An epistatic study on fitness landscape of Job Shop Scheduling problem

Project Report Submitted in Partial Fulfilment of the Requirements for the Degree of

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

Aditi Biswas Purba : (Roll No. 2016UGCS096)

Under the Supervision of

Dr. Vinay Kumar
Assistant Professor

Department of Computer Science and Engineering
National Institute of Technology Jamshedpur

Department of Computer Science and Engineering
National Institute of Technology Jamshedpur
Jamshedpur, India – 831014

CERTIFICATE

This is to certify that the work contained in this report entitled **"An epistatic study on fitness landscape of Job Shop Scheduling problem**" is submitted by the group members Ms. Aditi Biswas Purba (Roll. No: 2016UGCS096) to the Department of Computer Science and Engineering, National Institute of Technology Jamshedpur, for the partial fulfilment of the requirements for the degree of **Bachelor of Technology** in **Computer Science and Engineering**.

They have carried out their work under my supervision. This work has not been submitted elsewhere for the award of any other degree or diploma.

The project work in our opinion, has reached the standard fulfilling of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering in accordance with the regulations of the Institute.

Date:

(Supervisor(s))
Dept. of CSE
NIT Jamshedpur

(HoD)
Dept. of CSE
NIT Jamshedpur

Department of Computer Science and Engineering
National Institute of Technology Jamshedpur
Jamshedpur, India – 831014

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed. We will be solely responsible if any kind of plagiarism is found.

Date:

Ms. Aditi Biswas Purba
Roll. No: 2016UGCS096

Acknowledgement

I would like to place on record our deep sense of gratitude to Dr. Vinay Kumar, Department of Computer Science and Engineering, NIT Jamshedpur, India for his generous guidance, help and useful suggestions. I express my sincere gratitude to Dr. (Mrs.) Rashmi Sinha, HoD in Department of Computer Science and Engineering, Institute of Technology Jamshedpur, for his stimulating guidance, continuous encouragement and supervision throughout the course of present work. I also wish to extend our thanks to Mr. Aditya Hati for his insightful comments and constructive suggestions to improve the quality of this research work.

Thank you all.

Ms. Aditi Biswas Purba

Abstract

Job shop scheduling has risen in importance due to the demand of IT industries, over the last fifty years research into JSSP. It gained much popularity as researchers have worked to generate optimal solutions or near to that solutions which fits the flexibility of the difficult machine or job scheduling problem. As it is considered as a combinatorial problem, researchers' main motive is to find a job sequence with the minimum makespan or minimum fitness values from the machine sequences where many approaches like "Branch and Bound" method, "Genetic algorithm" and neighbourhood methods have presented solutions with less computational effort. Here, we have solved the JSSP with a new random walk algorithm where we are analyzing results in the fitness landscape through epistatic measures for each solution of the JSSP.

Contents

List of Tables

List of Figures

Chapter 1

1. Introduction :

Many planning and scheduling problems of industries nowadays seem to have their roots in Combinatorics. Usually, it is very hard to solve these types of problems in their general pattern. "Scheduling" is known as a technique of generating an optimal queue to execute a non-infinite set of operations that satisfies various constraints. As the formulated problem comprises different concurrent goals and various resources, it is very hard to solve. Concurrent goals like maximizing the utilization of each machine and minimizing the total time needed for completing the whole scheduled process, are the main goals. Scheduling can also be said as a method that is used for assigning the jobs or works to the resources or machines through some dispatching rules. The aim is to minimize the makespan by allocating the tasks or jobs into machines or resources within the given time. These resources and tasks can be of different forms. Here the objective will be to minimize the total completion time,meaning the makespan of each job[1]. By dynamic programming, branch and bound method, finding an optimal solution takes more time. So, we cannot expect a proper optimal solution but a nice solution that is near the optimal solution. To achieve our motive i.e. finding near-optimal solutions for a large range combinatorial optimization problems, modern search methods such as simulated annealing (Kirkpatrick, 1983) or tabu search (Glover, 1993), genetic algorithms can be helpful.

The JSSP has become the most attracting, popular scheduling problem to the researchers as it has mass aptness and implicit drawbacks. In the N×M classical JSP, M machines must process N jobs, where every job will have its own processing time and waiting time. Each job has a pre-specified, fixed processing sequence on the machines. N jobs with different routing orders will be processed on M machines. Hence, the complexity of scheduling is figured on the count of machines and jobs, orders of jobs. There are N! ways to create job permutation on each machine that can be calculated by various algorithms, methods. Scheduling problems with similar sequences for each machine have N! Solutions. On the other hand, scheduling problems with different sequences for each machine have $(N!)^M$ solutions. For solving the JSSP, many algorithms had been developed. In-Chan Choi [10] intended to build an algorithm based on local search that will reduce the fitness value where a condition depends on the orders of jobs or machines is added. This algorithm assists to decrease computation time of each job. Data mining was used by D.A. Koonce [11] to discover the relevance of sequence and augur the following job in the queue, also the pattern of scheduling for the JSSP. Data mining may explore the pattern of sequence, generate good solutions by summarizing dispatching rules. To achieve a minimized total holding cost and reduced computation time, Hiroshi [12] used a shift bottleneck method by adding no tardy job constraint. In job shop scheduling, time lag is the maximum and minimum between arrival times of operations. To solve the time lag JSSP set on a graph with partitions, Anthony [13] claimed a Memetic algorithm. Jobs may have controllable processing time. This kind of JSSP has been solved by Jansen [14].

Difficulties of JSSP can be investigated with landscape analysis where many approaches have been taken. An approach is to evaluate the ruggedness of landscape from the calculation of the auto-correlation of solution quality which can be spawned by a random walk analysis for characterizing fitness landscape structure [2,9]. The regularity of the search space means every element of the landscape must be looked over by a random walk having the same possibility, is a prerequisite of this approach . Many noticeable optimization problems, like the flow shop scheduling problem [5], the traveling salesman problem [3,4] reflect the regular search spaces. Particularly the problem with non-blocking or waiting constraint, can be formulated by the alternative graph and the pilot method with iterative optimization [8].

