# Two Improved Cuckoo Search Algorithms for Solving The Flexible Job-Shop Scheduling Problem

Ahmed T. Saadeq Al-Obaidi, Samer Alaa Hussein

*Computer Science Department, University of Technology, Baghdad, Iraq*

***Abstract***— The Cuckoo Search (CS) is heuristic search algorithm which is inspired from cuckoo bird behavior. In this paper, we propose two improvements for the cuckoo search algorithm for solving Flexible Job-Shop Scheduling problem (FJSP); the first one depends on Best Neighbors Generation (CS-BNG) and the second one is based on Iterative Levy Flight (CS-ILF). Some adaptation for the key points of CS algorithm has been done to enhance searching in the discrete state space. The proposed algorithms have increased the solutions' quality and convergence rate. The improved algorithms have been tested on some FJSP benchmark instances for performance examination. The experimental results demonstrate the effectiveness of the improved algorithms in comparison to the basic cuckoo search algorithm.

***Keywords***— Metaheuristic algorithm, Cuckoo Search, Combinatorial Problems, Flexible Job-shop Scheduling.

## I. Introduction

Scheduling plays an important role in most manufacturing and production systems as well as most of the information processing environment, such as sequencing the jobs in a manufacturing system to process on a sequence of machines.

Job-shop Scheduling Problem (JSP) is one of the main scheduling problems, which was classified as NP-hard problem [1] .In the JSP, there are a number of jobs that must be processed on a given collection of machines .Each job is composed of a fixed sequence of operation. Each operation has a specified machine to process on. A machine can only process one operation at a time [2].

A flexible Job-Shop Scheduling Problem (FJSP) is a generalization of the job shop [1]. In FJSP, each job operation can process on a set of machines with a processing time which made the problem more difficult to specify a proper machine from a given set that should process each operation [2].

The FJSP consists of two sub-problems: the routing and scheduling [3, 4], where the routing sub-problem means the determination of the assignment for each operation to a machine from a given set of the machines and the scheduling sub-problem is sequencing the assigned operations on all machines to get a feasible scheduling with a minimum predefined objective function. Several objectives have been achieved with JSP, but makespan minimization is the most widely studied [5], which is the maximum completion time for the jobs in the scheduling.

From the NP-harness of the FJSP, there is no efficient algorithm to find the optimal solution; therefore, the metaheuristic algorithms are the important alternative for solving this class of problems, which give a feasible solution in an acceptable amount of time [6].

The rest of the paper is organized as follows. Some related works are reviewed in Section 2. Section 3 presents a brief description for CS Algorithm. Discrete Levy Flight and parameters of Modified CS algorithm are explained in Section 4. The first improved CS is presented in Section 5 and the second improvement proposed in Section 6. The experimental results are introduced in Section 7.

Lastly, this paper ends with some conclusion in Section 8.

## II. RELATED WORKS

The robustness and the effectiveness of the Cuckoo Search algorithm encourage the researchers to improve its performance for many optimization problems.

H. Zheng and Y. Zhou [7] implemented a novel cuckoo search optimization algorithm based on Gauss Distribution (GCS). The GCS algorithm has been used to solve standard test function and engineering design optimization. The algorithm found a more quality solution than the best solution obtained by CS, and the GCS had higher convergence rate.

Y. Zhou et al. [8] proposed modified CS algorithm for solving the graph coloring problem.

The improvement consists of the greedy transform algorithm which has been added to enhance the traditional CS performance. The map of China was taken as an example to demonstrate ICS performance for solving the problem. The comparison result shows the efficiency and accuracy of the ICS and high effectiveness of this algorithm appeared in the moderate size of the graph coloring problem.

P. Ong [9] proposed Adaptive Cuckoo Search Algorithm (ACSA) for Unconstrained Optimization. The step size α is adjusted adaptively in this method instead of using constant value. From all considered benchmark optimization functions, the ability of the ACSA to converge was faster than CS in less iteration. This modification gave the preference of the ACSA over the standard CS in the convergence rate.

T. Nguyen and D. Vu [10] present an efficient CS modification to solve short-term hydrothermal scheduling (HTS) problem. In this modification, after sorting the eggs in the descending order according to their fitness function value, they will be classified in two groups. The top group is contains the egg with low fitness value while the abandoned group is contents the other eggs. This method proposed changing the step size of the Levy flight in CS with the number of iteration that will encourage more localized search in getting closer to the optimal solution. Furthermore, two eggs will exchange information in top eggs group to accelerate the search process of the eggs. The robustness and the effectiveness of the proposed method have been validated by comparing with different test systems.

