International Journal of Engineering Applied Sciences and Technology, 2020
Vol. 5, Issue 6, ISSN No. 2455-2143, Pages 79-84
Published Online October 2020 in IJEAST (http://www.ijeast.com)

META-HEURISTIC PROCEDURE TO FIND THE SOLUTION OF JOB SHOP SCEDULING PROBLEM

Rituraj Jain
Department of Electrical and Computer Engineering
Wollega University
Nekemte – Ethiopia

Yohannes Bekuma
Department of Electrical and Computer Engineering
Wollega University
Nekemte - Ethiopia

Abstract - **In most production environment, there is a requirement for effective use of the available resources. Job Shop Scheduling Problem (JSSP) because of its key impact on revenues is at the crux of production planning, of late also known as Enterprise Resource Planning (ERP). Industrial tasks ranging from assembling cars to scheduling airplane maintenance crews are easily modeled as instances of JSSP, and improving solutions by even as little as one percent can have a significant financial impact. Efforts have been put to find not necessarily an optimal solution, but a good one to solve these problems.**

Tabu Search (TS) is an effective local search algorithm for the JSSP, but the quality of the best solution found depends on the initial solution used. The motivation behind such a paper is to layout critical discussion on the Tabu Search technique for JSSP, based on neighborhood structures for finding initial solutions, in order to better serve the needs of manufacturing community trying to apply Tabu Search technique on their problems. In this paper a approach is presented that uses a different methods, that either single scheduling or double scheduling, with functions of neighborhood selection in Tabu Search Method. The approach is tested on a set of standards instances taken from the literature and compared with other approaches. The computation results validate some effectiveness of the proposed algorithm.

Key Words - Job Shop Scheduling Problem, Local Search Method, Tabu Search

I. INTRODUCTION

Scheduling of resource in any economic setting is of great importance, and yet a computationally difficult and complex process. The origin of Job Shop Scheduling problem (JSSP) is to utilize the available resources as efficiently as possible in manufacturing environments. The Job Shop Scheduling Problem is a very important practical problem. Dimitri Golenko et al. (1995) stated Job Shop Scheduling Problem as, there are j jobs and m machines, each job comprises a set of operations which must be done on different machines for different specified processing times, in a given job dependent order. Efficient methods of solving it will have an important effect on profitability and product quality. But Job Shop Scheduling Problem is among the worst members of the class of NP-hard Problems which has many practical implications. Ouelhadj D (2003) mentioned best examples which are Production system and also optimization of the part of a steel work (any thing considered).

F. Glover (1989) and Moraglio et al. (2004) have defined Tabu Search (TS) in their studies as a local search method designed to find a near-optimal solution of combinatorial optimization problems. The oddity of Tabu Search is a short-term memory used to keep track of forbidden (tabu) recent solutions, thus allowing the search to escape from local optima. Taillard E. (1994) and Nowicki, E. et al. (1996) mentioned that Tabu Search has revealed to be an effective local search algorithm for the Job Shop Scheduling Problem.

The aim of this paper is to show the combined use of different neighborhood functions with two different methods that are Single scheduling and Double

International Journal of Engineering Applied Sciences and Technology, 2020
Vol. 5, Issue 6, ISSN No. 2455-2143, Pages 79-84
Published Online October 2020 in IJEAST (http://www.ijeast.com)

scheduling to get initial solution of Tabu Search in order to solve the Job Shop Scheduling Problem.

II. REPRESENTATION OF JSSP

Take an example of 3X3 Job Shop Scheduling Problem as mention in the TABLE I. The data includes the routine of the job through each machine and the processing time for each operation is mentioned in the parenthesis.

Table 1. 3 x 3 JSSP PROBLEM

JOB	Operation routing (Processing Time)		
1	1(3)	2(3)	3(3)
2	1(2)	3(3)	2(4)
3	2(3)	1(2)	3(1)

The first number in each column represents the machine M_i on which the operation has to perform. Fig 1 depicts one out of many feasible schedules for this problem instances. Yamada T et al. (1997) defined this kind of graphical representation as the Gantt chart of a schedule.