1.1 Job Shop Scheduling Problem :

Job shop scheduling is a global optimization problem in the operating system where multiple sets of jobs are executed on different machines. We can minimize the total time from the first job being scheduled to the execution of the last job. All the jobs consist of some tasks that should be processed in a particular machine in a given order. Shortest time processing time rule is applied here with the earliest due date method and first out first out method. Basically, we are given N jobs and M machines and N jobs will be executed in a specific sequence in M machines, then a makespan will be created for that particular set. Makespan is the total time for the scheduling, which means the total time of finishing all the jobs properly. In case of just two machines, the optimal orders of jobs with Johnson's rule, can decrease the makespan.
There are some constraints for JSSP:

- Till the task of a particular job is not completed, no task for that job can be started.
- One machine can not process more than one job at a time.
- A job or a task of any job, once started must be completed.

JSSP may be dynamic or static, where in static job shop scheduling no of jobs that we are going to perform will be known and in dynamic job shop scheduling jobs can enter into the machine anytime with no makespan.

1.2 Factors of the Job Shop Problem :

Following are some variables or factors in Job Shop Scheduling that have an important effect :

1. Arrival pattern
1. Number of machines (work stations)
2. Work Sequence
3. Performance appraisal standard

The arrival matrix of jobs to machines are of two forms.

1. Static : n jobs arrive at an idle machine and should be scheduled for execution
2. Dynamic : Alternate arrival

Work sequences can be random, fixed, repeated.

1. Random sequence — All patterns are possible
2. Fixed, repeated sequence : flow shop

The performance criteria depends on following optimal job scheduling heuristics.

1. Makespan : total time of the execution of all the jobs
2. Average number of jobs in the matrix
3. Average time of jobs in matrix
4. Machine utilization
5. Workers Lateness utilization

1.3 Types of Job Shop Scheduling Algorithm :

This algorithm is a process of determining which job will enter into which machines for execution according to their sequences and processing times in respective machines. The main motive is to minimize the makespan or generate such sequence for each machine which will give less computation time. Job shop scheduling problems can be of two types.

1. Preemptive scheduling : There will be a priority list by which any job or machine can be resumed for some time until the other job or machine has performed the task.
2. Non-preemptive scheduling : Here, the machine has been allocated to a particular job where the machine can not stop processing until its completion. No timer is required for this type of scheduling.

There are six popular job scheduling algorithms

1. First-Come, First-Served (FCFS) Scheduling
2. Shortest-Job-Next (SJN) Scheduling
3. Priority Scheduling
4. Shortest Remaining Time
5. Round Robin(RR) Scheduling
6. Multiple-Level Queues Scheduling

Following algorithms are either non-preemptive or preemptive. The preemptive scheduling depends on the priority queue. The scheduler may forestall a running job or machine with low priority anytime when a job or machine with high priority enters into the queue, whereas a non-preemptive algorithm is invented to complete the allotted time of job in a machine at a time without any preemption of any other jobs.

1.3.1 First Come First Serve (FCFS) :

First come first serve is one of the operating system scheduling algorithms that executes requests automatically from the queue and the requests are processed according to the arrival. Each job is executed on a first come, first serve basis and it can be implemented in FIFO queue. As it is a non-preemptive JSP algorithm, so after a job has entered into a machine, it will never leave until it's processing time finishes. It gives poor performance as waiting time is high.
Five jobs which are being processed on six machines in this study. All the jobs are available at t=0 and there is a scheduling sequence of five common jobs on six machines. The processing time and routing matrix of the jobs in each machine is given in the table below:

Table No 1.1 : The routing of jobs in each machine

Jobs	Machines				
Job1	1	2	3	5	6
Job2	1	2	3	4	null
Job3	1	2	4	3	null
Job4	1	2	4	5	6
Job5	1	2	5	6	null

Table No 1.2 : The processing time of jobs in each machine

Jobs	Machines				
Job1	2	5	1	3	3
Job2	3	3	4	3	0
Job3	1	4	1	4	0
Job4	1	2	3	5	3
Job5	1	4	1	2	0

Output from the Gantt chart of the FCFS rule based on the job sequence which means the job entered the first job is not based on the length or shorten of the job time rather if the job comes fires enter the operation. As we can see the make span found is ??? minutes for the jobs and the sequence of jobs are J4-J1-J5-J3-J2.

1.3.2 Shortest Job Next (SJN) :

- It is a non-preemptive algorithm.
- It selects the job having the lowest processing time to process after.
- It has minimum average waiting time among all the scheduling algorithms.
- By using this algorithm, starvation may be caused because of the jobs with shorter processing time to keep coming and it can be solved by ageing.
- SJF algorithm is a greedy algorithm that is used to calculate the spending time of each job where accurate running times are available.
- Completion time, waiting time, turn around time are being calculated in the SJF algorithm.

1.3.3 Priority Based Scheduling :

- It is one of the most common batch system non-preemptive scheduling algorithms.
- Every job has its own priority and the job with the most priority is executed first and so on.
- If jobs have equal priority, they will be executed on FCFS basis.
- Priority is decided based on resource, memory and time requirements.

1.3.4 Shortest Remaining Time Scheduling :

- It is the preemptive metaphrase of the Shortest Job Next scheduling.
- Each machine is allocated to the job that is closer to the completion time. But a newer ready job with shorter time to completion, preempt previous jobs.
- In case of short jobs, SRT is used in batch environments.

1.3.5 Round Robin Scheduling :

- It is a preemptive type algorithm.
- All the jobs are given a fixed time to get completed which is known as quantum.
- After completion of one job for a particular period, it is forestalled and the next job starts execution for a given time duration.
- In order to save the status of preempted jobs, context switching is used.

1.3.6 Multiple-Level Queues Scheduling :

- It is a dependent algorithm.
- Multiple queues use other existing algorithms for grouping and scheduling all the jobs with the same characteristics.
- Every sequence or queue can have its own scheduling algorithms.
- Priorities are assigned to each sequence or queue.

1.3.7 Chromosome representation job shop scheduling problem :

An ordered series of jobs or operations where every gene represents one operation, is called a chromosome. Sequence of the operations illustrated in the chromosome is actually the sequence of schedules. Schedules can be represented directly or indirectly.

