Part 3: Bayesian Optimization of the flexible Job Shop Scheduling Problem

Group 06
Janette Rounds, David Rice, Mitch Vander Linden

October 19, 2015

1 Introduction

The following algorithm attempts to find an optimized way to schedule the Job-shop Scheduling Problem (JSP) in a more flexible manner called the flexible Job-shop Scheduling Problem (fJSP). The algorithm takes advantage of hybrid evolutionary algorithms that use particle swarm optimization and bayesian network structuring to learn the optimal relationships between machine, task, and objective function.

2 Motivation

The Job-Shop Scheduling Problem (JSP) is an NP-Hard problem in computer science[2]. Let us imagine we have a set of machines and a set of jobs to be completed. Each job is a set of operations and the operation order is fixed. In the classical formulation of JSP, each operation has a fixed processing time, and a required machine. However, these authors used the flexible job-shop scheduling problem (fJSP) that assumes that one machine can perform multiple kinds of operation, and as such, there is no fixed processing time [1]. There are several potential constraints we could use including: a job does not visit the same machines twice; there are no precedence constraints among operations of different jobs; operations can not be interrupted; etc.

Both JSP and fJSP are minimization problems, that is, we generally want to minimize the overall time to completion of all the jobs. As we stated before, there is no way to solve this problem in less than exponential time. Therefore, there are several algorithms we can use to approximate the solution. The authors combined Bayesian Optimization and Evolutionary Algorithms approaches to approximate a solution for the fJSP.

3 Algorithm

The Hybrid Evolution Algorithm displayed above takes as input the Job-Shop problem data and constraints, and returns an approximation of the best solution to the Flexible Job-Shop Problem. The variable t represents the t^{th} generation of the solution set - it is initialized to 0 on line 3. Next, the population p(t) is initialized to include a random set of candidate solutions to the problem. p(t) is further divided into subgroups in a random fashion. After the population has been initialized, the main **while** loop is entered, which will iterate until the the termination condition has been satisfied based on the problem constraints.

At the start of each loop iteration, the population is evaluated using particle swarm optimization (PSO). Compared to a genetic algorithm, a PSO is more likely to include the optimal solution in the problem space, so the result has a higher probability to be accurate[1]. The best solution for the population is kept, and if it is a valid solution, the population grouping is adjust according to the Bayesian Network for each subgroup. The loop then repeats, evaluating the data based on the new subgrouping of the population. When the loop termination condition is met, the best solution is returned.

Algorithm 1 Hybrid Evloution Algorithm

```
1: Input: Problem Data, Parameters
2: Output: Best Solution of S-fJSP
3: $t \leftarrow 0$
4: initialize population p($t$) by random solution candidates satisfying the constraints; get sub-p($t$)
   by randomly grouping
5: g($t$) $\leftarrow 0$
6: while not meeting termination condition do
7:     evaluate p($t$) by PSO with grouping mechanism and parameters adaptive
8:     get the training data set Data from each sub-p($t$)
9:     g($t$) $\leftarrow$g($t+1$)
10:    keep best solution of population $gbest(t)$ and personal best $pbest(t)$
11:    if g($t$) meets condition then
12:        select the best BN structure by $Data$
13:        readjust the grouping by BN structure for each sub-p($t$)
14:        g($t$) $\leftarrow 0$
15:    end if
16: end while
17:     return best solution of S-fJSP gbest(t)
```

4 Discussion

For the scope of this project, it will be most beneficial to only look at the Bayesian Grouping variant of the algorithm given the CCBhEA implimentation brought the most optimal results for all scalings of the problem size. It will also be helpful to keep the context of the project generalized to any grouping of machines that share a multitude of functions rather than applying the algorithm to one specific scenario. A task of primary interest will be to see if there are any ways to improve the current CCBhEA algorithm. There may be an time improvement optimization ready to be found within the structure of the update functions found in the while loop of the pseudocode.

References

[1] L. Sun, L. Lin, Y. Wang, M. Gen, and H. Kawakami, “A bayesian optimization-based evolutionary algorithm for flexible job shop scheduling,” *Procedia Computer Science*, vol. 61, pp. 521–526, 2015.

[2] R. Cheng, M. Gen, and Y. Tsujimura, “A tutorial survey of job-shop scheduling problems using genetic algorithmsi. representation,” *Computers & industrial engineering*, vol. 30, no. 4, pp. 983–997, 1996.