Academic Science Journal

Approximate Local Search Methods for Multi-objective Sequencing Problem

Adawiya A. Mahmood Al-Nuaimi and Seenaa J. Khamees*

Department of Mathematics - College of Science - University of Diyala

*seenaajabar164@gmail.com

Received: 27 August 2023 **Accepted: 24 September 2023**

DOI: https://doi.org/10.24237/ASJ.02.02.718B

Abstract

In the setting of a single machine, this study suggests approximation local search strategies to identify approximate solutions to the multi-objective sequencing problem, where the problem is the total of the four objectives: total completion time $\sum C_j$ j= 1,….,n, total lateness $\sum L_j$, maximum lateness L_{max} and maximum earliness E_{max}. Descent Method (DM), Simulated Annealing (SA), and genetic algorithm (GA) are three approximate local search techniques that are computer-implemented Matlab programs. On the basis of the outcomes of computing tests, conclusions are modeled on the effectiveness of the local search techniques.

Keywords: Local search, multi-objective, sequencing, genetic algorithm.

طرق بحث محلية تقريبية لمسألة ترتيب متعددة الهدف

عدوية علي محمود النعيمي و سيناء جبار خميس عبد الله

قسم الرياضيات - كلية العلوم- جامعة ديالى

الخلاصة

تم في هذا البحث التطرق الى طرق البحث المحلية التقريبية لايجاد الحلول التقريبية لمسألة ترتيب متعددة الهدف على ماكنة واحدة, حيث المسألة هي المجموع لأربعة أهداف وهي وقت الاتمام الكلي ($\sum C_j$) ، وقت التأخير الكلي($\sum L_j$) ، أعظم تأخير (L_{max}) وأعظم تبكير (E_{max}).

Academic Science Journal

طرق البحث التقريبية المحلية هي طريقة (DM)،(SA) و (GA) تمت برمجتها باستخدام Matlab على الحاسبة.
الاستنتاجات نمذجت على كفاءة طرق البحث المحلية بالاعتماد على نتائج التجارب الحسابية.

الكلمات المفتاحية: طرائق البحث المحلي، متعددة الاهداف، ترتيب، خوارزمية جينية.

Introduction

Many different application sectors have real-world problems, however they are often merely time-related [1]. Time restrictions are significant from two perspectives. They establish the conditions for feasibility and enable the evaluation of the effectiveness of feasible solutions. In the theory of sequencing, deadlines or due dates are typically used to create time constraints, and the quality of sequences is judged in relation to these factors. The informal objective (criterion) used by practitioners is made up of performance measurements with due dates. This makes this objective function a particularly alluring and extensively researched study topic. The literature on sequencing (scheduling) devotes a significant portion of its attention to traditional objective functions, like decreasing mean earliness-tardiness and the maximum lateness [3],[4]. Assigning a collection of tasks (jobs) to a group of machines in a timely manner while keeping in mind predetermined limits is referred to as the sequencing problem [4]. Tasks (jobs) j (j=1,2,...,n) have processing times (pj), due dates (dj), and estimated completion times ($C_j=\sum_{i=1}^{j} p_i$) that are specific to a given sequence of tasks (jobs).

The minimization of a linear function is the focus of the multi-objective (multi-criteria) problem that is taken into consideration of total completion time $\sum_{j=1}^{n} C_j$, total lateness $\sum_{j=1}^{n} L_j$, maximum lateness L_{max} and maximum earliness E_{max}. This problem falls under the category of concurrent multi-objective (multi-criteria) problems. In this class, two or more objectives (criteria) are taken into account at once. By branch and bound (BAB) method Sen and Gupta [5] solve the problem

$1//L_{max} + \sum_{j=1}^{n} C_j$.

The structure of this work is as follows: The formulation of the problem is found in Section 1. The methods for local search approximation are presented in Section .2 (Algorithms). Results

of computation experiments using local search approximation methods are provided in Section 4; conclusions are then offered in Section 5.