S. Singh and K. P. Singh [5] proposed a new hybrid approach for solving makespan objective in job shop scheduling problem by mixing a set of assorted individual enhancement with CSO technique to increase the cuckoo search ability. Random key encoding was represented to transform a real-valued vector to discreet one. The analyzing results show that the hybrid approach is better than simple CSO.

FJSP attracts many researchers for solving this problem due to its importance in the many practical fields.

A. Thammano and A. Phu-ang [11] solved FJSP problem using hybrid Artificial Bee Colony algorithm with local search for makespan. The initial solution was created using several dispatching rule and harmony search algorithm. For escaping from the local optimum, the simulated annealing algorithm will be applied. Finally, crossover operation was proposed for the exploration enhancement. The experimental results show the efficiency of the proposed algorithm.

M. Nouiri et al. [12] applied particle swarm optimization algorithm (PSO) to solve this FJSP problem. They built an effective algorithm characterized by reconfigured easily for embedded system appropriate for an unpredictable environment. The developed PSO algorithm tested on several benchmark data and the effectiveness of the algorithm had been proven for solving FJSP problem.

W. Zhou et al. [13] combined cultural algorithm with particle swarm optimization (CPSO) for minimizing maximum completion time of the FJSP problem. The belief space and the population space were modified. Low variability and stability of the hybrid algorithm CPSO were shown in the simulation results. From comparison with PSO, CPSO is more effective to deal with FJSP.

Y. Wang et al. [2] improved Social Spider Algorithm (SSA) for FJSP. They combined selection, crossover and mutation operations with the original SSA for the performance enhancement. The initial individual was generated using a combination of rules to have a quality and diverse solutions. The proposed algorithm produces better solutions quality in compression with other algorithms.

## III. Cuckoo Search Algorithm

CS is a metaheuristic algorithm proposed by Yang and Deb [14]. The theoretical inspiration of the algorithm came from the behavior of the cuckoo bird aggressive reproduction strategy. Laying cuckoos eggs would be in a different nest of the host bird even it was for different species. The cuckoo eggs may be discovered by the host bird and either it would be destroyed or the host bird nest would be abandoned and completely build a new nest. This has resulted in the cuckoo eggs evolution which mimics the local host bird’s eggs. Yang and Deb implement this strategy as an optimization tool using three ideal rules [14, 7]:

*1) Each cuckoo lays one egg, which represents set of solution coordinates, at a time and dumps it in a randomly chosen nest.*

*2) A fraction of the nests with high-quality eggs (best solutions), will carry over to the next generation.*

*3) The number of nests is fixed and the host can discover an outsider eggs with a probability $P_a \in (0,1)$, which is the elimination of the egg or the nest by the host and building a completely new nest in another area.*

In view of these three rules, the main steps of CS algorithm can be outlined as the pseudocode appeared in Fig. 1. A new solution $x^{(t+1)}$ for the ith cuckoo is produced using levy flight as follows

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus \text{Levy}(\beta) \qquad (1)$$

where $\alpha > 0$ is the step size that ought to be identified with the problem scales. In most of the time, we can utilize $\alpha$=1. The product $\oplus$ means entry-wise multiplication. Levy flights basically produce a random walk and their random step-lengths are pulled from the Levy distribution

$$\text{Levy} \sim u = t^{-\beta} \qquad (1 < \beta \leq 3) \tag{2}$$

which has an infinite variance. The successive steps/jumps of a cuckoo basically compose the process of a random walk which comply a power-law step-length distribution with a heavy tail. A fraction $P_a$ of the worst nests can be discarded and construct new nests at new places using random walk.

```
Cuckoo Search based on BNG Algorithm
Input: Population of the problem, $p_a$
Output: Best Solution
Begin
  Objective function $F(x)$, $x=(x_1, \dots, x_d)^T$
  Generate initial population of n host nests $x_i$ ($i = 1, 2, \dots, n$)
  While (t <Max Generation) or (not stop criterion)
    Generate new solution by levy flight say i
    Evaluate its quality/fitness $F_i$
    Choose a nest among n (say, j) randomly
    if ($F_i > F_j$)
      replace j by the new solution;
    A fraction ($p_a$) of worse nests are abandoned and new
      ones are built randomly;
    Keep the best solutions (or nests with quality solutions);
    Rank the solutions and find the current best
  end while
  Post process results and visualization
End
```

Fig. 1 Basic Cuckoo Search Algorithm

## IV. Discrete Levy Flight and Parameters of Modified CS

In the proposed algorithms, Levy flight was developed to suit the flexible job-shop scheduling problem and the population size and maximum generation have been proposed as follows:

### A. Modified Levy Flight

Levy Flight applied for continues space. Therefore, we proposed some modification for the Levy flight equation to appropriate searching in discrete space.