In direct scheduling, the production schedule is directly represented by the chromosomes where there are - 1. Binary and job based representation
2. Dual point crossover
3. Mutation by bit-flipping
4. Phenotype of each individual indicating job sequence for each machine.

Chapter 2

2. Related Works :

K. Mesghouni et al introduced an evolutionary algorithm which employs a probabilistic search process to solve JSSP. By choosing a suitable representation of chromosomes through encoding methods, genetic operators adapted for each representation have been developed. [15]

Erik Pitzer discussed the purpose of fitness landscape analysis that can obtain insights into problem classes through search space,fitness function, landscape variants, spectral analysis, basin analysis, epistasis, neutrality etc. [16]

Bart Naudts et al have discussed the epistasis variance and fitness distance correlation and proposed a new non-linear fitness scaling considering both GA-easy and GA-hard fitness functions. They draw the samples from epistasic structures for a given fitness function that helps for successive generations of the run of the evolutionary algorithm. [17]

Jeffrey Horn constructed a unimodal function and multimodal function for examining the limit of modality which is related to the finding of the best point on the fitness landscape by genetic algorithm and hill climber algorithm. Mainly, he addressed the solution of Sewall Wright's fitness landscape to be more appropriate. [18]

Jérémie Humeau et al proposed the need of software frameworks that includes hill climbing, tabu search, iterated local search algorithms for solving an optimization problem with a large variety. For designing these algorithms, fitness landscape analysis of the optimization problem is essential as the properties of the landscape have the ability to describe, predict the behaviour of the local search metaheuristics. [19]

Jason Adair et al claimed a complicated network analysis for continuous landscapes where it works as a discrete sample of combinatorial fitness landscape with local optimas and search operators. [20]

Fuqing Zhao et al have presented the fitness landscape of the factorial representation for no-wait flow shop scheduling problem where the makespan criteria has been observed along with the observation of fitness distance plot and the analysis of factorial coding theory. Also, the various local and high ruggedness have been visualised for the suitable searchability of the landscape. [21]

S´ebastien Verel et al have proposed an extended version of neutral fitness landscape for combinative inquiry locations, that is based on two impartial duplicates (N, K) and enhances the heuristic search. [22]

Ivan G Szendro et al have identified the measurement of epistasis that is compared on fitness landscapes and applied to the empirical landscapes to visualise the factors that affect ruggedness. [23]

Gabriela Ochoa et al have introduced a network characterization of combinative fitness landscape from obtaining the concept of implicit network which works on NK landscapes. They also stated the characteristic of the basin of attraction along with the connections with local maximum networks. [24]

Bart Naudts et al have generalized a concept of epistasis, the interdependency of bits in the encipher of the fitness function with two distinct invariants and bit decidability invariant has been considered to be less difficult for the exact epistasis value for a particular fitness function. [25]

Kanate Ploydanai et al have proposed a modern set of rules that is exhibited on a rushing scheduling heuristic by putting together the availability obligations of machines to decrease the total completion time objective in JSSP. [26]

Table No 2.1 : Combinations of various topics

Ref No.	**Publishing Year**	**Author's Name**	**Techniques**	**Present work and Future Works**
[15]	2014	K.Mashghouni et al	Parallel machine encoding, parallel job encoding	By creating parallel representation of chromosome and genetic operators, performance of JSP can be improved. Further, the genetic operators can be controlled to improve the performance of the proposed method.
[16]	2013	Erik Pitzer	Basin analysis, Isotropy analysis, spectral analysis etc	All the methods actually elucidate the clear understanding of the problem in their shapes in the fitness landscape which indicates the statistical significance
[17]	2000	Bart Naudts et al.	Monotone fitness function	By using 1-D nearest-neighbour interaction functions, an efficient prediction measure can be generated. Further, for more accurate prediction, susceptibility to nonlinear fitness scaling, constantness, and averaging can be constructed.
[18]	1995	Jeffrey Horn	Local Optima, Basin of attraction, Hill climbing method	Sewall Wright's landscape has been considered more useful to GA theory as it can tackle multimodality dimensions along with discussing isolation and meaningfulness.

[19]	2013	Jérémie Humeau et al.	Fitness landscape analysis, Local search methods	Automatic parameter setting, applying state of the art algorithms in optimization can be helpful for a fitness landscape of the optimization problem with a larger flexibility and robustness.
[20]	2019	Jason Adair et al.	Basin-Hopping algorithm, Benchmark functions	Variation of perturbation step size with global structure, local optima network as a high dimensional tool, the landscape's global structure have been found. Further, the sampling of initial solutions, the performance of local optimisation should be solved for cheaper computational study.
[21]	2019	Fuqing Zhao et al.	Factorial representation, Big valley structure	Big valley structure using factorial representation, BNS, SNS, RNS, INS on NWFSP improves the searchability of evolutionary algorithms. Further work can be done on the dispersion metric and local optima networks.
[22]	2011	S´ebastien Verel et al.	NK Landscapes, Basin of attraction, Fitness landscapes	Dynamic aspects of local search heuristic can be further analyzed by the information on the local limits and basin networks from present work in order to illuminate the nature of neutralism and evolutional investigation.
[23]	2012	Ivan G Szendro1, et al.	Fitness landscape models, Epistasis and ruggedness	The amount of epistasis in a particular landscape has been captured by the correlated solutions from the ruggedness and RMF model based on empirical landscapes. Depending on the availability of empirical landscapes, fitness landscapes and their evolutionary implications can be progressed further.

[24]	2008	Gabriela Ochoa et al.	Local optima, NK landscape	Extended concept of inherent network has been proposed and a maxima graph with short average path length has been shown by clustering coefficient. Neutral versions of NK landscapes can be improved for the designing and estimation of proficient finding techniques and operators.
[25]	1997	Bart Naudts et al.	Bit decidability, Higher order epistasis	The structure of NP-hard problems may be applied in the analysis of fitness landscape.
[26]	2010	Kanate Ploydana i et al.	Nodelay scheduling	Considering machine availability constraints, nodelay scheduling gives good results. In future, this constraint can be added to flexible job shop scheduling algorithms where complexity is high.

