AN ALGORITHM FOR CREATING MASTER SCHEDULES THAT MINIMISES SCHEDULE REORGANISATION RESULTING FROM ADVERSE RISK EVENTS

Steven R. Davis[1]

ABSTRACT

Lookahead planning is a process that identifies constraints that need to be removed in order for the activity to proceed. If an activity has constraints that cannot be removed in time for it to start then the activity needs to be rescheduled and work may need to be found for the resources that would otherwise be idle. This may impact many subsequent activities that depend on the delayed activity reducing the reliability of the schedule and potentially extending the project.

This paper presents an algorithm based on a risk model for creating a master schedule for a project arranged to minimise the reorganisation required when delays occur to activities that have some probability of being delayed. The model assumes that the project will be utilising a lookahead process so that it will be recognised ahead of time that a particular activity will be delayed. It also assumes that the probabilities that different activities will be delayed, and an estimate of the amount of warning of each delay that will be given by the lookahead process, can be obtained from risk analysis.

A case study example is presented that highlights the algorithm's effectiveness.

KEY WORDS

Risk management, lookahead process, schedule planning, schedule reorganisation, replanning.

INTRODUCTION

The Last Planner^{TM} system of production control breaks planning into three levels: initial planning, lookahead planning, and commitment planning. Initial planning produces the overall budget and schedule, lookahead planning ensures that activities will be able to be executed when they should, and commitment planning assigns work tasks to individual work crews (Ballard and Howell, 1998).

It is common on construction projects for events to arise that prevent activities from being carried out when originally scheduled. In the Lean Construction paradigm anything that can prevent an activity from being executed is referred to as a constraint (Hamzeh et al, 2008). Lookahead planning is the process of going through all the constraints for each activity in the period leading up to the scheduled start of that

[1] Lecturer, School of Civil and Environmental Engineering, University of New South Wales, Sydney NSW 2052, Australia, Phone +61 2/9385-5052, FAX 2/9385-6139, s.davis@unsw.edu.au

activity and resolving or removing each one. If an activity has one or more constraints that have not been removed by the time it is scheduled to start it will not be ready for commitment planning and so cannot be started. This creates two problems: firstly different work needs to be found for work crews that would have been allocated to the unavailable work, secondly other activities may not be able to start until the delayed activity has finished. This flow on effect will reduce the reliability of the schedule as more activities need to be rescheduled, and may potentially extend the duration of the project.

Ideally a well run lookahead process would recognise ahead of time that a constraint is not going to be removed in time. This gives advance warning to the managers to take action. This action needs to address the two problems of the delay by making other work ready for the work crews that would otherwise have nothing to do (referred to as workable backlog in Lean Construction terminology, Ballard, 1997), and replanning other parts of the project to bring it back on schedule and keep the reliability of the schedule high.

This paper presents an algorithm based on a risk model for creating/selecting a master schedule for a project that is arranged in such as way as to maximise the ability of managers at later times to rearrange the schedule with minimum effect on the planning reliability of later activities and on the overall project duration. In this model it is assumed that a risk management exercise executed before the schedule is created has identified risk events that have some likelihood of preventing the removal of certain constraints during the lookahead process. Planning reliability is preserved by ensuring that after replanning every activity still has sufficient time for its lookahead process.

With regard to the initial planning stage Lean Construction differentiates between master scheduling and phase scheduling (Ballard and Howell, 2003). The algorithm proposed here can be used at either or both stages.

MODEL

GENERAL CONCEPT

With the ubiquitous understanding of the critical path method (CPM) many managers think of their project in terms of predecessor to successor relationships, even though many of these relationships are really resource limited non-concurrency relationships (Huber and Reiser, 2003). For example the electrical work on the first floor needs to be completed before the electrical work on the second floor because there is only one electrical crew to do the work. However, when something happens to delay the start of the electrical work on the first floor the manager will recognise that this is not a hard constraint and that the work on the second floor can be done first, with the first floor being done later. Of course there are many more constraints on the project that need to be considered. For example the plumber also needs to do work on the first and second floor, and both trades are more productive when they have exclusive access to a particular working space. Therefore simple reorganisations as above may turn out to be not so simple at all.

