Local search for stochastic parallel machine scheduling: improving performance by estimating the makespan

Guido Passage, Marjan van den Akker, Han Hoogeveen
Department of Information and Computing Sciences
Utrecht University, Princetonplein 5, 3584 CC Utrecht, The Netherlands
g.j.p.n.passage@gmail.com, J.M.vandenAkker@uu.nl, J.A.Hoogeveen@uu.nl

Abstract

We consider the problem of finding a baseline schedule for stochastic planning problems with precedence relations. In such a situation it is hard to assess the quality of a move within a local search approach. It can be estimated using simulation, but this is slow. We present a new estimation method that is generally applicable and that is far more efficient than using simulation.

Keywords: Stochastic planning, resource-constrained project scheduling, local search, normal distribution.

Suppose that we have m workers who have to execute n jobs. For each job j we know its release date r_j, which is the earliest time at which job j is allowed to start, and we know its processing time p_j, which is the amount of time required for processing job j. The goal is to minimize the makespan, that is, to complete the last task as soon as possible. This problem is $\mathcal{NP}$-hard, but a good solution is easily found by using local search. In many situations, however, the processing times are not deterministic but stochastic. If we cannot change the allocation of the jobs to the workers during run time, then we can look for a good *baseline schedule* and minimize the expected makespan. In this case, we can still use local search effectively, since the expected makespan of a given solution can be computed rather easily as the maximum of a set of independent stochastic variables. The situation becomes much more difficult if the jobs are related through *minimum delay precedence constraints*, which specify that the execution of job j cannot start until a given amount q_{ij} of time has passed since the start of job i. Because of the interdepencies, there is no easy way to compute the expected makespan, which we need within local search to see whether a neighbor should replace the incumbent solution. In [1] we use discrete event simulation, but this requires a sufficient number of simulation runs to get a reliable estimate. To alleviate this computational burden, we have investigated several algorithms to estimate the makespan of a new candidate solution in each iteration of the local search.

We test our methods on the scheduling problem described above, which using the three-field notation scheme is denoted by $P|r_j, prec|C_{\max}$. The deterministic problem can be solved to optimality efficiently using column generation for all types of precedence relations (see [2]). In our application we restrict ourselves to minimum delay precedence constraints, where we assume that the corresponding values q_{ij} are deterministic, just like the release dates r_j. The processing times, on the other hand, are stochastic with known expected value and variance; we assume that the processing times are *indepent*. To specify a baseline schedule, we need to determine for each machine which jobs it executes and in which order this is done.

Our local search approach is based on Iterated Local Search in combination with Variable Neighborhood Descent. As neighborhoods we either move one or two jobs to another

Proceedings of the 12th International Conference on the Practice and Theory of Automated Timetabling (PATAT-2018), Vienna, Austria, August 28–31, 2018

machine or we swap two jobs. We have tested two estimations of the resulting makespan, which we compare to using k simulations per iteration:

- Approximating each start time using a normal distribution;
- Computing the expected makespan *without precedence constraints* and then add compensation for the expected delays.

Below, we only work out the first approach, which gave the better results and is more generally applicable.

Given a schedule, we compute the start time S_j of job j as the maximum of a number of stochastic variables, which are: r_j; the values $S_i + q_{ij}$ for all predecessors i of j; and the completion time of the machine predecessor of j. Denoting these stochastic variables by $D_1, \ldots, D_l$, we find that $S_j = \max\{D_1, \ldots, D_l\}$. Next, we use [3], who describe how to find the expected value and variance of $X = \max\{D_1, D_2\}$, where D_1 and D_2 are two normally distributed stochastic variables with correlation coefficient ρ. Pretending that all these stochastic variables D_k are normally distributed, we estimate $E[S_j]$ and $Var[S_j]$, by iteratively computing $X_k = \max\{X_{k-1}, D_k\}$, where $X_1 = D_1$; again, we pretend that the maximum is normally distributed. We found that the quality of the approximation depends on the order in which we consider the variables D_k; it is best to first handle the variables D_k describing precedence constraints involving jobs scheduled on the same machine as job j starting with the machine predecessor of job j, then take the remaining precedence constraints into consideration, and end with r_j. For stochastic variables corresponding to jobs on different machines we use $\rho = 0$; for the other ones, the stochastic variables describing the part after the last common time-point are unrelated again.

Computational experiments reveal that local search with the above algorithm finds solutions that are just as good as local search with estimating the makespan using 300 simulation runs per iteration, but only requires a fraction of the computation time. This suggests that estimating the makespan works better and faster than simulation in local search for stochastic parallel machine scheduling problems, and we expect that this approach can be applied in more general, comparable situations in other areas of planning.

References

[1] J.M. van den Akker, K. van Blokland, and J.A. Hoogeveen (2013). Finding robust solutions for the stochastic job shop scheduling problem by including simulation in local search. In V. Bonifaci, C. Demetrescu, and A. Marchetti-Spaccamela (Eds.) In *Experimental Algorithms – SEA 2013*, Vol. 7933 of Lecture Notes in Computer Science, 402–413. Springer Berlin Heidelberg.

[2] J.M. van den Akker, J.A. Hoogeveen, and J.W. van Kempen (2012). Using column generation to solve parallel machine scheduling problems with minmax objective functions. *Journal of Scheduling 15*, pp. 801–810.

[3] S. Nadarajah and S. Kotz (2008). Exact distribution of the max/min of two Gaussian random variables. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 16*, 210-212.