Let $X_i^{(t)}$ and $X_j^{(t)}$ are two random solutions from the population of the problem in the t-th iteration. The new solution generated by the equation:

$$X_i^{(t+1)} = X_i^{(t)} + \alpha\,(X_j^{(t)} - X_i^{(t)}) \times t^{-\beta} \tag{3}$$

where the term $\alpha\,(X_j^{(t)} - X_i^{(t)})$ represented as the difference from the current solution $X_i^{(t)}$ , $\alpha$ =1, and the ($t^{-\beta}$) is the random steps which drown from Levy Distribution, as in (2). The parameter ($\beta$) is chosen from the formula:

$$\beta = \beta_{min} + (\beta_{max} - \beta_{min} / t_{max}) \times t \tag{4}$$

where $\beta_{min}$ and $\beta_{max}$ are the minimum and maximum allowable value from Levy Distribution. In our proposed algorithms, $\beta_{min}$=1.1 and $\beta_{max}$= 3.

The subtraction, multiplication, and addition operations in the Levy Light, as in (3), are modified by some neighborhood structure for the discrete scheduling problem as the following:

#### 1) Subtraction:

While the solution is a jobs permutation vector and machine assignable vector, the subtraction operation presented as a comparison between two solutions (jobs permutation vector) to produce a new solution chain $X_{sub}$.

Let $X_1$ and $X_2$ are two solutions, for all the elements if given elements of two solutions are equal then the element in the new chain is set as zero. Otherwise, the element in the new solution chine takes the value of the corresponding element in $X_1$. See Fig. 2.

| | | | | | | |
|---|---|---|---|---|---|---|
| $X_1$ | 1 | 3 | 1 | 2 | 3 | 2 |
| $X_2$ | 2 | 1 | 3 | 2 | 3 | 1 |
| $X_{sub}$ | 1 | 3 | 1 | 0 | 0 | 2 |

Fig. 2 The subtraction operation $X_1$-$X_2$

#### 2) Multiplication

The multiplication operation is a product of ($\sigma$: = $t^{-\beta}$) and the solution chain ($X_{sub}$) which taken from subtraction operation. A random number from the uniform interval (0,1) is generated for each element in the chain ($X_{sub}$) then the random number is compared with $\sigma$, if the random number is equal or more than $\sigma$, the element in the new chain ($X_{mult}$) set equal to the ($X_{sub}$) element at the same position. Otherwise, it takes zero number. Fig. 3 represents the multiplication process.

$\sigma$ = 0.25

| | | | | | | |
|---|---|---|---|---|---|---|
| $X_{sub}$ | 1 | 3 | 1 | 0 | 0 | 2 |
| rand(0, 1) | 0.81 | 0.15 | 0.34 | 0.95 | 0.08 | 0.76 |
| $X_{mult}$ | 1 | 0 | 1 | 0 | 0 | 2 |

Fig. 3 The multiplication operation.

#### 3) Addition

The addition operation is represented as a number of swap moves. It is the final step to generate a new solution $X_{new}$ from the Current Solution X.

For each element $X_{mult}(i)$ of $X_{mult}$, if $X_{mult}(i)$ is equal to zero, the $X_{new}(i)$ of $X_{new}$ takes the same job of X(i) for the same position. Otherwise, $X_{new}$ takes the job of X(i) in a random position. Means, The jobs which have positions corresponding to non-zero elements in $X_{mult}$ are swapped randomly. Fig. 4 illustrates the addition operation.

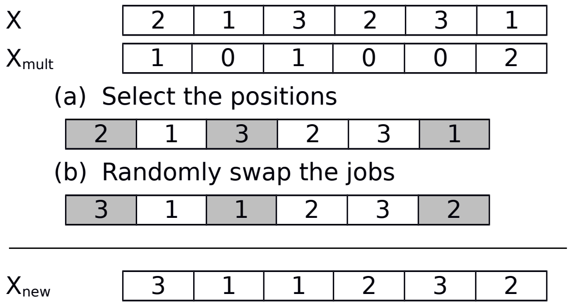

Fig. 4 The addition operation.