2.1 Problem Statement :

Job shop scheduling problem is a combative problem in this modern era. All the researchers have been working on the minimization of the makespan and fitness values of the jobs or machines. But there may be some dependencies among the jobs or machines. This dependency is known as epistasis where each job is actually interacting with others. We have chosen a problem where we will analyze the performance of epistatic interaction of local search techniques on the permutations of job shop scheduling.

2.2 Motivation :

As Job Shop Scheduling, an optimization problem, is a np hard one, no such exact algorithm is there by which JSSP can be solved in polynomial time. Therefore approximation algorithms such as dynamic programming, heuristic based approach etc come to the picture. But all heuristics will not perform better due their different strategies. So, it is required to have a micro level analysis of the solution landscape rather than macro level evaluation of the algorithms in terms of foreseeing their probable performance and of selecting an eligible algorithm. Epistasis is a genetic measure of effects of mutation on genotypic aka problem solution spaces. It is very efficient in the study of ruggedness and neutrality of the fitness landscape. Our motive is to analyse job shop scheduling problem hardness through epistatic measures for algorithm selection of the same.

Chapter 3

3. Proposed Methods :

In the earlier sections, we have seen that in the job shop problem, there are N number of jobs with M number of machines where every job has processing time for each machine. Also each job has their different machine sequence which affect the preempted job shop scheduling.

Time	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
J1	86	67	89	92	88	99	94	99	68	77
J2	92	82	94	50	75	72	69	94	66	63
J3	83	65	55	77	85	85	64	78	61	83
J4	76	61	68	54	99	66	75	94	67	63
J5	99	86	67	69	88	68	95	67	95	82
J6	88	99	80	79	81	64	66	69	80	62
J7	95	86	66	96	97	99	52	50	97	71
J8	62	71	51	82	98	94	71	85	95	79
J9	94	66	90	81	76	58	71	85	93	97
J10	59	82	96	50	81	59	58	56	67	96

order	o1	o2	o3	o4	o5	o6	o7	o8	o9	o10
j1	5	9	7	6	2	3	10	8	1	4
j2	6	4	7	5	3	9	1	2	8	10
j3	10	9	1	2	7	6	8	5	3	4
j4	8	3	2	5	4	7	6	1	10	9
j5	4	5	10	9	1	3	7	6	8	2
j6	2	5	6	7	9	3	8	10	4	1
j7	8	2	5	4	1	9	3	6	7	10
j8	5	7	4	3	2	6	8	1	9	10
j9	1	7	4	8	2	3	5	6	9	10
j10	4	1	2	9	8	10	7	5	6	2

Figure No 3.1: A sample dataset

We have taken a sample data set with N jobs and M machines where N = M = 10. We will go through the following sections to analyze our fitness landscape where the epistatic measures of each solution will be plotted for the above sample set and also for other sample sets with different dimensions.

3.1 Factorial Number Representation:

In combination mathematics, factorial number representation can be used to generate a decimal number for a particular permutation where t will be the number of digits in a permutation and range will be 0 to (t!-1). It was first proposed by Laisant[6] and he stated that each permutation of n digits will have a unique decimal number which will represent that particular permutation.

Here, we can have n jobs and the factorial number will be in the range of 0 to t!-1. We will take a list,S (J_1,J_2,...J_t) of job permutation as an input. (Then we will initialize a list, J of all the probable counts of the factorial base in the rising series, an ascending job list, L and empty factorial sequence fact) To get a factorial number instead of a particular job sequence, following steps will be done :

1. Initialization of J_G = {0,1,2,...,t} , L = {1,2,3,...,t}, fact={}, Fac = 0, j = 1
2. do

        position = Binarysearch (L[j]==J)
        fact[j] = J_G [position]
        Removal of P[position] from L
        Fac = Fac + fact[j] * J_G [t-j+1]!
        j = j + 1

    while(j <= t)

3.2 Neighbourhood Structure :

A neighborhood structure is one of the mapping functions where the neighbourhood operator is actually moving an element of the existing solution by performing a small movement. Here, the main motive is to enhance the efficiency of the local finding in the neighbourhood. The neighbourhood structures, definitions had been proposed by Marmion and Regnier-Coudert [27]. To create more solutions from the selected neighbourhood, classical neighbourhood structures such as insertion neighbourhood structure (INS), swap neighbourhood structure (SNS), reversion neighbourhood structure (RNS), block-shift neighbourhood structure (BNS) can be considered.

From the INS neighbourhood structure, such jobs or machine permutations are created where the x^{th} operator is removed and is inserted after the y^{th} operator where $x \neq y$ and $x \neq y + 1$. The SNS neighbour is actually obtained by swapping the x^{th} and the y^{th} operator with $x \neq y$. In the RNS neighbourhood, operators are swapped and the operators between the swapped operators are reversed. In the BNS neighbourhood, k consecutive operators are removed from the position a i.e $a < n - k + 1$ and reinserted into position b where $b > 1 + a - k$ or $a > b$.

In the following fig no 3.2, we can see the perturbation process of INS, SNS, RNS, BNS neighbourhood structure.