This leads to the situation where a delay in an activity may lead to modification of the project CPM network in order to bring the project back on schedule. The danger of doing this is that reorganising the project schedule like this can play havoc with the reliability of the schedule. Scheduling activities earlier may not leave enough time for removing their constraints. Scheduling other activities later may upset the resourcing plans of subcontractors unless an equivalent amount of work is scheduled earlier to take their place.

What the algorithm proposed in this paper does is to look at all possible predecessor successor networks that fit the hard predecessor to successor, resource and exclusive access to location constraints and chooses the one that gives the most flexibility to deal with the sort of network modifications mentioned above with the minimum repercussions. Since it uses data from a risk assessment exercise the flexibility is provided where it is more likely to be useful. In doing this it considers the amount of notice required to perform the lookahead process on each activity and ensures that this amount of time is available.

SPECIFIC DETAIL

The underlying model assumes that a project will employ a lookahead process. Each work task will be examined over a period of weeks prior to its planned execution and during that period actions will be set in motion in order to remove constraints that would prevent the work task from being executed at the planned time. Thus, for each task there will be a minimum notice period defined that provides the time necessary for this process.

In addition it is assumed that for at least some of these work tasks that there are risks whereby the actions initiated will not be able to remove the constraints in time. It is further assumed that a risk assessment exercise is able to identify these risks and give an estimate of their likelihood of occurring, the length of the delay that they will cause to the start of the work task, and the amount of time before the work task is scheduled to start that it would be known whether or not the risk has eventuated.

Furthermore it is assumed that each work task can be performed by only one particular trade/crew and hence two tasks requiring the same trade/crew cannot be performed at the same time. Similarly tasks that occur in the same location cannot occur at the same time.

In order to determine the optimum schedule the algorithm begins by ordering all of the project activities into an activity list. All possible permutations of the activity list are determined. Predecessor to successor relationships are placed between successive pairs of activities so that the activity list forms one long chain. These relationships will be referred to as soft relationships. Each activity list is then tested for feasibility by ensuring that no chain of soft relationships violates any of the hard predecessor successor relationships and all activity lists found to be non-feasible are rejected.

Next each of the remaining long chains of soft relationships is condensed into a project network. This is done by starting at the beginning of the chain and examining each of the soft relationships in turn. If the successor in the soft relationship cannot occur at the same time as the predecessor because of a hard predecessor to successor relationship, or because the two activities both either require the same trade/crew or the same location, then the soft relationship will be left as is. Otherwise the relationship will be deleted and new relationships will be created between the predecessors of the predecessor and the successor, and the predecessor and the successors to the successor, see Figure 1.

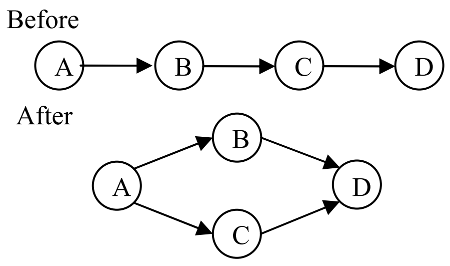

Figure 1: Condensing the Network when Tasks B and C can be Performed Simultaneously

The same result would have occurred in Figure 1 if the activity list was ordered ACBD as has occurred with ABCD. Therefore the network will be discarded whenever a link is broken like this if the predecessor has a higher activity index than the successor. This ensures that only one of the duplicates is retained for analysis. However, the schedules are not duplicates if one of the activities has a longer notice period for the lookahead process and that notice period will prevent its activity from starting at the same time as the other activity. In this case both networks need to be retained.

For each of the remaining networks a schedule is calculated based on the earliest start time for each activity. Note that these start times cannot be any sooner than the notice period required for the lookahead process. These schedules are guaranteed at this point to be at least semi-active following the language of Sprecher et al (1995).