1. Problem Formulation:

Every task (job) j=(1,2,....n) is to be handled uninterruptedly on one machine that can handle only one task at a time. A set of n independent tasks (jobs) N=1,2,.....,n are provided for processing at time zero. for a specific order of the tasks, processing time p_j and deadline date d_j are required tasks (jobs), completion time $C_{\sigma(j)} = \sum_{i=1}^{j} p_{\sigma(j)}$, total completion time $\sum_{j=1}^{n} C_{\sigma(j)}$, the total lateness $\sum_{j=1}^{n} L_{\sigma(j)}$ can be computed where $L_{\sigma(j)} = C_{\sigma(j)} - d_{\sigma(j)}$, maximum lateness $L_{max(\sigma)} = \max\{L_{\sigma(1)}, L_{\sigma(2)}, \ldots, L_{\sigma(n)}\}$ and maximum earliness $E_{max(\sigma)} = \max\{E_{\sigma(1)}, E_{\sigma(2)}, \ldots, E_{\sigma(n)}\}$, where $E_{\sigma(j)} = \max\{d_{\sigma(j)} - C_{\sigma(j)}, 0\}$.

Sequencing the tasks (jobs) is the goal in order to make the four goals function $\sum_{j=1}^{n} C_j + \sum_{j=1}^{n} L_j + L_{max} + E_{max}$ is minimized. This difficulty represented by (P), can be expressed as follows:

$$\left.\begin{aligned} Z = \min_{\sigma \in S}\Big\{ & \sum_{j=1}^{n} C_{\sigma(j)} + \sum_{j=1}^{n} L_{\sigma(j)} + L_{\max(\sigma)} + E_{\max(\sigma)} \\ & s.t \\ & C_{\sigma(1)} = p_{\sigma(1)} \\ & C_{\sigma(j+1)} = C_{\sigma(j)} + p_{\sigma(j+1)} \qquad j = 1,2,\ldots,n-1 \\ & L_{\sigma(j)} = C_{\sigma(j)} - d_{\sigma(j)} \qquad j = 1,2,\ldots,n \\ & E_{\sigma(j)} \geq d_{\sigma(j)} - C_{\sigma(j)} \qquad j = 1,2,\ldots,n \\ & E_{\sigma(j)} \geq 0 \end{aligned}\right\} \ldots Problem.[P]$$

Where S stands for the collection of every sequence.

Finding a processing order for the tasks (jobs) on a single machine with the goal of minimizing the sum of total completion time $\sum C_j$, total lateness $\sum L_j$, maximum lateness (L_{max}), and maximum earliness (E_{max}) is the goal of problem (P). (i.e. to minimize the 1//F($\sum_{j=1}^{n} C_j + \sum_{j=1}^{n} L_j + L_{max} + E_{max}$).

2. Local Search Approximation Methods:

The same characteristic applies to local search techniques: they iteratively go from one workable option to another while scouring the search space. Many of these techniques get their inspiration from the natural world and investigate the area around potential solutions [1]. The way that local search strategies represent the problem, describe the area on that representation, and conduct searches within that neighborhood varies. Some techniques for local search have evolved. This suggests that the following definition is necessary:

Definition (1) [1]:

A pair (S,f) that has the function f as a mapping f:S→ R is an example of a combinatorial optimization problem. The solution set S is the set of all feasible solutions. Finding a global optimal minimum solution, or a $s^* \in S$ such that $f(s^*) \leq f(s)$, for all $s \in S$.

2.1 Resolution Representation [1]

The problem description affects how to formulate a solution. A permutation of the integers (1,....,n) is used to indicate a solution to a sequencing problem with n tasks (jobs).

Definition (2) [2]:

A mapping $N^*: S \rightarrow P(S)$ that specifies a subset $N^*(s)$ of S neighbors of each $s \in S$ is known as a neighborhood function N^*.

There are three conventional neighborhoods for permutation. They are described by applying certain movements to a number of tasks (jobs):

- **A.** Shift (Insert): This neighborhood is created by shifting a task (job) from position (1,2,3,4,5,6,7,8) in the sequence to a position before (left insert) or after (right insert), depending on the order in which it was originally placed. For instance, the numbers 1, 5, 2, 3, 4, 6, 7, and 8 are both neighborhoods.
- **B.** Interchange (swap): Change over two responsibilities that aren't always adjacent to one another. One neighboring sequence is (1,6,3,4,5,2,7,8).