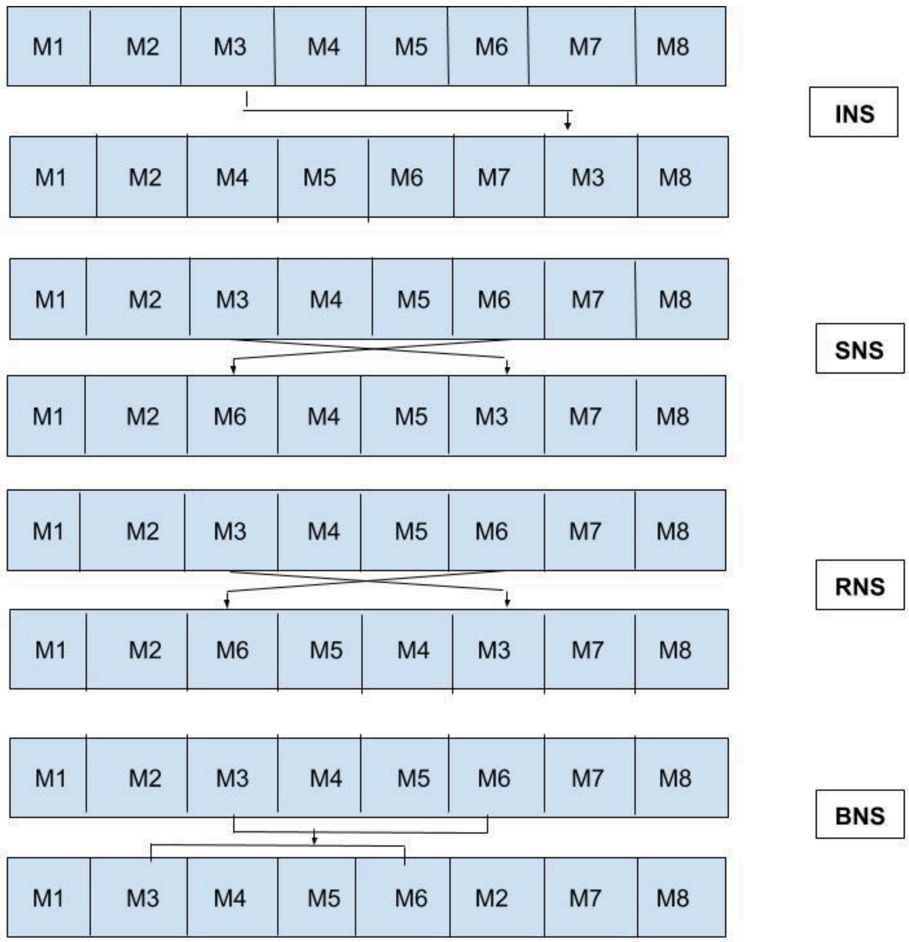

Fig No. 3.2 : Neighbourhood structure

3.3 Epistasis :

William Bateson, who discovered the word "genetics' ' as well as the word "epistasis" in the early 1900s to describe deviations from Mendelian inheritance. 'Epistasis' means "standing upon," and Bateson explained the characters which were layered on top of other characters by masking their expressions. So, epistasis mentions the behavioral outcome of interaction among genes at different locations. The dependency of the phenotypic variations among individuals with the same genotype at one location on their genotypes at another locus can be noticed. If the offshoots of each separated gene does not compute the offshoots of two genes after adding, epistasis is probably present. We can assume that two genes are leading to grown weight ('B' and 'C'). Each sets on a 1-pound gain in body weight. If one individual having both genes gained 2 pounds, that would mean a simple additive design of inheritance which indicates no epistasis is present. If one individual having both genes showed a 10-pound weight gain or weight loss that would mean epistasis is present. From this example, the effect of a gene is detectable only when the knowledge of epistatic interactions with other genes is incorporated.

Epistasis can be introduced as a generalization process of investigating between one or more locations on a chromosome. It is actually a concept of the fitness landscape, where it gives a more proper resolution. Epistasis is the reciprocal action among the outcomes of the mutations at the various locations and is identified as a nonadditive portion and the non-multiplicative part of the fitness effect of the mutations in log-scale. Epistasis expresses correlation between the bits and optimal bit decidability along with the large classes of the function. In (Rawlins 1991), Rawlins characterized maximal epistasis, where any exact part of a set of genes is not dependent on any other genes and minimum epistasis, where each bit is not dependent on any other gene. Epistasis can also be 1-epistasis and n-epistasis. 1-epistasis is measured by how much the function differs from those which depend on each bit and in case of n-epistasis, function will depend on the group of n-bits.

To evaluate epistasis, a fitness function must be defined in terms of its components.Without any epistatic interactions, linear fitness function could be reformulated as $f(x) := \sum^{n}_{i=0} f_i(x_i)$,
where a solution candidate $x \in S$ is decomposed into its "components" $x = \{x_i\}^{n}_{i=1}$
The fitness function will contain linear function components ,such as $f_1(x_1)$ based on a single component as well as more and more epistatic components, such as $f_{1;2;3}(x_1;\ x_2;\ x_3)$; as epistasis increases.

3.3.1 Epistasis Variance :

The measurement of epistasis was first proposed by Davidor in 1990 where it was said that a linear function can be considered as a fitness function if it doesn't have any epistasis and the amount of epistasis is called epistasis variance. It is the comparison between linear approximation and actual fitness function. By assuming a linear model as $f'(x) = \sum f_i\,(x_i)$ where we have to measure the difference between the variance of the original fitness function and the linear approximation.

$$EV(f) \;=\; (VAR[f(x)] \;-\; VAR[f'(x)]\,)\,/\,VAR[f(x)]$$

In (Rochet et al., 1998[7]), to deceive the epistasis measurements a different encoding is used which minimizes the measurable epistasis. Let S is a chromosome of Ω where Ω is the population and C(S) will be the fitness of $S \in \Omega$.
$b = \{0,\ 1,\ \#\}$ where # represents either 0 or 1. A schema, s is an element of b^n where n is the number of genes in the chromosome, S.

Notations.

- $[s] = \{ S \in \Omega ; (S_i = s_i) \Leftrightarrow (s_i \neq \#) \}$.
- $o(s) =_{def} cardi\{ j ; s_j \neq \#\}$ is known as the sequence of s .
- $Hyp(S) =_{def} \{ s \in b^n ; S \in [s]\}$
- $Hyp_j(S) =_{def} \{ s \in b^n ; S \in [s] \text{ and } o(s) = j \}$ for j= 0 to n-1
- $\mu_s = (1 / |[s]|) \sum_{S \in [s]} C(S)$

Let's assume, n = 6 and s = 1#00#1, then [s] = {100001, 110011, 100011, 110001}, |[s]| = 4, o(s) = 4. Now, n = 4 and S = 1001.
So, Hypo(S) = { 1001, #001, 1#01, 10#1, 100#, ##01, 1##1, 10##, 1#0#, #0#1, #00#, ###1, 1###, #0##, ##0#, #### }, H_0(S) = { #### }, $Hypo_1$(S) = { #001, 1#01, 10#1, 100# },
$Hypo_2$(S) = { ##01, 1##1, 10##, 1#0#, #0#1, #00# }, $Hypo_3$(S) = { ###1, 1###, #0##, ##0# },
And $Hypo_4$(S) = { #### }

