MISTA 2009

Variable Neighborhood Descent methods for the Distributed Permutation Flowshop Problem

Rubén Ruiz • Bahman Naderi

1 Introduction

In the permutation flowshop problem (PFSP) a set N of n unrelated jobs are to be processed on a set M of m machines. These machines are disposed in series and each job has to visit all of them in the same order. This order can be assumed to be $1, \dots, m$ without loss of generality. Therefore, each job $j, j \in N$ is composed of m tasks. Once a processing order for the first machine has been determined, this order is to be maintained throughout all machines. There is a common and universal assumption in the PFSP literature: there is only one production center, factory or shop. This means that all jobs are assumed to be processed in the same factory. Nowadays, single factory firms are less and less common, with multi-plant companies and supply chains taking a more important role in practice (Moon et al., 2002). In the literature we find several studies where distributed manufacturing enables companies to achieve higher product quality, lower production costs and lower management risks (see Wang, 1997 and Kahn et al., 2004 for some examples). However, such studies are more on the economic part and distributed finite capacity scheduling is seldom tackled. In comparison with traditional single-factory production scheduling problems, scheduling in distributed systems is more complicated. In single-factory problems, the problem is to schedule the jobs on a set of machines, while in the distributed problem an important additional decision arises: the allocation of the jobs to suitable factories. Therefore, two decisions have to be taken; job assignment to factories and job scheduling at each factory. Obviously, both problems are intimately related and cannot be solved sequentially if high performance is desired.

In this paper we study the distributed generalization of the PFSP, i.e., the distributed permutation flowshop scheduling problem or DPFSP in short. More formally, the DPFSP can be defined as follows: a set N of n jobs have to be processed on a set F of f factories, each one of them contains the same set M of m machines, just like in the regular PFSP. All factories are capable of processing all jobs. Once a job $j, j \in N$ is assigned to a factory $f, f \in F$ it cannot be transferred to another factory as it must be completed at the assigned plant. The processing time of job j on machine i at factory f is denoted as $P_{j,i,f}$. We are going to assume that processing times do not change from factory to factory and therefore, $P_{j,i,1} = P_{j,i,2} = \dots = P_{j,i,f} = P_{j,i}$, $\forall i \in M$. We study the minimization of the maximum makespan among all factories and denote the DPFSP under this criterion as $DF/prmu/C_{max}$. The total number of possible solutions for the DPSP is $\binom{n-1}{F-1}n!$, much more than the regular n! solutions of the PFSP. Note that for reasons of space, we do not demonstrate this result here. Furthermore, for makespan minimization, no factory should be left empty with no jobs assigned. The demonstration for this is also omitted.

Rubén Ruiz
Grupo de Sistemas de Optimización Aplicada. Instituto Tecnológico de Informática (ITI). Ciudad Politécnica de la Innovación, Edificio 8G. Acceso B. Universidad Politécnica de Valencia. Camino de Vera s/n, 46022 Valencia, Spain.
E-mail: rruiz@eio.upv.es

Bahman Naderi
Department of Industrial Engineering, Amirkabir University of Technology, 424 Hafez Avenue, Tehran, Iran.
E-mail: bahman.naderi@aut.ac.ir

We present a simple Variable Neighborhood Descent (VND) method (Mladenovic and Hansen, 1997) for this problem, together with some heuristic methods. The choice of the VND is motivated by the simplicity of this approach, which is a desirable characteristic of solution methods. Our intention is to provide some initial results for this interesting problem generalization of the PFSP.

2 Variable Neighborhood Descent method

In Variable Neighborhood Descent (VND), several different neighborhoods are explored in order typically from the smallest and fastest to evaluate, to the slower and bigger one. The process iterates over each neighborhood while improvements are found, doing local search until local optima at each neighborhood. Only strictly better solutions are accepted after each neighborhood search. Our proposed VND incorporates one local search to improve the job sequence at each factory, and another one to explore job assignments to factories. In the following we further detail the proposed VND.

2.1 Initial solution

Starting VND from a random solution quickly proved to be a poor decision since the method rapidly converges to a local optimum of poor quality. Recall that VND is a very simple form of local search. Furthermore, effective heuristic initialization of metaheuristic algorithms is a very common approach in the scheduling literature. As a result, we initialize the proposed VND from an adaptation of the well known NEH method from Nawaz et al. (1983). Basically, we follow all steps of the NEH, the only difference is that each job in the insertion list is inserted in all positions of all factories and is finally placed in the position resulting in the minimum makespan among all factories.