Each activity will be considered to either be started or waiting. At the beginning of the schedule all activities will be considered to be waiting, and the set of started activities will be empty.

The algorithm then starts at the beginning of each schedule and steps through it day by day. For each day it firstly checks through each of the waiting activities to see if this is the point in time where it will be discovered whether or not a risk related to that activity has eventuated. If so then the risk will be removed from that permutation and two copies of the project will be created, one where the notice period of the activity is extended by the amount of delay caused by the risk and one where it isn't. For each of these projects the algorithm recurses to the point where all possible permutations of the activity lists are generated, except that this time only permutations of the waiting activities will be considered. Activities that have been already started are not considered in these permutations because the algorithm has already reached the point where those activities have been started and so they can no longer be rescheduled. The stepping through the schedules created by these permutations will continue from the current time rather than the start. On returning from this recursion the expected duration of the permutation will be set to the sum of the probability of the risk eventuating multiplied by the duration of the project where the risk delay has been added plus the probability of the risk not eventuating multiplied by the duration of the schedule where the risk delay was not added.

If at this point any activity is ready to start then it will be moved from the set of waiting activities to the set of started activities. The algorithm will then reduce the lookahead notice period of any activity that has entered its lookahead period by one day. Finally when the set of waiting activities is empty it will set the project duration for that permutation to be the finish time of whichever activity finishes last.

This algorithm has been implemented in Python 2.6 and the example in the following section executed on a PC with an Intel Core 2 duo CPU running Microsoft Windows Vista operating system.

EXAMPLE

In order to illustrate the algorithm an example project has been formulated. Details are given in Table 1 and Figure 1.

Table 1: Example Project for Illustration

Activity	Predecessors	Trade	Floor	Duration	Notice	Risk
Struct-1		Struct	1	5	0	
Struct-2	Struct-1	Struct	2	5	0	
Cure-1	Struct-1		1	20	0	
Cure-2	Struct-2		2	20	0	
Elect-1	Cure-1	Elect	1	5	10	5 / 0.2 / 5
Elect-2	Cure-2	Elect	2	5	10	5 / 0.2 / 5
Plumb-1	Cure-1	Plumb	1	5	15	10 / 0.2 / 10
Plumb-2	Cure-2	Plumb	2	5	15	10 / 0.2 / 10

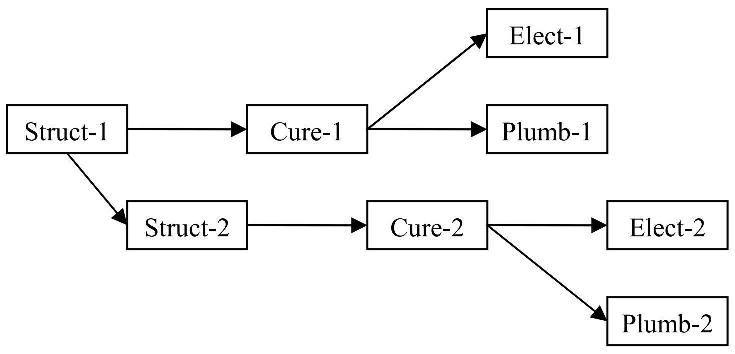

Figure 1: Precedence Diagram for Example Project Showing Hard Logical Constraints

The first four activities are not expected to have any risk and do not require a lookahead process. They essentially exist so that the rest of the project does not start in a vacuum. The latter four activities each require a notice period for their lookahead process and have a risk that might be discovered to have eventuated five days after their lookahead period starts.

For this example project there are 40,320 different permutations of activity list order. Of these only 140 permutations are feasible in that they do not violate any of the hard logical constraints. The condensing process discussed in the previous section eliminates all but 14 of these permutations. Four of the remaining 14 permutations create optimal schedules that will finish in 40 days if none of the risk events eventuate, six create schedules that will finish in 45 days and the remaining four create schedules that finish in 50 days. Figure 2 shows the four different 40 day optimum schedules, labelled A, B, C and D.