3.3.2 Davidor's Epistasis :

To calculate the amount of non-linearity in a representation, Davidor has introduced a variance where the probability of the fitness of a chromosome decreases with the increment of the dimension of reciprocal action among various locations.
Davidor introduced a function, W defined on Ω mean fitness and fitness of schema by

$$W(S) = \mu + \sum_{s \in Hypo(S)} (\mu_s - \mu) = \sum_{s \in Hypo(S)} \mu_s - (n -1)\mu$$

We can calculate the fitness of the chromosome by the fitness of the given schema.
EXAMPLE 1 : n = 4 and C(0000) = 2,

C(0001) = C(0010) = C(0100) = C(1000) = 3
C(1100) = C(1010) = C(0110) = C(0011) = C(0101)= C(1001) = 4
C(1110) = C(1011) = C(0111) = C(1101) = 5
C(1111) = 6

If S=1001, then $Hypo_1$(S)={1###,#0##,##0#,###1} and
$\mu_{1\#\#\#}$ = (C(1000)+ C(1100)+ C(1010)+ C(1110)+C(1011)+C(1101)+C(1111)+C(1001)) / 8 = 5
$\mu_{\#0\#\#}$ = (C(0001)+ C(0010)+ C(1000)+C(1010)+ C(0011)+C(1011)+ C(0000)+C(1001)) / 8 = 4
$\mu_{\#\#0\#}$ = (C(0001)+ C(0100)+ C(0000)+C(1000)+C(0101)+C(1100)+C(1101)+C(1001))/ 8 = 4
$\mu_{\#\#\#1}$ = (C(0001)+ C(1111)+ C(0011)+C(0101)+C(1011)+C(0111)+C(1101)+C(1001)) / 8 = 5
μ= (C(0000) + C(0001) + C(0010) + C(0100) + C(1000) + C(1100) + C(1001) + C(1010) + C(0110) + C(0011) + C(0101) + C(1110) + C(1011) + C(0111) + C(1101) + C(1111)) / 16 = 4
Then , W(S) = (5+4+4+5) - (3*4) = 6.
We have got the value of Davidor's function, W(S) . Now, we can calculate the epistasis of chromosome S, D(S) by the difference of X(S) and W(S).

$D(S) = X(S) - W(S) = 4 - 6 = -2$, which indicates reciprocal epistasis.

3.4 Graded Epistasis :

More orders of epistatic interaction are proposed in (Rochet,1997). To calculate the non-linear order of epistatic interaction , graded epistasis variance can be considered. There will be a direct correlation between the simplified and original fitness function where comparison of variances is measured in normal epistasis and the different orders of nonlinearity cannot be measured by Davidor's epistasis.

Now, graded epistasis of chromosome S, $G_j(S) = C(S) - W_j(S)$, where W_i is defined on the population, Ω for $j = 1$ to n as

$$W_j(S) = \mu + \sum_{s \in Hypo(S)} (\mu_s - \mu)$$

From example 1, by taking same S = 1001 and j = 2,
we get $Hypo_2(S)$ = {10##,#00#,##01,1##1,1#0#, #0#1}.
$\mu_{10\#\#}$= (C(1000)+C(1010)+C(1001)+C(1011)) / 4 = 4
$\mu_{1\#\#1}$= (C(1001)+C(1011)+C(1101)+C(1111)) / 4 = 5
$\mu_{\#00\#}$= (C(0000)+C(0001)+C(1000)+C(1001)) / 4 = 3
$\mu_{1\#0\#}$= (C(1000)+C(1001)+C(1101)+C(1100)) / 4 = 4
$\mu_{\#0\#1}$= (C(0001)+C(0011)+C(1001)+C(1011)) / 4 = 4
$\mu_{\#\#01}$= (C(0001)+C(1101)+C(1001)+C(0101)) / 4 = 4

μ=(C(0000)+C(0001)+C(0010)+C(0100)+C(1000)+C(1100)+C(1001)+C(1010)+C(0110)+C(0011) +C(0101)+C(1110)+C(1011)+C(0111)+C(1101)+C(1111)) / 16 = 4
So, $W_2(S)$ = (4+5+3+4+4+4) - (3*4) = 12 and $G_2(S)$ = 4 - 4 = 0 , which indicates no epistasis.
From the amount of graded epistasis, we can get the graded epistasis variance.

$$\varepsilon^2_i = \sum_{S \in \Omega} (G_i(S))^2 / |\Omega|$$

3.4.1 Epistasis as Correlation:

The measurement of epistasis in the fitness landscape can be expressed as correlation, γ. It calculates how the alteration of central mutation by another mutation of another location affects the atmosphere across the total possible features. It is a physical amount of local dependency.The range of γ lies between 1 and -1. From correlation, the effects of epistasis in the fitness landscape can be seen in three types of epistatic interactions which are magnitude epistasis, sign epistasis, reciprocal epistasis.

1. Magnitude epistasis indicates the interactions between a pair that keep the sign of fitness effects constant. The value of γ lies between 1 and 0 ($1 > \gamma \geq 0$)
2. Sign epistasis also refers the pairwise interaction but the fitness effect of one mutation will change after another mutation and the value of γ lies in $1 > \gamma \geq -\frac{1}{3}$
3. Reciprocal epistasis changes the signs of both fitness effects by pairwise interactions and it may have a negative value of $0 > \gamma \geq -1$.

Chapter 4

4 Results and discussions :

Experimental setup and random walk

All experiments have been carried out on Intel i5 dual core CPU and Windows 10 with Python 3.7 with the aid of Jupyter notebook. We have considered the three cases of job shop scheduling as for abz5[28], la37[29] and yn1[30] as the benchmark. The benchmarks, abz5 (10 x 10), la37 (15 x 15) and yn1(20 x 20) have the order of machines of every job along with the processing time of every machine of every job. We studied 10000 random solutions through a random walk described below:

Procedure : Random_Walk

1. Generate a random solution from the problem space.
2. Calculate the fitness of the answer and store it with the present answer's factorial representation.
3. While ending criteria not met, do:
    a. Generate a sample feasible answer set of the current answer based on different neighborhood definitions.
    b. Elect an individual answer randomly from the sample answer set.
    c. Calculate the fitness of the answer and store it with the present answer's factorial representation.