*B. Population Size related with Problem size*

The population size of the algorithm is proposed to be related to the size of the problem (number of jobs and machines) by the formula:

$$\text{Population size} = 0.5 \times (n \times m) \qquad (5)$$

where n is the number of jobs and m is the number of machines. The number of the population has an impact on the time and diversity in the algorithm, so this equation gives a moderate size for the population to find feasible solutions.

*C. Maximum Generation*

The iteration number has a role in the levy flight to find a new solution. Thus, the maximum generation number and the problem size suggested to be related as follows:

Max generation = 800, if (n×m) < 50.
Max generation = 900, if (n×m) = 50.
Max generation = 1000, if (n×m) > 50.

## V. CS BASED ON BEST NEIGHBORS GENERATION

The first proposed method, CS-BNG improved the CS algorithm in the step of generating new solutions to replace the abandoned ones. In basic CS generating solution, the replacement of the worst solution was made totally by generating solutions randomly in state space. That may be lead to decrease convergence rate to the optimal solution. Therefore, we proposed to divide a fraction of abandoned nests into two halves, the first half will replace by random generating of the solution and the second will be generated randomly from the neighborhood of the current best solution.

We kept random generation for the solutions which replace the abandon ones to maintain the power of the randomization in an exploration of the state space and using the BNG to increase the exploitation of the best location which has been reached in the generation at a time. A balance between diversification and convergence has been made to enhance the CS-BNG algorithm ability for finding the solutions.

Fig. 5, which shows the first improved CS algorithm based on BNG. The improvement step proposed to increase the convergence rate to the optimal solution and decreased time to get optimal or near to the optimal solution.

```
Cuckoo Search based on BNG Algorithm
Input: Population of the problem, $p_a$
Output: Best Solution
Begin
  Objective function F(x), $x=(x_1, \dots, x_d)^T$
  Generate initial population of n host nests $x_i$
(i = 1, 2, …, n)
  While (t <Max Generation) or (not stop
criterion)
    Generate new solution by levy flight say i
    Evaluate its quality/fitness $F_i$
    Choose a nest among n (say, j) randomly
    if ($F_i > F_j$)
      replace j by the new solution;
    A fraction $(p_a)/2$ of worse nests are
abandoned and new
      ones are built randomly;
    A fraction $(p_a)/2$ of worse nests are
abandoned and new
      ones are built by call BNG ($(p_a)/2$, current
best);
    Keep the best solutions (or nests with
quality solutions);
    Rank the solutions and find the current best
  end while
  Post process results and visualization
End
```

Fig. 5 Improved Cuckoo Search Algorithm based on CBN generation.

```
Best Neighbor Generation Algorithm
Input: A : (pa)/2, CB : current best;
Output: New Nests.
Begin
  for i=1 to A
    Generate a random neighbor $N_i$ from CB by 3-
opt
  End for
End
```

Fig. 6 Current Best Neighbor Generation Algorithm.

## VI. CS BASED ON ITERATIVE LEVY FLIGHT

Levy Flight gives robustness for CS algorithm for finding a new position in the state space which means generating a new solution from the current solution. Thus, we proposed to repeat the levy flight step which includes (individual selection, new solution generation, and survival of the better solution) multiple times in the iteration, instead of once in basic CS, for more exploitation. This improvement combined with the first improvement to present the second improvement CS-ILF.

A number of repetitions affect the time, thus, a small number of repeated Levy Flight step must be chosen for a reasonable time, and we suggest this number related to the population size of the problem. Iterative Ratio (IR) presented as the ratio of the population size for the repetition times of the Levy flight. This improvement enhances exploration of the CS-ILF algorithm in an acceptable time. Fig. 7 shows the second improved CS algorithm based on iterative Levy flight.