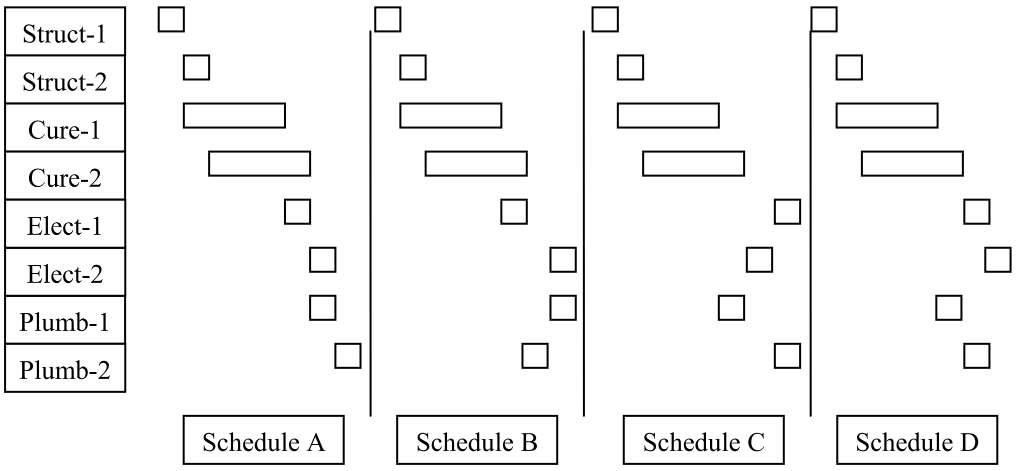

Figure 2: The Four Possible Optimum Schedules for the Example when Risks are Ignored. Each of these Takes 40 Days to Complete.

SINGLE RISK CASE

In looking at the behaviour of the algorithm it is useful to consider what happens when less than the full complement of risks is present in the example. This enables the behaviour of the algorithm to be highlighted in simpler examples where it is easier to see.

If a delay of 5 days to activity Elect-1 is considered in isolation, with no delay to any of the other activities then schedule B could still be completed in 40 days since Elect-1 has 5 days of free float in that schedule, which is equal to the delay, while schedules A, C and D would take 45 days if the networks of soft links behind the original schedules are adhered to. However, for schedule A the project can still be completed in 40 days, without changing the start time of any other activities, if Elect-1 is delayed by 10 days instead of 5 to be executed after Elect-2 and concurrently with Plumb-2. Thus if either schedule A or schedule B is selected initially then all of the other work tasks can be carried out at their originally scheduled time, despite the risk of a potential delay to Elect-1, and without affecting the reliability of the remaining schedule.

Similarly if a delay to activity Plumb-1 is considered in isolation then selecting schedule C as the initial schedule will preserve the 40 day project duration. However, it would require changing to schedule B at the time it is discovered that the delay has eventuated if this occurs. At this point in time there is still plenty of time to schedule the notice periods and carry out the lookahead processes for all of the activities that have been rescheduled, and so the reliability of the plan can still be preserved.

Isolated delays to Elect-2 and Plumb-2 will delay the schedule no matter which schedule is used. For the delay to Elect-2 any schedule can be used and the project will finish in 45 days. Disruption is not cased to other activities if Elect-1 is simply executed between time 40 and 45. However, if Plumb-2 is delayed then in order to finish the project within 45 days it is necessary that the project initially use schedule B or schedule D to ensure that Plumb-2 has entered the lookahead process early enough. Given that the example sets the probability of these risks as being 20%, the expected duration[2] for the project, if its only risk is a one of these second floor tasks, is 41 days.

DUAL RISK CASE

The situation is more complicated in the case where two of the activities face risks of being delayed. If the risk is only to the first floor activities then the project can still be completed in 40 days by starting with schedule D, switching to schedule A if the delay to Plumb-1 occurs (this would be discovered at time = 15, giving plenty of time for the lookahead processes of the activities that are moved to earlier starting times), and delaying Elect-1 until time = 35 if it is delayed (in either the schedule A or schedule D case).