The above procedure utilizes RNS, INS, BNS, and SNS neighborhood structure as described in the previous chapter. To generate the neighborhood sample from a current solution we have considered the sample size would be 50. From there, a random solution is being selected as the current one for the next iteration. The random solutions' makespan are graphically displayed in figure 4.1 by selecting 10% solutions randomly of the sample to observe the fitness distribution of the specified problem. It has been observed that the sample space is flat in nature with many local minima existing within it. It almost resembles a Rastrigin function landscape with the flat valley like terrain geometry containing multiple numbers of local optima.

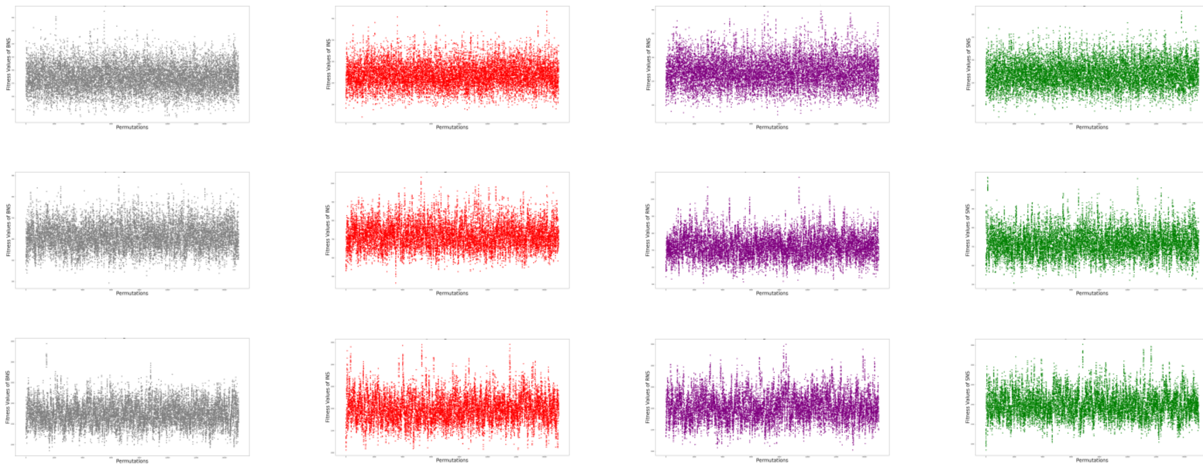

Fig 4.1 Makespan distribution for abz5, la37 and yn1 dataset for BNS, INS, RNS and SNS neighborhood structure respectively

From the previous study on literature **[][][][],** it has been observed that the landscape becomes more rugged with the increase of the epistatic interaction in the landscape path discovered through some local search because the local investigations discover the neighborhood solution of the present one with the course of the every iteration. To observe this phenomenon, we have considered 1% of the random solutions generated by the random walk defined here and calculated their epistatic variance and Devidor's epistasis measure and plotted in the figure 4.2 and 4.3. Figure 4.2 depicts the Devidor's epistasis of the BNS, INS, RNS and SNS neighborhood with all three benchmark problems. It is observed that INS representation shows most representation whereas RNS shows most vitiation for all of the benchmark problems. Also it has been observed that all of the measures have a general trend parallel to the x axis. This phenomenon concretes the assumption that the general landscape is like a flat valley with existence of many local optima. Figure 4.3 exhibits the epistatic variance of the three datasets for all four neighborhood structures. It also justifies the above assumption. Moreover the changes of only magnitude epistasis measures have been observed as all the epistasis values are greater than 0. It also implies the truth of the assumption of existence of flat valleys like landscapes with many local optima.

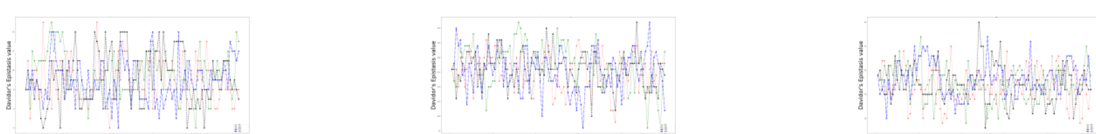

Fig 4.2 Devidor's epistasis for BNS, INS, RNS and SNS neighborhood structure of each abz5, la37 and yn1 dataset respectively.

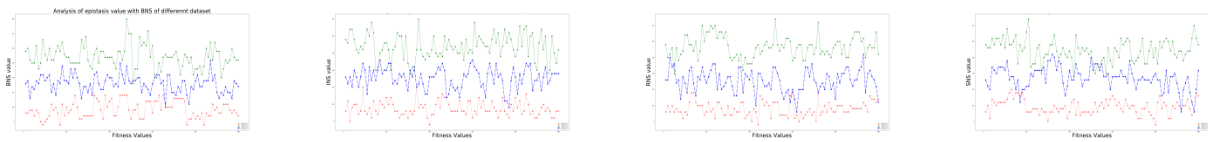

Fig 4.3 Comparison of epistatic variance of abz5, la37 and yn1 dataset of each BNS, INS, RNS and SNS neighborhood structures

Chapter 5

Future work :

In this paper, we have worked on the fitness landscape analysis of the epistatic measurements with the different neighbourhood solutions which actually shows the dependency measurement of the solutions on each other. In the future, detailed fitness landscape analysis can be done on different representations of the job shop scheduling problem and its variants like neutral network, ruggedness. Also epistasis based performance study on any evolutionary algorithm on the JSSP. Based on the state-of-the-art autoML or deep learning techniques, a selection strategy algorithm can be designed for the JSSP. The trained solutions from this paper work can be further provided to receive an optimized model as output in the autoML system. This autoML ensemble strategy formulation will help to select an appropriate evolutionary operator in the memetic algorithm framework.

Conclusion :

The macro-level analysis of the evolutionary algorithm to a job shop scheduling problem with sample data sets has been defined. As fitness landscape analysis has been devised as a general method for problem understanding, here we have demonstrated the landscape analysis by choosing random solutions from neighbourhood structures where the results are being analyzed by plotting graphs between permutations and their epistasis values as well as between neighbourhood structures. An ample survey of this random walk method showed the disparity and diversity of the results from different neighbourhood structures elucidating different aspects of optimizations.

