Heuristic Optimization : Implementation exercise 2

Nikita Marchant (nimarcha@vub.ac.be)

May 17, 2017

1 Introduction

This document is the report the implementation exercise 2 for the Heuristic Optimization course. The exercise asks to implement two stochastic local search algorithms for the the permutation flow-shop problem with the sum weighted completion time objective (also called PFSP-WCT).

The PFSP-WCT has not been thoroughly studied in the literature so i used some inspiration from papers studying the PFSP with flowtime that is more studied.

2 Algorithms

The first algorithm is an Iterated Local Search (ILS) inspired from [Pan and Ruiz, 2012] and the second is a genetic algorithm inspired from [Zhang et al., 2009].

2.1 Iterated Local Search

The Iterated Local Search is a stochastic meta-heuristic known to be applicable to multiple optimization problems [Gendreau and Potvin, 2010].

It consists of these steps :

1. Generate a initial solution
2. Apply a local search to the solution
3. When stuck in a local optimum, apply a perturbation
4. Apply a local search to the solution
5. Use an acceptance criterion for the perturbed then optimized solution
6. Go back to 3 or stop when a termination criterion is met.

2.1.1 Initialization method: $LR(x)$

The initialization method used in this implementation and in [Pan and Ruiz, 2012] is the $LR(x)$ heuristic introduced by [Liu and Reeves, 2001]. It consist in three steps :

1. Rank the jobs by their weighed sum of flowtime
2. Generate x solutions by inserting the job from position x at the front of the solution
3. Select the sequence with the minimum weighted flow time

2.1.2 Local search

The local search that was implemented is the iterated RZ (IRZ). The RZ uses a insertion neighborhood. It sequentially inserts each job at each possible position in the candidate solution (thus is in $O(n^2)$ complexity if n is the number of jobs) and keeps the best one.

The IRZ applies RZ until a local optimal solution is found (i.e. the RZ does not yield a better result).

2.1.3 Perturbation method

After finding a local optimum, a perturbation is applied to be able to escape the local optimum and extend the search space.

The perturbation method consists of γ random insertion moves : each move selects randomly a job and moves it to a random position.

2.1.4 Acceptance criterion

After finding a local optimum, we have to decide if we keep the solution. I chose to implement the simulated annealing as the criterion with $\lambda \cdot \frac{\sum_{j=1}^{n} \sum_{i=1}^{m} p_{ij}}{10mn}$ as a constant temperature.

2.1.5 Termination criterion

For both algorithm, the termination criterion is the time. For instances of size $n = 50$, i stop at 70 seconds and for $n = 100$ i stop at 200 seconds. Theses values were chosen as 100 times the runtime of the algorithms implemented in the first part.

2.2 Genetic algorithm

My implementation of a genetic algorithm is inspired form [Zhang et al., 2009] with some slight differences.

1. Generate a initial population
2. Generate a new generation with the crossover operator
3. Perturbate each chromosome with a given probability
4. Apply a local search on each chromosome
5. Merge the old and the new population and select the eligible chromosomes to be kept
6. Goto 2 or stop if the termination criterion is met

2.2.1 Generation of the initial population

The initial population consists of one chromosome generated with $LR(x)$ (see 2.1.1) and the other are generated as random permutation of the job set.

2.2.2 Crossover operator

2.2.3 Mutation operator

The mutation operator if the same as in 2.1.3 and if applied only with $P_m \cdot \sqrt{U+1}$ probability. U is defined as the number of generation since the last improvement to the global solution. This is not present in [Zhang et al., 2009] but helps to unstuck the algorithm when it is stuck in a local optimum since a long time.

2.2.4 Local search

The local search method is RZ (as explained in 2.1.2) but also as an original addition, if U is bigger than U_{IRZ}, a IRZ algorithm is used instead because the mutation rate is much higher so the chromosome is highly perturbed.

Using a IRZ all the time was tried but yielded no better results and was significantly slower[1].

2.2.5 Population selection

3 Parameters

3.1 Computation power limitation

3.2 Iterated Local Search

3.3 Genetic algorithm

4 Results

5 Implementation

6 Conclusion

References

[Gendreau and Potvin, 2010] Gendreau, M. and Potvin, J.-Y. (2010). *Handbook of metaheuristics*, volume 2. Springer.

[Liu and Reeves, 2001] Liu, J. and Reeves, C. R. (2001). Constructive and composite heuristic solutions to the p//$\sum$ ci scheduling problem. *European Journal of Operational Research*, 132(2):439–452.

[Pan and Ruiz, 2012] Pan, Q.-K. and Ruiz, R. (2012). Local search methods for the flowshop scheduling problem with flowtime minimization. *European Journal of Operational Research*, 222(1):31 – 43.

[Zhang et al., 2009] Zhang, Y., Li, X., and Wang, Q. (2009). Hybrid genetic algorithm for permutation flowshop scheduling problems with total flowtime minimization. *European Journal of Operational Research*, 196(3):869 – 876.

[1] Due to the lack of computing power (see 3.1), this hypothesis could not be formally proved.