s_i + RCP_{im}$, where s_i is the start time of the activity (still in progress) and RCP_{im} is the remaining critical path length of the activity computed with the duration of activity i as is under progress, and all the succeeding activities are scheduled in the shortest duration modes.

Besides the above precedence-based lower bound, Demeulemeester *et al*. [4] also provide a resource based lower bound, however its inclusion did not improve the results in this research.

In the beam search, for each search node, DLB is computed, and the best nodes (number = beam width) are selected for further search, the rest are discarded.

4.3 Heuristic Based Schedule Extension (HSE)

A node in the beam search may be evaluated simply by extending the schedule represented by the node until all the activities are finished by using a simple heuristic [2]. The LFT-SFM heuristic [1] is employed here for this purpose. To evaluate a node, all the unscheduled activities in the current node are scheduled following this heuristic. The heuristic selects an activity based on the minimum of the latest finish time (LFT) and uses the shortest feasible mode (SFM) for this activity. The LFT heuristic is chosen here over the SLK rule for the sake of efficiency, even though SLK is known to be marginally better [1]. The LFT is calculated before the start of the search by backward recursion from the last activity in a schedule, which is constructed using the earliest start times and the shortest duration modes. The makespan of such a schedule provides an evaluation of the mode. Resource limitation is maintained throughout the schedule. The beam search based on this schedule extension is denoted by HSE hereinafter.

4.4 Resource Based Schedule Extension (RSE)

In this evaluation, a node in the beam search is extended into a schedule, but without regard to resource limits. Thus all the previously scheduled activities follow the resource limits, but the rest of the activities are scheduled with earliest starts, using their shortest durations, without considering resource limits. The makespan of such a schedule is later modified to the extent of resource limit violation.

At the next decision point after the activity-mode combination under consideration, the scheduling of all the activities is extended until all the activities are finished. This is carried out without regard to the resource limitation, and using the least duration mode for each future activity. Only the precedence constraints are heeded. The makespan achieved by such a schedule is then incremented each time a resource violation occurs. If over a period from t_1 to t_2, the amount of resource used in the extended schedule is P ($P > R$), then the increment is $(t_2 - t_1) * (P-R)/R$. The motivation of this increment is that there is a single resource, the activities have work content W_i, and the modes are efficient in regard to the use of the resource.

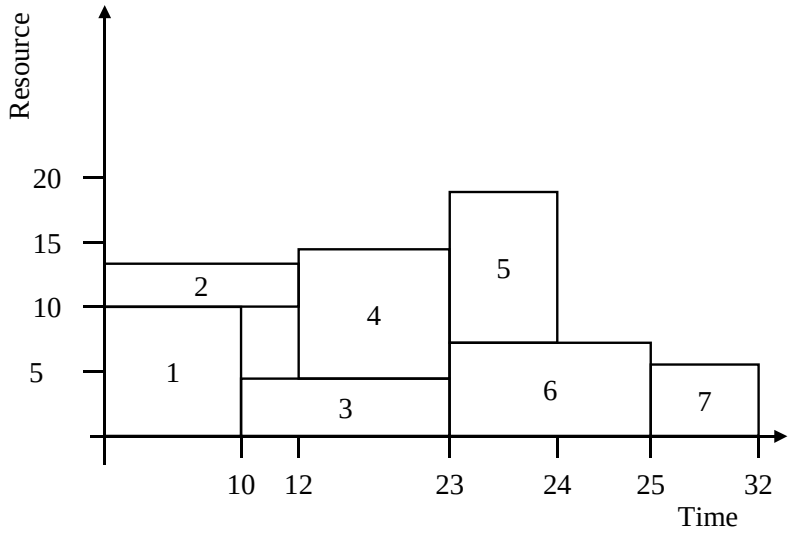

Figure 2. Resource based schedule extension

Figure 2 shows the resource histogram for a project. Say at a node, the time is 12 up to which point the schedule is feasible of course. At time 12, one of the activity mode combination to be sprouted is the scheduling of activity 4. To evaluate this sprout, the schedule is extended in a non-constrained fashion from that time on. Say the resource limitation is 16. The makespan is 32 now, but there is a resource violation from time 23 to 24: the resource used is 20. So the makespan is incremented by (24-23) $\times$ (20-16) /16 = 0.25, and the evaluation of this sprout is 32 + 0.25 = 32.25. This is done each time there is a resource violation. The evaluation is meant to be an estimation of the makespan if this sprout was continued. This evaluation is applied to each activity-mode combination at this point, and the number of sprouts retained is determined by the beam width employed. The beam search based on this resource consumption based schedule extension is denoted by RSE hereinafter.