We have generated different possible permutations of machines for a particular job from INS, SNS, RNS, BNS neighbourhood structure by evaluating algorithms. A proper analysis of these epistatic measurements on the fitness landscape is still an open issue. As there are no guidelines for analyzing the job shop scheduling problem considering different variants like natural network, ruggedness etc., investigations are still necessary to determine the performance of different variances of job shop scheduling in the fitness landscape. In the work, we have actually stepped on a new analysis ground, combining the neighbourhood structures with epistasis measurements in a large scale study for the practical and suitable applicability of fitness landscape analysis.

Chapter 6

Reference :

1. Ziaee, M. (2014). Job shop scheduling with makespan objective: A heuristic approach. International Journal of Industrial Engineering Computations , 273–280.
2. Weinberger, E.: Correlated and uncorrelated fitness landscapes and how to tell the difference. Biol. Cybernetics 63, 325–336 (1990)
3. Stadler, P.F., Schnabl, W.: The landscape of the traveling salesman problem. Physics Letters A 161, 337–344 (1992)
4. Kirkpatrick, S., Toulouse, G.: Configuration space analysis for traveling salesman problems. Journal de Physique 46, 1277–1292 (1985)
5. Reeves, C.R.: Landscapes, operators and heuristic search. Annals of Operational Research 86, 473–490 (1999)
6. C. A. Laisant, ``Sur la numération factorielle, application aux permutations," Bull. Soc. Math. France, vol. 16, pp. 176183, Nov. 1888.
7. Rochet, S., Venturini, G., Slimane, M., and Kharoubi, E. M. E. (1998). A critical and empirical study of epistasis measures for predicting GA performances: A summary. In Artificial Evolution: Third European Conference, AE ’97, pages 275–286. Springer.
8. Carlo Meloni, D., Pacciarelli, M., Pranzo (2004). A Rollout Metaheuristic for Job Shop Scheduling Problems. Annals of Operations Research volume 131, pages 215–235(2004)
9. Kauffman, S.A.: Adaptation on rugged fitness landscapes. In: Stein, D. (ed.) Lectures in the sciences of complexity, pp. 527–618. Addison-Wesley, Reading (1989)
10. I. Choi and D. Choi, “A local search algorithm for job shop scheduling problems with alternative operations and sequence dependent setups”, Computer and industrial engineering, vol. 42, pp. 43-58, 2002.
11. D. A. Koonce and S. C. Tsai, “Using data mining to find patterns in genetic algorithms to a job shop schedule”, Computer and industrial engineering vol. 38, pp. 361-374, 2000.
12. H. Ohta and T. Nakatani, “A heuristic job shop scheduling algorithm to minimize total holding cost of completed and in progress products subject to no tardy job”, International journal production economics, vol. 11, pp. 19-29, 2006.
13. A. Caumond, P. Lacomme, and N. Tchernev, “A memetic algorithm for a job shop with time lag”, Computer and operational research.
14. K. Jansen, M. Mastrolilli, and R. S. Oba, “Approximation scheme for job shop scheduling problem with controllable processing time”, European journal of operational research, vol. 1672, pp. 97-319, 2005.
15. K. Mesghouni, S. Hammadi, P. Borne, “EVOLUTIONARY ALGORITHMS FOR JOB-SHOP SCHEDULING”, Int. J. Appl. Math. Comput. Sci., 2004, Vol. 14, No. 1, 91–103
16. Erik Pitzer, “Applied Fitness Landscape Analysis” , Linz, Jänner, 2013

17. Bart Naudts, Leila Kallel , “A Comparison of Predictive Measures of Problem Difficulty in Evolutionary Algorithms” ,IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 4, NO. 1, APRIL 2000.
18. Jeffrey Horn, “Genetic Algorithms, Problem Difficulty, and the Modality of Fitness Landscapes” , IIiGAL Report No. 95004, 1995.
19. Jérémie Humeau, Arnaud Liefooghe, El-Ghazali Talbi, Sébastien Verel. ParadisEO-MO: From Fitness Landscape Analysis to Efficient Local Search Algorithms. [Research Report] RR-7871, INRIA. 2013. hal-00665421v2
20. Jason Adair, Gabriela Ochoa, Katherine M. Malan, “Local optima networks for continuous fitness landscapes”, Conference Paper, July 2019.
21. F. Zhao, F. Xue, G. Yang, W. Ma, C. Zhang, H. Song, “A Fitness Landscape Analysis for the No-Wait Flow Shop Scheduling Problem With Factorial Representation”, Volume 7, 2019.
22. S´ebastien Verel, Gabriela Ochoa, Marco Tomassini, “Local Optima Networks of NK Landscapes
With Neutrality”, IEEE Transactions on evolutionary computation, Volume 15, No. 6, December, 2011.
23. I.G Sexendro, M.F Schenk, J. Franke, J.Krug, J.A. GM de Visser, “Quantitative analyses of empirical fitness landscapes”, arXiv:1202.4378v2 [q-bio.PE] 17 Oct 2012.
24. G. Ochoa, M. Tomassini,S. Verel, C. Darabos, “A Study of NK Landscapes’ Basins and Local Optima Networks”, arXiv:0810.3484v1 [cs.NE] 20 Oct 2008.
25. Bart Naudts, Dominique Suys, Alain Verschoren, “Epistasis as a Basic Concept in Formal Landscape Analysis”,Published in ICGA 1997
26. Kanate Ploydanai, Anan Mungwattana, “Algorithm for Solving Job Shop Scheduling Problem Based on machine availability constraint”,Kanate Ploydanai et. al. / (IJCSE) International Journal on Computer Science and Engineering, Vol. 02, No. 05, 2010, 1919-1925.
27. M.-E. Marmion and O. Regnier-Coudert, "Fitness Landscape of the Factoradic Representation on the Permutation Flowshop Scheduling Problem," vol. 8994, pp. 151-164, 2015.
28. https://github.com/tamy0612/JSPLIB/blob/master/instances/abz5
29. https://github.com/tamy0612/JSPLIB/blob/master/instances/la37
30. https://github.com/tamy0612/JSPLIB/blob/master/instances/yn1