This method increased the solution quality to reach the optimal or near to the optimal solution.

```
Cuckoo Search based on ILF Algorithm
Input: Population of the problem, $p_a$ , IR
Output: Best Solution
Begin
  Objective function F(x), x=($x_1$, …, $x_d$)$^T$
  Generate initial population of n host nests $x_i$ (i = 1, 2, …, n)
  While (t <Max Generation) or (not stop criterion)
    For I = 1 to k      (k=IR*nest no.)
      Generate new solution by levy flight say i
      Evaluate its quality/fitness $F_i$
      Choose a nest among n (say, j) randomly
      if ($F_i$ > $F_j$)
        replace j by the new solution;
    next I
    A fraction ($p_a$)/2 of worse nests are abandoned and new
      ones are built randomly;
    A fraction ($p_a$)/2 of worse nests are abandoned and new
      ones are built by call BNG (($p_a$)/2, current best);
    Keep the best solutions (or nests with quality solutions);
    Rank the solutions and find the current best
  end while
  Post process results and visualization
End
```

Fig. 7 Improved Cuckoo Search Algorithm based on iterative Levy flight.

## VII. EXPERIMENTAL RESULTS

FJSSP is a combinatorial problem that used as a tool to measure the performance of the two improved methods.

Cuckoo Search is an efficient and robust optimization method in continues state space. Therefore, some adaptation proposed for applying its efficiency in the discrete state space.

CS and its improvements were implemented in MATLAB on a computer that has Intel Core2 Duo P7350, 2.0 GHz processor with 4 GB Memory (RAM) and 300 GB hard drive. The algorithms have been applied on Hurink et al.[15,16] instances with three different instance sets “edata”, “rdata”, and “vdata” which the set of assignable machines with a particular probability distribution has been expanded . Ten independents run have been performed on each algorithm and the outcomes appeared in Table I.

The difference between CS and its two improvements for finding better makespan was clearly shown in Fig. 8.

TABLE 1
AVERAGE OPTIMALITY FOR CS AND ITS IMPROVEMENTS

| Instances | Lower Bound [15] | Average Makespan of CS | Average Makespan of Proposed CS -BNG | Average Makespan of Proposed CS-ILF |
|---|---|---|---|---|
| edata-mt06 | 55 | 56 | 55 | 55 |
| edata-mt10 | 871 | 1191 | 986 | 979 |
| edata-la1 | 609 | 729 | 636 | 634 |
| edata-la2 | 655 | 783 | 707 | 694 |
| edata-la3 | 550 | 667 | 593 | 588 |
| edata-la4 | 568 | 709 | 620 | 619 |
| edata-la5 | 503 | 605 | 525 | 526 |
| edata-la6 | 833 | 976 | 864 | 861 |
| edata-la7 | 762 | 960 | 818 | 819 |
| edata-la8 | 845 | 1001 | 880 | 868 |
| rdata-mt06 | 47 | 55 | 50 | 50 |
| rdata-mt10 | 679 | 1067 | 802 | 801 |
| rdata-la1 | 570 | 723 | 607 | 609 |
| rdata-la2 | 529 | 680 | 573 | 567 |
| rdata-la3 | 477 | 621 | 518 | 512 |
| rdata-la4 | 502 | 646 | 542 | 538 |
| rdata-la5 | 457 | 577 | 484 | 480 |
| rdata-la6 | 799 | 974 | 832 | 821 |
| rdata-la7 | 749 | 917 | 779 | 776 |
| rdata-la8 | 765 | 938 | 793 | 790 |
| vdata-mt06 | 47 | 55 | 49 | 48 |
| vdata-mt10 | 655 | 1000 | 746 | 729 |
| vdata-la1 | 570 | 728 | 613 | 609 |
| vdata-la2 | 529 | 675 | 565 | 564 |
| vdata-la3 | 477 | 627 | 515 | 520 |
| vdata-la4 | 502 | 652 | 534 | 531 |
| vdata-la5 | 457 | 587 | 485 | 499 |
| vdata-la6 | 799 | 981 | 826 | 821 |
| vdata-la7 | 749 | 941 | 774 | 773 |
| vdata-la8 | 765 | 952 | 779 | 787 |

In our experiment, many tests have been applied for the algorithms to determine the suitable values of the algorithms parameters. The fraction of the worst nest Pa=0.4, IR=0.2.

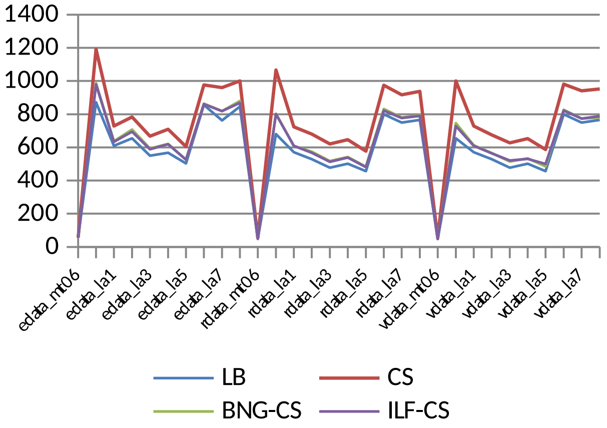

Fig. 8 Difference between CS and Its improvements.