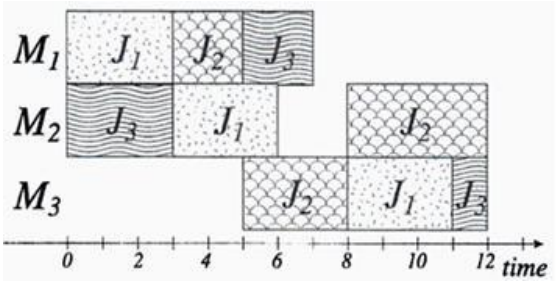


Fig 1. A Gantt –Chart representation of a solution for a 3 x 3 problem

Omar et al. (2008) described another way to represents the job shop problem using the concept of disjunctive graph. Consider a set of *n* jobs to be processed on *m* machines. Let disjunctive graph, *G* with a set of nodes *N* and two sets of arcs *A* and *B*. The nodes *N* represent the set of operations' node as well as two dummy nodes at the beginning (source node, *U*) and the end (sink node, *V*) of schedule. The weight of each node is the processing time for each operation. The source node and the sink node have zero processing time. The conjunctive (solid) arcs *A* represent the routes of each job including arcs connecting the source node to the first operation node and the last operation to the sink node. These The disjunctive (broken) arcs *B* refer to operation that belong to the different job but have to be processed on the same machine.

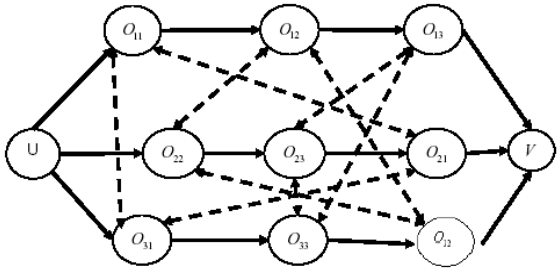


Fig 2. Disjunctive Graph for 3 Jobs and 3 Machines Problems

III. LOCAL SEARCH AND TABU SEARCH

Sampels M. et al. (2002) stated the basic local search algorithm to find a local optimum is called iterative improvement. Starting at some initial feasible solution, its neighborhood is searched for a solution of lower cost. If such a solution is found, the algorithm is continued from there, otherwise a local optimum has been found. Often, the problem remains. The local optima obtained may be of poor quality. Therefore, several variants of iterative improvement have been proposed. The main variants can be divided into threshold algorithms, Tabu Search algorithms, variable depth search algorithms and GA's.

The Tabu Search is a meta-heuristic approach designed to find a near-optimal solution of combinatorial optimization problems. This method suggested by Glover. Tabu Search can be briefly sketched as follows. In tabu Search one selects from a subset of permissible neighbors of the current solution, a solution of minimum cost. In basic Tabu Search a neighbor is permissible if it is not in the 'tabu list' or satisfies a certain 'aspiration criterion'. The tabu list is recalculated on each iteration. It is often implicitly defined in terms of forbidden moves from the current solution to a neighbor. The aspiration criterion expresses possibilities to overrule the tabu status of a neighbor.

The performance of a local search algorithm, both in terms of the quality of solutions, and in the time required to reach them is heavily dependent on the neighborhood structure. Formally, given a solution s a neighborhood is a set N(s) of candidate solutions, which are adjacent to s. This means that if currently examining solution is s the next solution examine will be some s' $\in$ N(s). Yamada T et al. (1997) mentioned that typically, the solutions in N(s) are generated from s with small, local modifications to s commonly called moves. Sampels M. et al. (2002)

International Journal of Engineering Applied Sciences and Technology, 2020
Vol. 5, Issue 6, ISSN No. 2455-2143, Pages 79-84
Published Online October 2020 in IJEAST (http://www.ijeast.com)

and Zhang et al (2007) mentioned that there are so many neighborhood structures like N0, N1, N2, NA, RNA, NB, N1a, N1b, N1c, and N4 can be used for local search method.

IV. PROPOSED APPROACH

The neighborhood function is the most important part of the tabu search algorithm, as it significantly affects both the running time and the quality of solutions. The neighborhood used in this implementation is one introduced by Dell' Amico et al. (1993), which they call NC. NC is the union of the neighborhoods RNA and NB. NC is connected because NB is, and NC is a smaller neighborhood than NA because each arc examined in NC leads to fewer than 5 possible adjacent moves.