C. Insert a block: Add a new position to the subsequence of tasks (jobs). An illustration of a neighbor is the list (1,4,5,2,3,6,7,8).

Definition (3) [1]:

Let (S,f) serve as an illustration of a combinatorial optimization problem, and let N* serve as a neighborhood function. If $f(s^*) \leq f(s)$ for all $s \in N^*(s^*)$, then a solution $s^* \in S$ is said to be a local optimal (minimum) solution with regard to N*. If any local minimum with regard to N* also counts as a global minimum, the neighborhood function N* is said to be exact.

2.2 Descent Method (DM): [1]

Only improving moves are permitted in the straightforward neighborhood search technique known as the Descent Method. A local optimum, not necessarily a global optimum, is the outcome.

The following are the main elements of a descent method:

A. Initialization:

As with the current solution, the initial set of workable solutions is produced at random, according to some established rule, or both. The objective function's value is computed in the current solution.

B. Neighborhood generation:

To choose a neighbor s' of s, a move is done from neighbor to neighbor via the solution space S.

C. Criteria (objective) for termination:

Up until a set of termination requirements are met, the algorithm is repeated. The best created solution will be the result.

2.3 Algorithm of Simulated Annealing (SA):

Simulated annealing (SA) has its roots in statistical physics, where the term "annealing" refers to the process of gradually cooling materials until they reach a low energy state. [2] And was first utilized by Kirkpatrick et al. for combinatorial optimization problems. [1]. such an approach does not require the goal function values to decrease monotonically. A starting sequence s is used to generate (usually randomly) a neighbor s' in a particular neighborhood.

The difference between the values of the objective function F is therefore determined as $\Delta = F(s') - F(s)$. Sequence s' is accepted as the first solution for the new iteration when $\Delta <= 0$. Sequence s' is accepted as a new starting solution in the event that $\Delta > 0$ with probability exp $-\Delta/T$, where T is a metric associated with temperature. In the initial stages, the current temperature is often high, making it relatively simple to escape from a local optimum.

Following the creation of a collection of sequences, the temperature typically decreases. A geometric cooling strategy is frequently used for this, and we will use it as well. In this instance, the new temperature T^{new} has been selected so that $T^{new} = \lambda T^{old}$, where T^{old} and $0 < \lambda < 1$ stand for the previous temperature and T^{new} for the new temperature, respectively. A cycle with a final temperature that is sufficiently near to zero would then be a potential halting criterion and may be employed as such. We determine T = 10 based on the starting temperature, much like in [2]. The algorithm for the simulated annealing approach is shown in Figure (1).

Step (1): Pick a beginning temperature of $t_0>0$, an initial solution of $s \in S$, and $s^* = s$ with K=0 and G=1**.**

Step (2): Defining B: $P(\Delta,t_k) = \exp(\Delta-/ t_k)$; select $s' \in N^*(s)$; $\Delta = f(s')- f(s)$;

s = s' if $\Delta \leq 0$, otherwise ($\Delta > 0$)**;**

With G=G+1 and $[0,1] \leq p(\Delta,t_k)$,

Step (3): If $G \leq B$ go back to step (2),

Step (4): Repeat step (2) up until a certain stopping condition is met;

k = k+1; update the temperature.

Figure 1

2.4 Genetic Algorithm (method) (GA): [1]

The (GA) technique used by genetic algorithms uses a population of solutions, each of which is represented as a string. Typically, a population is used to represent the solution space. By using basic genetic operators like cross-over, mutation, and select, new structures can be created. Similar to Darwin's theory of "survival of the fittest," members of the existing population with higher fitness values, i.e., better objective function values, will have a greater chance of being selected as parents. The optimality of the final solution cannot be guaranteed since the initial population is generated at random. Because of this, the objective function of our problem is included using select, cross-over, and mutation, to build new populations and save the best solution in each generation. At least one solution with the shortest length must thus be included in the original population. From the stored solutions, the top choice becomes the GA solution. The fitness value of a solution is a vector of function values. The best answers from the present population are chosen to create a parent. After applying the genetic operators to each new offspring, we obtain a new population by selecting solutions with high fitness values within each population and recombining them. It's important to remember that the mutation operation, for instance, relies on the pairwise swapping of two tasks in the pertinent sequence. Genetic algorithms (GA) have a number of applications that are frequently used in a variety of fields. They are used in fields including business, science, and engineering, such as:

(A complex function system's optimization classifier systems, machine learning, pattern recognition, error diagnosis, partitioning objects and graphs, self-adapting systems, clustering, design, and process control). The use of (GA) in a wide range of optimization tasks (jobs), including numerical optimization and combinatorial optimization problems like shop-job sequencing, has been demonstrated by researchers [7].

The basic elements of a genetic algorithm are as follows:

A. Initialization:

Chromosomes are initially generated as the first population. In order to come as close as possible to the optimal solution, sequencing heuristic dispatching rules heuristics methods are mixed with random methods to create the initial population of chromosomes.

B. Evolution (Fitness):

When assessing chromosomal fitness, the objective function is applied. When a population is created, each chromosome is evaluated, and its fitness is determined for each chromosome. Each chromosome is then given a fitness value based on the population size.

C. Selection:

When chromosome parents are chosen from the population to combine to create new chromosome offspring using selection techniques often based on fitness value, natural selection of some chromosomes occurs.

D. Crossover:

In order to create one or more child chromosomes, a crossover operator combines components from two parent chromosomes.

E. Mutation:

In order for other operators to continue working, a mutation operator must make sure that a population's variety is preserved.

F. Termination:

Traditionally, a method ends when a predetermined number of generations (or iterations) have been completed. For instance, Chen et al. [6] employ 20 generations since they see that the solutions stabilize after 20 generations**.**

3. Test Problems and Computing Results for the Local Search Approximation Algorithms:

By creating local search algorithms in Matlab and running them on a desktop computer hp with 32 GB of RAM, they are put to the test. The following is how test issues are generated: the discrete uniform distribution is used to generate an integer processing time p_j for each task (job) j [1,10]. Additionally, an integer due date is created from the discrete uniform distribution for each task (job) j[P(1-TF-RDD/2), P(1-TF+RDD/2], where P= $\sum p_j$, j= 1,…,n, depending on the average tardy factor and the relative range of due dates (RDD) (TF). The values 0.2,0.4,0.6,0.8,1.0 are taken into consideration for both parameters. In order to provide 10 issues for each value of n, two problems are generated for each of the five values of the parameters.

The following tables compare computational results and display the problem's time (in seconds) (P). When a problem cannot be optimally solved in the allotted 1800 seconds, computation is stopped for that problem. In each of these tables, we have:

Ex: Number of example.

No. of best: The number of samples that best capture the value.

Best= The value of best.

DM: The result of using the descent method.

SA: The result of using the simulated annealing technique.

GA: Value calculated using a genetic algorithm.

Time: Seconds of time.

Av. Time: The typical time it takes to solve a problem.

Table 1: The outcomes of local search techniques for n= 50

The no. Of examples	The Best	The DM	The Time For The DM	The SA	The Time For The SA	The GA	The Time For The GA
1	7920	7937	0.06632	7920	0.07090	7932	0.14212
2	6448	6455	0.07234	6448	0.06797	6450	0.14501
3	5599	5600	0.06777	5599	0.06832	5600	0.14045
4	5642	5654	0.06618	5642	0.06835	5654	0.14470
5	6752	6755	0.16673	6752	0.07002	6755	0.36496
6	7723	7723	0.16964	7725	0.17393	7723	0.36207
7	6836	6856	0.08112	6836	0.17214	6851	0.14153
8	10165	10165	0.06737	10165	0.06916	10165	0.14090
9	8615	8615	0.06722	8615	0.06717	8615	0.14360
10	8455	8455	0.06611	8455	0.07003	8455	0.14177
The number of the Best		5	Av. Computation Time 0.08908	9	Av. Computation Time 0.089799	4	Av. Computation Time 0.186711

In this table, the number of examples that gives the best-known solution yet is 9 for the SA, 5 for the DM and 4 for the GA.