The worst two risk case occurs is if it is both of the second floor activities that might be delayed. Schedule B copes with this case the best. If Plumb-2 is delayed it will be discovered at time = 20, at which time Elect-2 can be rescheduled 5 days earlier to take Plumb-2's place, giving Elect-2 5 days free float to cover its own possible delay. In any of the situations where one or both of the Elect-2 and Plumb-2 is delayed the project duration will be 45 days. Thus the expected duration becomes 41.8 days. Starting with schedule D can achieve the same result, but with potentially more intermediate juggling, so is not preferred.

In any of the cases where only one activity on each floor is facing a potential delay then it is possible to select an initial schedule that will result in an expected duration of 41 days. Table 2 shows the various combinations with their preferred initial schedules.

Table 2: Optimal Initial Schedules if Facing Two Risks

Activities with Risk of Delay	Best Initial Schedules	Expected Duration
Elect-1, Elect-2	A or B	41
Elect-1, Plumb-1	D	40
Elect-1, Plumb-2	B	41
Elect-2, Plumb-1	C or D	41
Elect-2, Plumb-2	B or D	41.8
Plumb-1, Plumb-2	D	41

[2] In this paper expected duration will refer to mathematical expectation, ie the time weighted by the probabilities. In this case 0.2 * 45 + 0.8 * 40 = 41. This gives what would be the average duration if the project could be executed a statistically large number of times.

COMPLETE EXAMPLE – FOUR RISK CASE

The complete example, where it is possible that any combination of the four risk events may occur, has an expected duration of 41.8 days, the same as the worst dual risk case. In fact in the worst case the project still finishes in 45 days, despite all four activities actually being delayed. Details of the step by step rescheduling involved in this worst case are given in Table 3. Note that the particular decisions made in this case are not "locked in" until the particular delay is found to actually eventuate. Thus if Plumb-2 and Elect-2 are not delayed then different decisions made at the time these facts are found out can still lead to the project finishing in 40 days.

Table 3: Optimal Rescheduling Sequence if All Four Risks Eventuate in the Complete Example. Initial Plan Is Schedule D.

Time	Delay Discovered	Other Rescheduling
15	Plumb-1 from 25 to 35	Elect-1 from 30 to 25
20	Plumb-2 from 30 to 40 Elect-1 from 25 to 40	Elect-2 from 35 to 30
25	Elect-2 from 30 to 35	

This example shows that by examining all available options the algorithm presented in this paper can improve project outcomes. This would not have occurred if the project had simply been scheduled using schedule A, being the natural schedule to select given the presentation of the activities in Table 1.

DISCUSSION

The case study example shows that the algorithm has the potential to improve planning reliability and project schedule outcomes by adding flexibility to the initial schedule. It does this in a slightly different way to the normal phase scheduling in Lean Construction. Part of the phase scheduling process involves allocating float to create particular buffers after activities that have a higher risk of not being completed in time (Knapp et al, 2006). The algorithm in this paper does something similar in that it tends to schedule risky activities earlier so that if they are delayed it has less impact on the project finishing time. In fact after scheduling activities using the algorithm it is still possible to add buffers in strategic places if desired.

The example presented here was not especially contrived to have the particular complex behaviour observed. It was simply used in testing and debugging the computer program and the author was pleasantly surprised to discover how effective the algorithm is at finding schedules that had the desired flexibility properties even in this simple case.

Even if the complete algorithm is not used as described here it would be possible to apply a simplified version after a delay occurs that causes a big impact on the schedule. This simplified version would not consider all of the risk events, but would simply sort through all of the possible schedules to find the best way to resolve the current crisis. It could still insist that no activities are scheduled that do not have sufficient time for their lookahead process.