5 Computational Results

To gauge the performance of the above evaluation schemes, beam search was implemented with the above evaluation schemes. The schedules generated by the beam search were also compared with a schedule based entirely on the SLK-SFM heuristic, where the priority of an activity at any decision point is given by the dynamic slack of that activity, and the shortest feasible mode for this activity is chosen.

5.1 Problem Generation

The addition/deletion algorithm of Demuelemeester *et al*. [5] was used to generate the network. Given the number of nodes and arcs, this algorithm generates a network such that:

i) the network has one start and one end node;
ii) the indegree of all nodes (except the first) is equal to or greater than one,
iii) the outdegree of all nodes (except the last) is equal to or greater than one,
iv) the generated network is completely randomised, all feasible networks with the node and arc counts are equally probable.

The work content of each activity was sampled from a uniform distribution of (10, 100). The number of modes for all activities was sampled from a uniform distribution of (1, 3). For each of these modes, the resource requirements were generated from a uniform distribution of (1, 10). The duration of the mode was calculated as the ratio (rounded up) of work content to the resource requirement. To generate the number of resources available, the above activities were scheduled on an earliest start basis without resource limits, with the shortest duration modes. The number of resources was set at the average (rounded) of the amount of resource used in this earliest start schedule.

5.2 Results

The algorithms were all programmed on an IBM PC (233 HZ, 64 MB RAM) in Microsoft Visual C++ (version 6). A beam width of 3 was employed. This implementation of beam search allows activity-mode combinations that are not necessarily maximal: a maximal activity-node combination is such that resources are fully used - no further activities can be added and the chosen activities cannot be carried out in shorter mode durations. The average makespans and average run time in seconds are shown in Table 1. The problem size is shown in the table as [nodes, arcs], 15 problems of each size were randomly generated and solved.

Table 1. Computational results

Problem size	SLK-SFM		DLB		HSE		RSE	
	Average Makespan	Average Time in seconds	Average Makespan	Average Time in seconds	Average Makespan	Average Time in seconds	Average Makespan	Average Time in seconds
[30, 90]	200.4	0.01	175.8	0.08	170.8	0.17	168.47	0.09
[50,250]	308.73	0.03	283.87	0.28	267.53	0.49	264.53	0.23
[100,1000]	660.6	0.13	619.6	1.23	579.73	2.40	579.4	0.48
[150,2250]	1026.13	0.30	948.4	0.75	918.67	4.96	895.2	2.60

As can be seen from the table, all the three beam search heuristics provide significantly better solutions than the simple SLK-SFM heuristic but at the cost of a significantly higher computational burden. The DLB heuristic provides better solutions than SLK-SFM, even though its evaluation function was not originally designed for this purpose. The RSE and HSE heuristics provide comparable performance, but RSE performs slightly better in terms of makespan, and significantly better in terms of computational time.

6 Conclusion

This paper presented three approaches to employ beam search in the multi-mode, single resource, project scheduling problem with makespan minimisation objective. While the beam search procedure generally performs better than a simple heuristic, the improvement in makespan comes at significant computational expense. Within the beam search algorithms, the RSE heuristic proposed in this paper performs slightly better than the other heuristics for the test problems generated in this study. There is a need for further work in developing a better evaluation function for use in the beam search technique and in carrying out more tests with different network characteristics.

References

[1] F.F. Boctor, "Heuristics for Scheduling Projects with Resource Restrictions and Several Resource-Duration Modes," *International Journal of Production Research*, 31 (1993), pp. 2547-2558.

[2] Y.L. Chang, H. Matsuo, and R.S. Sullivan, "A Bottleneck-Based Beam Search for Job Scheduling in a Flexible Manufacturing System," *International Journal of Production Research*, 27 (1989), pp. 1949-1961.

[3] E.W. Davis, "Resource Allocation in Project Network Models - a Survey," *Journal of Industrial Engineering*, 17 (1966), pp. 177-188.

[4] E. Demeulemeester, B. De Reyck, and W. Herroelen, "The Discrete Time/Resource Trade-Off Problem in Project Networks: a Branch and Bound Approach," *IIE Transactions*, 32 (2000), pp. 1059-1069.

[5] E. Demeulemeester, B. Dodin, and W. Herroelen, "A Random Activity Network Generator," *Operations Research*, 41 (1993), pp. 972-979.

[6] E. Demeulemeester and W. Herroelen, "A Branch and Bound Procedure for the Multiple Resource-Constrained Project Scheduling Problem," *Management Science*, 38 (1992), pp. 1803-1818.

[7] W. Herroelen, B. De Reyck, and E. Demeulemeester, "Resource Constrained Project Scheduling - a Survey of Recent Developments," *Computers and Operations Research*, 25 (1998), pp. 279-302.

[8] L. Ozdamar and G. Ulusoy, "A Survey on the Resource-Constrained Project Scheduling Problem," *IIE Transactions*, 27 (1995), pp. 574-586.