Optimal Factory Scheduling using Stochastic Dominance A*

Peter R. Wurman
Artificial Intelligence Laboratory
University of Michigan
1101 Beal Avenue
Ann Arbor, MI, 48109-2110
pwurman@umich.edu

Generating optimal production schedules for manufacturing facilities is an area of great theoretical and practical importance. During the last decade, an effort has been made to reconcile the techniques developed by the AI and OR communities. The work described here aims to continue in this vein by showing how a class of well-defined stochastic scheduling problems can be mapped into a general search procedure. This approach improves upon other methods by handling the general case of multidimensional stochastic costs.

We consider scheduling in a multi-product, single machine factory with sequence independent setup times. The machine processing and setup times are random variables. The factory is faced with a set of jobs that incur a late penalty if not completed by their deadlines. The optimization problem is to generate a static schedule that minimizes the expected penalty.

Given that factory performance is stochastic, it is easy to show that an optimal solution to a deterministic model using the expected run times will lead to sub-optimal schedules. Instead, we tackle the stochastic problem directly using an algorithm called Stochastic Dominance A* (Wellman, Ford, & Larson 1995). SDA* is designed for problems with path-dependent, stochastic operator costs. In SDA*, paths can be pruned only if their path cost is stochastically dominated by an already discovered path to the same state. For heuristics to be admissibile, the actual remaining cost must be stochastically dominated by the heuristic estimate. In addition, we impose a benign consistency condition to retain validity.

Our work extends SDA* to the scheduling problem. We formulate the problem as a state space search where the state is defined by the current inventory, the current machine setup, and the orders that have been filled. Three classes of operators are allowed. `Make` operators increment the inventory of the state. `Ship` operators decrement the inventory and change orders from unfilled to filled. `Setup` operators change the current product that can be built.

The path costs are defined by the pair <*time*, *penalty*>. We must keep the full probability distribution for the time attribute, but, assuming we are risk-neutral in penalty, it is sufficient to store only the expected value of the penalty. The `make` and `setup` operators increment only the time element of the path cost. The `ship` operator incurs a penalty as a function of the current time.

The definition of costs as a two-attribute structure requires some small elaborations to the SDA* algorithm. First, the priority queue is sorted by the estimated expected penalty of the paths. The estimated expected penalty is the sum of the penalty accumulated so far plus the heuristic estimate of the rest of the penalty. Second, paths can be pruned only if a path has already been found to the same state that dominates it in both time and penalty. This notion of dominance over multi-element costs is the same used in multi-objective A* (Stewart & White 1991).

To compare the algorithm's performance versus the deterministic model, we ran it on randomly generated problems where the number of orders and the total requested capacity were varied. We solved 400 such problems with both SDA* and A* using a deterministic model based on mean run and setup times. We found that in more than 70% of the problems, the solution found using SDA* had a lower expected cost than the deterministic solution applied to the stochastic data. We also found that the number of nodes expanded and the percentage of pruning between the stochastic and deterministic algorithms were very similar.

References

Stewart, B., and White, III, C. 1991. Multiobjective A*. *Journal of the Association for Computing Machinery* 38(4):775–814.

Wellman, M. P.; Ford, M.; and Larson, K. 1995. Path planning under time-dependent uncertainty. In *Proc. 11th Conf. on Uncertainty in AI*, 532–539.