An argument that could be presented against using the algorithm is that the problem could be solved by extending the lookahead periods for the risky activities so that there is always plenty of time for removing constraints. This many not be appropriate in certain circumstances. For example, one type of constraint that needs to be removed is ensuring the availability of resources to do the work. It may not be possible to schedule this too far ahead as these resources may be used on multiple projects and creating too rigid a schedule too early may decrease planning reliability instead of increasing it. Another issue is that it is assumed that the risk events being guarded against by the algorithm are unlikely. Therefore having extra time in the lookahead process for most of the activities will be unnecessary, and will come with the disadvantage that the lookahead process itself will have much more work in process, making the job of the planner more difficult.

Finally in order to use this algorithm a risk management exercise needs to be carried out initially. The very act of identifying risks is likely to lead to actions being taken to eliminate or mitigate these risks anyway. This will have a direct improvement on the reliability of the planning process.

FURTHER WORK

The algorithm as presented here is in an early stage of development. Improvements being considered include:

Improving the algorithm so that it can handle larger projects. Currently each schedule is calculated each time it is needed. Methods of caching schedules should be able to reduce execution time considerably allowing the computer implementation to handle much bigger projects.

More focus on continuous work flow for crews. Currently the only objective implemented in the algorithm is to minimise expected duration, while still allowing all activities to have sufficient time for their lookahead processes. Some form of penalty function could be introduced to schedules that do not provide continuous workflow for all crews.

Effect of activities taking more or less time than expected. The current implementation assumes that variability in task execution times will be dealt with at the commitment planning and lookahead planning stages. However, the algorithm could be modified to consider these types of delays/speed ups.

Division of float amongst activities to create buffers. The algorithm currently schedules activities to start at their earliest times based on the hard predecessor relationships, resource constraints and access to location constraints. This could be changed to explore the part of the solution space where buffers are inserted after risky activities

Conversion of semi active schedules into active schedules. The condensing process used by the algorithm to convert the chain of activities arising from each permutation into a useable schedule only generates semi-active schedules. Sprecher et al (1995) describes how active schedules can be created from semi-active schedules by “globally left shifting” activities. This may lead to better schedules in some instances.

CONCLUSIONS

This paper has presented an algorithm that can be used in the initial planning of a project to create a master schedule that provides maximum flexibility to deal with risk events that cause delays during the lookahead process. The algorithm is effective at finding schedules that have the desired properties of minimising project duration and preserving plan reliability. The algorithm is in the early stages of development and several improvements are planned.

REFERENCES

Ballard, G. (1997). "Lookahead Planning: The Missing Link in Production Control", *Proc. 5th Annual Conf. Int'l. Group for Lean Constr.*, IGLC 5, July, Gold Coast, Australia. pp. 13-26.

Ballard, G., and Howell, G. (1998) "Shielding Production: An Essential Step in Production Control", ASCE, *J. of Constr. Engrg. and Mgmt.*, 124 (1) 18-24

Ballard, G., and Howell, G. (2003) "An update on Last Planner", *Proc. 11th Annual Conf. Int'l. Group for Lean Constr*, IGLC 11, July, Virginia, USA.

Hamzeh, F.R., Ballard, G., and Tommelein, I.D. (2008) "Improving Construction Workflow – The Connective Role of Lookahead Planning", *Proc. 16th Annual Conf. Int'l. Group for Lean Constr.*, IGLC 16, July, Manchester, UK, pp635-646.

Huber, B. and Reiser, P. (2003) "The Marriage of CPM and Lean Construction", *Proc. 11th Annual Conf. Int'l. Group for Lean Constr*, IGLC 11, July, Virginia, USA.

Knapp, S., Charron, R., and Howell, G. (2006) "Phase Planning Today", *Proc. 14th Annual Conf. Int'l. Group for Lean Constr*, IGLC 14, July, Santiago, Chile, pp431-441

Sprecher, A., Kolisch, R., and Drexel, A. (1995) "Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem" *Eur. J. of Op. Res.*, 80 94-102.