Table 2: The outcomes of local search techniques for n= 100.

The no. of examples	The Best	The DM	The Time for the DM	The SA	The Time For The SA	The GA	The Time For The GA
1	27651	27653	0.07792	27655	0.08219	27651	0.17906
2	16408	16408	0.07854	16433	0.08231	16408	0.17953
3	28973	28973	0.07868	28992	0.08008	28973	0.18074
4	36303	36303	0.07786	36303	0.08027	36303	0.18225
5	24155	24155	0.07872	24161	0.08112	24155	0.17624
6	26188	26188	0.07931	26188	0.07970	26188	0.17851
7	27120	27120	0.18440	27134	0.08169	27120	0.45086
8	37406	37406	0.20073	37406	0.20227	37406	0.37199
9	40188	40188	0.07599	40188	0.07647	40188	0.17853
10	43290	43290	0.07477	43290	0.08270	43290	0.17552
The number of the Best		9	Av. Computation Time 0.100692	5	Av. Computation Time 0.09288	10	Av. Computation Time 0.225323

Table 3: The outcomes of local search techniques for n= 500

The no. of examples	The Best	The DM	The Time For The DM	The SA	The Time For The SA	The GA	The Time For The GA
1	560861	560861	0.17299	560911	0.43220	560861	0.61350
2	712876	712876	0.17343	712876	0.17614	712876	1.13587
3	752539	752539	0.31933	752539	0.43956	752539	0.60943
4	621749	621749	0.17394	621749	0.17648	621749	0.95317
5	549242	549242	0.40099	549242	0.44457	549242	0.61465
6	577150	577150	0.17282	577195	0.17700	577195	0.61730
7	895565	895565	0.16987	895565	0.17375	895565	0.62243
8	943103	943103	0.16996	943103	0.17297	943103	1.38676
9	760656	760656	0.17163	760663	0.27225	760656	0.61392
10	725156	725156	0.17164	725156	0.17331	725156	1.28195
The number of the Best		10	Av. Computation Time 0.20966	7	Av. Computation Time 0.263823	9	Av. Computation Time 0.844898

Table 4: The outcomes of local search techniques for n= 1000.

The no. of examples	The Best	The DM	The Time For The DM	The SA	The time for The SA	The GA	The Time For The GA
1	2801104	2801104	0.29577	2801104	0.29906	2801104	1.63435
2	2575076	2575076	0.29666	2575076	0.29480	2575076	1.60130
3	2743618	2743618	0.29477	2743627	0.29306	2743618	2.55611
4	2107473	2107473	0.29696	2107524	0.29870	2107503	2.47588
5	2560011	2560011	0.29687	2560011	0.29400	2560011	2.47478
6	2994871	2994871	0.29443	2994871	0.29233	2994871	1.61614
7	3056426	3056426	0.73445	3056483	0.50551	3056462	1.89041
8	2991250	2991250	0.73034	2991250	0.68687	2991250	1.60288
9	3309458	3309458	0.72029	3309458	0.70827	3309458	1.79151
10	3367007	3367007	0.57812	3367012	0.73290	3367007	1.95749
The number of the Best		10	Av. Computation Time 0.453866	6	Av. Computation Time 0.44055	9	Av. Computation Time 1.960085

Table 5: The outcomes of local search techniques for n= 5000.