2.1 Solution representation and local search

Solution representation is a key issue for every metaheuristic method. In the regular PFSP, a job permutation is the most widely used representation. Recall that the DPFSP adds the job factory assignment dimension and therefore, such representation is not enough. We propose a representation composed by a set of F lists, one per factory. Each list contains a partial permutation of jobs, indicating the order in which they have to be processed at that factory. Notice that this representation is complete and indirect. It is complete because all possible solutions for the DPFSP can be represented and it is indirect since we need to decode the solution in order to calculate the makespan.

For the proposed VND we have two different neighborhoods. The first one is the typical insertion neighborhood where all assigned jobs to one factory are extracted, one by one, and reinserted in all possible sequence positions at that factory. Moves are accepted as long as improvements in the makespan of a given factory are obtained. This process is repeated for all factories. Therefore, the first local search works over the job sequencing decision of the DPFSP. The second local search deals with the factory assignment dimension. All jobs are extracted from their currently assigned factories and ares inserted in all possible positions of all other factories. The acceptance criterion for this second neighborhood is more complicated. We propose two techniques: a) accept movements if the global makespan is decreased or b) accept movements if a “net gain” is achieved, i.e., we permit one of the two factories (origin or destionation) to increase its makespan value provided that the other factory decreases its makespan and that the net effect is negative (increased makespan is less than decreased makespan).

Both proposed neighborhoods are very fast to evaluate as the traditional PFSP accelerations for makespan calculations can also be used in the DPFSP.

3 Computational results

We evaluate both proposed VND methods (each one applying one of the two proposed movement acceptance rules for the second local search, i.e., VNDa and VNDb) over a comprehensive benchmark. The benchmark contains 720 instances being the maximum size 500 jobs, 7 factories and 20 machines per factory. We include in the comparison straightforward extensions of existing methods from the literature (extended for the DPFSP). We measure the relative percentage deviation of each method from the best solution known for each instance. All tests have been carried out in a Core 2 Duo computer running at 2.0 GHz. The average results (proven to be statistically different) are given in the following table:

	SPT	LPT	Johnson	CDS	PALMER	NEH	VNDa	VNDb
Deviation	16.79	29.93	10.55	8.20	8.85	1.03	0.10	0.43
Time (sec)	<0.001	<0.001	<0.001	<0.001	<0.001	0.026	0.147	0.096

Table 1 – Average results of the proposed NEH and VND methods, compared with standard heuristics from the PFSP literature.

As we can see, all three proposed methods, NEH, VNDa and VNDb provide very good results. The best proposed method is VNDa, which gives the best solutions on average. The CPU time employed is extremely short as it is only 0.147 seconds for the slowest proposed method.

4 Conclusions

To the best of our knowledge, this paper represents the first attempt at solving an interesting generalization of the permutation flowshop problem: the distributed permutation flowshop problem. Several identical factories are available to process the jobs and an additional decision of job factory assignment has to be carried out. We have presented an extension of the well known NEH heuristic and two simple Variable Neighborhood Descent search methods based on two simple neighborhood structures. The presented methods are very fast and provide much better results than typical heuristic approaches. Of course, much more work needs to be done in this interesting problem. Exact approaches are needed in order to gauge the maximum solvable instance size. More advanced metaheuristics could improve the results even further. Also, the assumption that all factories are identical is not realistic and varying processing times when changing from factory to factory should be investigated. Lastly, minimizing the maximum makespan across factories is a good start but more work needs to be done in order to determine other plausible objectives like minimizing the sum of makespans or even the standard deviation of the makespans across factories.

References

Mladenovic, N. and Hansen, P. (1997). “Variable neighborhood search”, Computers & Operations Research, 24 (11):1097-1100

Kahn, K. B., Castellion, G. A. and Griffin, A. (2004). The PDMA Handbook of New Product Development, Second edition, Wiley, New York.

Moon, C., Kim, J. and Hur, S. (2002). “Integrated process planning and scheduling with minimizing total tardiness in multi-plants supply chain”, Computers & Industrial Engineering, 43 (1-2):331-349.

Nawaz, M., Enscore, E. E. Jr., and Ham, I. (1983). “A Heuristic Algorithm for the m-Machine, n-Job Flow-Shop Sequencing Problem”, OMEGA, The international Journal of Management Science, 11(1):91-95.

Wang, B. (1997). Integrated product, process and enterprise design, Chapman & Hall, London.