Table II shows the difference between the algorithm and its improvements in the iteration number that formatted in bold text.

To see clearly the difference in iteration number between CS and its improvement see Fig. 9. The two improved methods are better than the pure CS in finding higher quality solutions in less number of iteration for most instances.

From Table II, we see the two improvement speed up the convergence rate of the basic CS algorithm to find better solutions.

TABLE 2
COMPARISON OF CS AND ITS IMPROVEMENTS FOR AVERAGE NUMBER OF ITERATION

| **instances** | **Average of iteration number in CS** | **Average of iteration number in CS- BNG** | **Average of iteration number in CS-ILF** |
|---|---|---|---|
| edata-mt06 | 444 | **39** | **69** |
| edata-mt10 | 452 | **718** | **466** |
| edata-la1 | 475 | **352** | **369** |
| edata-la2 | 540 | **316** | **293** |
| edata-la3 | 507 | **399** | **477** |
| edata-la4 | 584 | **432** | **433** |
| edata-la5 | 653 | **370** | **199** |
| edata-la6 | 498 | **519** | **460** |
| edata-la7 | 546 | **674** | **490** |
| edata-la8 | 488 | **576** | **550** |
| rdata-mt06 | 482 | **136** | **406** |
| rdata-mt10 | 569 | **726** | **792** |
| rdata-la1 | 676 | **532** | **437** |
| rdata-la2 | 528 | **498** | **638** |
| rdata-la3 | 503 | **527** | **615** |
| rdata-la4 | 397 | **538** | **596** |
| rdata-la5 | 387 | **577** | **651** |
| rdata-la6 | 591 | **393** | **601** |
| rdata-la7 | 458 | **440** | **692** |
| rdata-la8 | 595 | **402** | **482** |
| vdata-mt06 | 449 | **86** | **132** |
| vdata-mt10 | 648 | **713** | **678** |
| vdata-la1 | 425 | **497** | **643** |
| vdata-la2 | 510 | **656** | **581** |
| vdata-la3 | 468 | **586** | **450** |
| vdata-la4 | 412 | **529** | **577** |
| vdata-la5 | 540 | **592** | **363** |
| vdata-la6 | 425 | **769** | **650** |
| vdata-la7 | 692 | **606** | **681** |
| vdata-la8 | 479 | **650** | **520** |

Fig 9, illustrate the difference between CS and Its improvements in the number of iteration. Most of the result which gets from the second improvement obtained more quality solution comparing with the first one. In terms of the speed, a third of the benchmark instances results from the second improvement are slower than the first one with a slightly increased the iteration number.

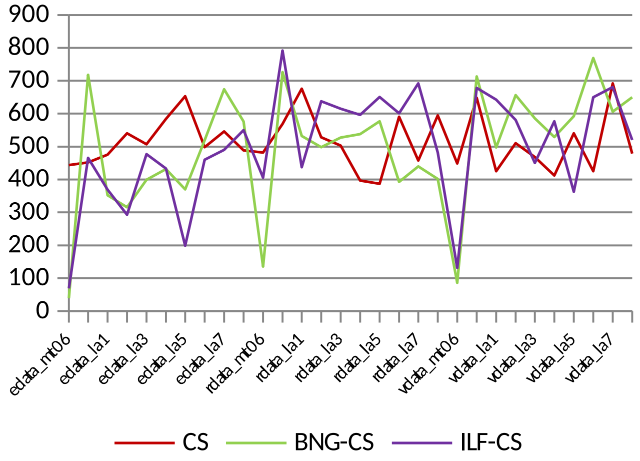

Fig. 9 Average of iteration in CS and Its improvements.