The no. of examples	The Best	The DM	The Time for The DM	The SA	The Time for The SA	The GA	The Time for The GA
1	55552224	55552224	1.74018	55552231	2.15420	55552224	25.08475
2	62001099	62001099	1.72398	62001099	1.77613	62001099	25.20003
3	75440363	75440371	2.02494	75440384	2.01447	75440363	22.10149
4	69577865	69577865	2.03075	69577865	2.03423	69577865	22.06904
5	70268664	70268664	1.96673	70268664	2.01504	70268664	22.00144
6	67867834	67867840	2.00973	67867840	2.00693	67867834	21.94663
7	90066012	90066012	2.02987	90066012	1.97751	90066012	21.91440
8	68377879	62096879	2.00343	62096883	2.01404	62096879	21.81086
9	68377875	68377875	1.98116	68377875	1.99280	68377875	21.80032
10	68479487	68479487	1.96784	68479494	1.98464	68479487	21.76059
The number of the Best		8	Av. Computation Time 1.947861	5	Av. Computation Time 1.996999	10	Av. Computation Time 22.56896

Table 6: The outcomes of local search techniques for n= 10000.

The no. of examples	The Best	The DM	The Time for The DM	The SA	The Time for The SA	The GA	The Time for The GA
1	277714113	277714113	3.68264	277714115	4.06225	277714113	82.95882
2	274565042	274565042	4.27113	274565042	4.41646	274565042	82.57181
3	245911784	245911784	4.24846	245911784	4.47899	245911784	83.76724
4	194436349	194436349	3.27996	194436354	3.96797	194436349	93.5343
5	269453874	269453874	3.20907	269453874	3.81978	269453874	90.66106
6	327322901	327322901	3.08131	327322901	3.19360	327322901	88.45110
7	299242189	299242189	3.11009	299242203	3.18514	299242189	88.70884
8	245717939	245717939	3.38126	245717939	3.29688	245717939	87.63161
9	390088690	390088690	3.05294	390088690	3.09325	390088690	88.17397
10	355770818	355770818	3.62001	355770818	3.16602	355770818	88.26580
The number of the Best		10	Av. Computation Time 3.493687	7	Av. Computation Time 3.668034	10	Av. Computation Time 87.47246

Table 7: The outcomes of local search techniques for n= 20000.

The no. of examples	The Best	The DM	The Time For The DM	The SA	The Time For The SA	The GA	The Time For The GA
1	767980793	767980793	7.37607	767980807	7.84754	767980807	324.35400
2	783823353	783823367	8.80861	783823367	11.89398	783823353	333.90831
3	992696008	992696008	7.34686	992696008	7.65371	992696008	342.29557
4	1211307475	1211307482	7.20560	1211307489	7.41542	1211307475	346.06200
5	1089667571	1089667571	7.25666	1089667585	8.36303	1089667571	340.11538
6	873675987	873675987	7.33182	873676001	7.72568	873676001	335.22102
7	996994585	996994585	7.27201	996994606	8.37318	996994599	338.24310
8	1004984235	1004984235	7.36034	1004984240	7.69322	1004984235	344.56415
9	1310890799	1310890799	7.11029	1310890799	7.35092	1310890799	338.39672
10	1542283660	1542283660	6.84256	1542283660	7.41578	1542283660	335.12469
The number of the Best		8	Av. Computation Time 7.391082	3	Av. Computation Time 8.173246	7	Av. Computation Time 337.8285

Table 8: The outcomes of local search techniques for n= 30000.

The no. of examples	The Best	The DM	The Time For The DM	The SA	The Time For The SA	The GA	The Time For The GA
1	1747913811	1747913811	10.93307	1747913818	16.90783	1747913811	602.00562
2	1989074131	1989074131	11.26824	1989074138	11.70550	1989074131	601.02725
3	2236432668	2236432668	11.18756	2236432668	11.43571	2236432668	600.09888
4	1725315017	1725315017	11.34028	1725315017	11.76335	1725315017	601.59488
5	2494579121	2494579121	11.04908	2494579121	12.10659	2494579121	603.38197
6	2967199802	2967199802	10.81800	2967199802	10.98016	2967199802	11156.90415
7	3204035293	3204035293	15.84867	3204035293	14.82118	3204035293	602.01389
8	2955179690	2955179690	13.19660	2955179690	14.97962	2955179690	602.37626
9	3203406382	3203406382	14.85260	3203406382	11.73867	3203406382	601.17884
10	2713185244	2713185244	10.95600	2713185244	11.39220	2713185244	601.14717
The number of the Best		10	Av. Computation Time 12.14501	8	Av. Computation Time 12.783021	10	Av. Computation Time 1657.1729

The number of examples in this table that provide the best-known answer is 10 for DM, 10 for GA, and 8 for SA.