The items placed on the tabu list are the reversed arcs, and a move is considered tabu, if any of its component arcs are tabu. This model is used because, in the case of neighborhoods, which may reverse multiple arcs, making only the move itself tabu would allow many substantively similar moves (i.e. those which share arcs with the tabu move) to be taken.

Single Scheduling is constructive heuristic, which examine a subset of operations and schedule these operations one at a time. These algorithms have the advantage of running in sub-quadratic time and producing reasonable result with any of a number of good priority rules. While Single Scheduling algorithms are no longer considered to be the best for solving large job shop instances, they can still produce good initial solutions for local search algorithms.

Double Scheduling is an extension of the basic single-scheduling framework. This algorithm take two lists in consideration for the first operation and the last operation of each job. The algorithm then alternates between lists, scheduling one operation and updating any necessary data each time, until all operations are scheduled. This algorithm tries to pass up a critical problem with single scheduling algorithms, namely that as they close to completion, most of the operations are scheduled poorly.

Additionally, the proposed double search chooses from the respective lists using a cardinality-based semi-greedy heuristic with parameter c, which means that the priority rule selects an operation uniformly at random from amongst the c operations with the lowest priority. This provides for a greater diversity of initial solutions which means that over several successive runs, a local search algorithm will explore a larger amount of total solution space than would otherwise be possible. In this implementation, the parameter c was set to 3.

The Tabu Search Framework as shown in Fig 3 shows the tabu search in its canonical form. There are two high-level goals for improving the quality of solutions. The first is to attempt to visit nearby improving solutions that would be unreachable. The second goal is to increase the total amount of the solution space the tabu search visits. The former tries to ensure that all nearby local optima are explored to find reasonable solutions quickly. The latter tries to find solutions close to a global optimum by visiting many different areas of the solution space.

One optimization critical to an efficient local search algorithm is the rapid computation of the value of a neighboring solution. Ideally it is possible to perform an exact evaluation quickly, but if this cannot be done, a good estimation will suffice. In the present problem, computing the exact value of the makespan for a neighboring solution is expensive. However, we can find the value of a reasonable estimation in time proportional to the number of arcs reversed by the move. That is, we can compute the value of the longest path through the affected arcs.

```
TS_JSSP_ Algorithm:
    sol = getinitialsol(JSSP)
    bestcost = getcost(sol)
    bestsolution = sol
    tabulist = 0
    while keepsearching()
    do N_val(sol) ={ s ∈ N(sol) | move[sol; s]
    ∉ tabulist }
        if N_val(sol) ≠ 0
            then sol' = x ∈ N_val(sol) | ∀y ∈
                N_val(sol) cost(x) ≤ cost(y)
        updatetabulist(sol')
        if cost(move[sol, sol'] ) < bestcost
            then bestSolution = sol'
                bestcost = cost(sol')
        sol = sol'
return bestsolution
```

Fig 3. Tabu_JSSP_Algorithm

International Journal of Engineering Applied Sciences and Technology, 2020
Vol. 5, Issue 6, ISSN No. 2455-2143, Pages 79-84
Published Online October 2020 in IJEAST (http://www.ijeast.com)

Another optimization important to the overall running time of a Tabu search algorithm is the implementation of the Tabu list. It is convenient to use this structure as an actual list. Another approach is to store a matrix of all possible operation pairs (i.e. arcs). A time stamp is affixed to an arc when it is introduced into the problem by taking a move, and the time-stamping value is incremented after every move. With this representation, the tabu list may be dynamically resized in constant time.
It is important for a tabu search algorithm to cover as much of the solution space as possible to increase the probability of finding a better solution. One particular problem to overcome is cycling amongst solutions.
Visiting the same solutions repeatedly wastes moves that could otherwise be leading the search to unexplored solutions. The tabu list prevents the algorithm from spinning in small, tight cycles by making recently visited solutions tabu. However, this cannot guard against cycles whose length is longer than the tabu list.

V. COMPUTATIONAL RESULTS

The results for the three problems (FT06, FT10, and FT20) are compared with results from results of Ventresca M. et al (2003).
Here,

BKS = Best Known Makespan
RTS = Results from Designed method
SGA = Simple Genetic Algorithm results by Ombuki et. al.
SA = Simulated Annealing results by Ombuki et. al.