## VIII. CONCLUSION

This paper presented two improved CS algorithms. In the first improvement, the division of the generating solutions instead of the deleted ones speeds up the convergence rate and maintaining a good amount of diversification. Levy Fight gives the effectiveness of CS algorithm, therefore, we proposed, in the second improvement, to repeat it multiple times in an acceptable ratio (IR). From the experiments, we found (0.2) of the iterated Levy flight is a suitable ratio for obtaining feasible solutions and increasing this ratio has a time consuming for finding better solutions. The Experimental results show that the two improvements have higher quality solutions and speed up the convergence rate in most instances in comparison with the basic cuckoo search algorithm.

## REFERENCES

[1] M. L. Pinedo, *Scheduling: Theory, Algorithms, and Systems*, 4th ed., Springer-Verlag, 2012.

[2] Yao Wang, LinBo Zhu, Jiwen Wang, and Jianfeng Qiu, “An Improved Social Spider Algorithm for the Flexible Job-Shop Scheduling Problem,” *Estimation, Detection and*

*Information Fusion (ICEDIF) International Conference on. IEEE*, pp. 157-162, 2015.

[3] Ling Wang, Gang Zhou, Ye Xu, Shengyao Wang, and Min Liu, "An effective artificial bee colony algorithm for the flexible job-shop scheduling problem", *The International Journal of Advanced Manufacturing Technology, Springer-Verlag*, vol. 60, Issue 1, pp. 303-315, 2012.

[4] Jin Feng Wang, Bi Qiang Du, and Hai Min Ding, "A Genetic Algorithm for the Flexible Job-Shop Scheduling Problem" *Advanced Research on Computer Science and Information Engineering International Conference (CSIE) Part I*, pp. 332-339, 2011.

[5] Shekhar Singh and Krishna Pratap Singh, "Cuckoo Search Optimization for Job Shop Scheduling Problem", *Advances in Intelligent Systems and Computing, Proceedings of Fourth International Conference on Soft*

[10] Thang Trung Nguyena and Dieu Ngoc Vo, "Modified cuckoo search algorithm for short-term hydrothermal scheduling", *International Journal of Electrical Power & Energy Systems, Elsevier*, vol. 65, pp. 271–281, 2015.

[11] Arit Thammano and Ajchara Phu-ang, "A Hybrid Artificial Bee Colony Algorithm with Local Search for Flexible Job-Shop Scheduling Problem", *Procedia Computer Science 20, Elsevier*, pp. 96-101, 2013.

[12] Nouiri M., Jemai A., Ammari A. C., Bekrar A., and Niar S., "An effective particle swarm optimization algorithm for flexible job-shop scheduling problem", *Industrial Engineering and Systems Management (IESM), Proceedings of 2013 International Conference on. IEEE*, pp. 1-6, 2013.

[13] Wei Zhou, Yan-ping Bu, and Ye-qing Zhou, "Combining CA and PSO to solve flexible job shop scheduling

*Computing for Problem Solving, Springer*, pp. 99-111, 2015.

[6] El-Ghazali Talbi, *Metaheuristics from Design to Implementation*, John Wiley & Sons, 2009.

[7] Hongqing Zheng and Yongquan Zhou, "A Novel Cuckoo Search Optimization Algorithm Base on Gauss Distribution", *Journal of Computational Information Systems*, vol. 8, no. 10, pp. 4193- 4200, 2012.

[8] Yongquan Zhou, Hongqing Zheng, Qifang Luo, and Jinzhao Wu, "An improved Cuckoo Search Algorithm for Solving Planar Graph Coloring Problem", *Applied Mathematics & Information Sciences,* vol. 7, no. 2, pp. 785-792, 2013.

[9] Pauline Ong, "Adaptive Cuckoo Search Algorithm for Unconstrained Optimization", *the Scientific World Journal,* vol. 2014, 2014.

problem", *Control and Decision Conference (2014 CCDC), The 26th Chinese. IEEE*, pp. 1031-1036, 2014.

[14] X. S. Yang and S. Deb, "Cuckoo search via Lévy flights", *World Congress on Nature & Biologically Inspired Computing (NaBIC 2009).IEEE Publications*, pp. 210–214, 2009.

[15] Behnke, D., Geiger, M.J, "Test instances for the flexible job shop scheduling problem with work centers," Helmut-SchmidtUniversit¨at, Hamburg, Germany. Rep. RR-12-01-01, 2012.

[16] M. Mastrolilli, "Flexible Job Shop Problem", Dalle Molle Institute for Artificial Intelligence (IDSIA), The University of Applied Sciences and Arts of Southern Switzerland (SUPSI). [Online]. Available: http://www.idsia.ch/~monaldo/fjsp.html