Table 9: The outcomes of local search techniques DM and SA for n= 350000.

The no. of examples	The Best	The DM	The Time For The DM	The SA	The Time For The SA
1	1996700402	1996700402	13.91202	1996700409	15.16962
2	2041634896	2041634896	13.69903	2041634896	14.15247
3	2361445156	2361445156	14.65608	2361445163	13.92254
4	3026373455	3026373455	13.86546	3026373455	13.60440
5	4031878003	4031878003	13.42641	4031878003	13.35087
6	3381603739	3381603739	13.60944	3381603746	13.75502
7	3004499806	3004499806	13.64750	3004499806	13.84152
8	3033947109	3033947109	13.53441	3033947109	13.93719
9	4732380331	4732380331	11.92322	4732380331	12.35146
10	4409310000	4409310000	18.65207	4409310000	13.04744
The number of the Best		10	Av. Computation Time 14.092564	7	Av. Computation Time 15.16962

Table 10: The outcomes of local search techniques for n= 400000.

The no. of examples	The Best	The DM	The Time For The DM	The SA	The Time For The SA
1	2652178337	2652178337	15.34472	2652178337	16.14349
2	2636034634	2636034634	15.31169	2636034639	15.56658
3	3530067135	3530067135	15.10108	3530067135	15.25890
4	4403968539	4403968539	14.88985	4403968544	14.88951
5	4807864438	4807864438	14.79516	4807864452	14.79010
6	5307349326	5307349326	14.68639	5307349326	14.56266
7	4432556044	4432556044	14.83017	4432556044	14.89085
8	4394779517	4394779517	14.94245	4394779517	14.97891
9	4836858819	4836858819	14.85552	4836858819	15.00056
10	5743594531	5743594531	19.40139	5743594531	14.52033
The number of the Best		10	Av. Computation Time 15.415842	7	Av. Computation Time 15.060189

GA stops for n=35000 and 40000 in tables (4.9) and (4.10), respectively. DM is able to come up with the ideal answer. SA and DM have similar average computation times.

Conclusions

On this article, approximate solutions to the issue of sequencing multi-objective$\sum C_j + \sum L_j + L_{max} + E_{max}$, where tasks (jobs) j= 1, 2,..., n, are found using approximate local search

methods such as the descent method (DM), simulated annealing (SA), and genetic algorithm (GA).

On a sizable number of test problems, computation results for approximation local search strategies are shown.

From our results, the main conclusion is that DM and GA is better than SA for n= 50, 100, 500, 1000, 5000, 10000, 20000, 30000 in tables (4.1),….,(4.8). DM takes about as long to compute on average as SA.

For GA the average computation time increases when n increases.

For n = 35000, 40000 DM is better than SA. While the average computation time is convergent.

References

1. A. I. Khamees, Solving Multiobjective Sequencing Problem on one Machine,Thesis, University of Diyala, College of Science, Dept. of Mathematics, (2022)
2. A. A. M. Al-Nuaimi, Local Search Algorithms for Multiobjective Scheduling Problem, Journal of Al Rafidain University College, 36,201-217(2015)
3. P. Bruker, Scheduling algorithms, (Berlin-Heidelberg- New York, Springer, 2007)
4. M. Pinedo, Scheduling: theory, algorithms and systems, (New York: Springer, 2008)
5. T. Sen, S. K. Gupta, A branch and bound procedure to solving a bicriterion scheduling problem, IIE Transactions, 15,84-88(1983)
6. C. L. Chen, V. S. Vempati, N. Aljaber , An application of genetic algorithms for flow shop problems, European Journal of operation research, 80,389-396(1995)
7. A. S. Al-Tameemi, A. A. M. Al-Nuaimi, Exact and Local Search Algorithms to Minimize Multicriteria Scheduling Problem, Turkish Journal of computer and Mathematics Education, 12, (1) 7821-7831(2021)