Table 2. Comparison of the result from design method (using double scheduling) with the SGA and SA methods for FT06, FT10, FT20, ABZ05, ABZ06 problems

Problem	Size	BKS	RTS (using double scheduling)					SGA	SA
			N1	N2	NA	NB	RNA U NB		
FT06	6 x 6	55	55	55	55	55	55	55	55
FT10	10 x 10	930	930	935	930	935	937	994	988
FT20	20 x 5	1165	1175	1165	1173	1165	1165	1274	1230
ABZ05	10 x 10	1234	1234	1236	1234	1236	1236	-	-
ABZ06	10 x 10	943	943	943	943	943	943	-	-

Comparison with various Approaches for solving FT10, and FT20 problems collected from various research papers is shown in TABLE IV.

From this comparison it is cleared that with both of this problems FT10 and FT20 the designed method is very much closer to the best-known result for some of the neighbourhood functions. While N1 and NA neighbourhood functions gives the best results for FT10 problem, N2, NB and RNA U NB gives the best results for the FT20 problem.

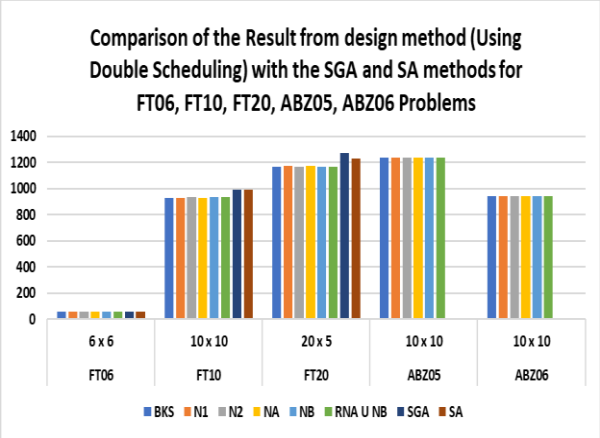
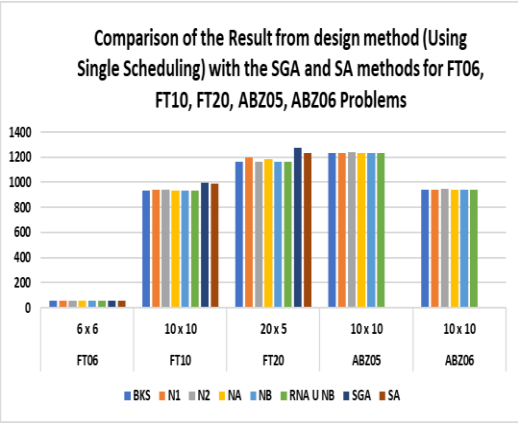


Fig 4 Comparison of the result from design method (using double scheduling)

Table 3. Comparison of the result from design

Problem	Size	BKS	RTS (using single scheduling)					SGA	SA
			N1	N2	NA	NB	RNA U NB		
FT06	6 x 6	55	55	55	55	55	55	55	55
FT10	10 x 10	930	937	940	936	935	935	994	988
FT20	20 x 5	1165	1199	1165	1185	1165	1165	1274	1230
ABZ05	10 x 10	1234	1234	1238	1236	1236	1236	-	-
ABZ06	10 x 10	943	943	945	943	943	943	-	-

method (using single scheduling) with the SGA and SA methods for FT06, FT10, FT20, ABZ05, ABZ06 problems



International Journal of Engineering Applied Sciences and Technology, 2020
Vol. 5, Issue 6, ISSN No. 2455-2143, Pages 79-84
Published Online October 2020 in IJEAST (http://www.ijeast.com)

Fig 5 Comparison of the result from design method (using single scheduling)

Table 4. Comparison of the result from design method (with both methods) with the other methods for FT10 and FT20 problems

Research		Best Solution for FT10	Best Solution for FT20
Croce		946	1178
Dornorf		938	1178
Fang		948	1165
Mattfeld		930	1165
Ono		930	1165
Storer et al.		954	1180
Dorndorf & Pesch		930	1165
Juels & Wattenberg		937	1174
Bierwirth		936	1181
Kobayashi		930	1173
Lin et al.		930	1165
Atlan		943	1182
Fang et al.		949	1189
Dorndorf & Pesch –Priority based GA		960	1249
Baker & McMohan		960	1303
Nakano & Yamada		965	1215
RTS (using double scheduling)	N1	930	1175
	N2	935	1165
	NA	930	1173
	NB	935	1165
	RNA U NB	937	1165
RTS (using single scheduling)	N1	937	1199
	N2	940	1165
	NA	936	1185
	NB	935	1165
	RNA U NB	935	1165

VI. CONCLUSION

The JSSP is to find the sequence of jobs on each machine in order to minimise a given objective function. The objective function most often used is to minimise makespan. The makespan of a schedule can be defined as the time elapsed from the beginning of processing until the last job has finished.
Some other objective functions that could be used include tardiness, earliness, and flow time. The difficulty of this problem makes it very hard for conventional search-based methods to find near-optima in reasonable time.

Local search method is defined to solve scheduling problems. One of them is Tabu Search (TS). The basic idea of Tabu search is to explore the search space of feasible scheduling solutions by a sequence of moves and choosing the best available. The elements of Tabu Search are short-term memory, tabu tenure, and aspiration criteria.

This work has demonstrated that it is possible to take existing tabu search algorithms and adjust them to provide reasonable solutions to a wider class of problems. As is evident from the data, the initial solution provided by the double scheduling algorithm is substantially poorer for the instances with sequence dependent setup times than for those instances without them. This is likely because the double scheduling algorithm does nothing to prevent large setup times on the machine arcs connecting the left and right halves. Even so, the Double schedule typically found better initial solutions that those found by the unidirectional single schedule tested.

VII. REFERENCES

1. Dimitri Golenko-Ginzburg, Shmuel Kesler and Zinoviy Landsman (1995) Industrial job-shop scheduling with random operations and different priorities, International Journal of Production Economics Volume 40, Issues 2-3, Pages 185-195
2. Ouelhadj D (2003) A multi-agent system for the integrated dynamic scheduling of steel production. PhD Thesis, Department of Computer Science, University of Nottingham, United Kingdom.
3. F. Glover, (1989) "Tabu search—part I", ORSA Journal on Computing, vol. 1, no. 3, pp. 190-206
4. Moraglio, Alberto & Eikelder, Huub & Tadei, Roberto. (2004). Genetic Local Search for Job Shop Scheduling Problem. Technical Report CSM-435 ISSN 1744-8050
5. Taillard E. (1994) Parallel tabu search techniques for the job shop scheduling problem, ORSA J. Computation, 6, pp. 108-177
6. Nowicki, E., & Smutnicki, C. (1996). A Fast Taboo Search Algorithm for the Job Shop Problem. Management Science, 42(6), 797-813
7. Yamada T, Nakano R (1997) Job-shop scheduling. Genetic algorithms in engineering systems. Cambridge University Press, New York, pp 134–160
8. Omar, Mahanim (2008) A Modified Multi-Step Crossover Fusion (MSXF) In Solving Some

Deterministic Job Shop Scheduling Problem (JSSP) [TS157.5. M214 2008 f rb]. Masters thesis, University Sains Malaysia.

9. Sampels M., Blum C., Mastrolilli M., Rossi-Doria O. (2002) Metaheuristics for Group Shop Scheduling. In: Guervós J.J.M., Adamidis P., Beyer HG., Schwefel HP., Fernández-Villacañas JL. (eds) Parallel Problem Solving from Nature — PPSN VII. PPSN 2002. Lecture Notes in Computer Science, vol 2439. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45712-7_61
10. Zhang, Chaoyong & Li, Peigen & Guan, Zailin & Rao, YunQing. (2007). A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem. Computers & Operations Research. 34. 3229-3242. 10.1016/j.cor.2005.12.002
11. Dell'Amico, Mauro & Trubian, Marco. (1993). Applying Tabu Search to the Job-Shop Scheduling Problem. Annals of Operations Research. 41. 231-252. 10.1007/BF02023076.
12. Ventresca M. and Ombuki B. M. (2003) Meta-heuristics for the Job Shop Scheduling Problem, CS-03-12, December 2003.