Resolving Set-Streaming Stream-Shop Scheduling in Distributed System by mean of an aFOA

Anurag Rana
Department of Computer Science and Engineering,
Arni School of Technology.
ARNI University.

Ankur Sharma
Department of Computer Science and Engineering,
Arni School of Technology.
ARNI University.

Abstract. Recently, a new fruit fly optimization algorithm (FOA) is proposed to solve stream-shop scheduling. In this paper, we empirically study the performance of FOA. The experimental results illustrate that FOA cannot solve set-streaming stream-shop scheduling in distributed system with same-size sub-sets effectively. In order to enhance the performance of FOA, an amended FOA (named aFOA) is proposed. Numerical testing proves and comparisons of aFOA with FOA and GA show that aFOA can greatly enhance the scheduling efficiency and greatly improve the scheduling quality. The resolving the set-streaming stream-shop scheduling in distributed system (SSSS) with same-size sub-sets by mean of an amended fruit fly optimization algorithm (aFOA) is intended in this paper. In the intended aFOA, a result is delineated as two vectors to find the dividing of tasks and the sequence of the sub-sets simultaneously. An aFOA is based on the encoding system three kinds of neighborhoods are developed for generating new results. To considerably balance the development and exploration, including the neighborhood-based search (smell-vision-based search) and the global cooperation-based search, two main search processes are designed within the evolutionary search model of the aFOA. Finally, on the basis of numerical testing results are provided, and the comparisons demonstrate the effectiveness.

Keywords: fruit fly optimization algorithm, set-streaming stream-shop scheduling, task dividing, neighborhood-based search, global cooperation-based search.

I INTRODUCTION

The set-streaming stream-shop scheduling is an important application in distributed practical production system. The SSSS is inspired from classical flow-shop scheduling problem and it is the propagation. In traditional distributed system n task are intended on m system in a given sequence. In a stream with sub-streaming, each task consist of many identical process which can be divide into several sub-sets, and then all these sub- sets will be processed on m systems one after another. Therefore the scheduling should decide the dividing of each task as well as the sequence of all the sub-sets.

In ahead of time, researchers concentrated on the problems with only few systems and a single task. Heuristic and operational research analytical methods have been developed to solve these problems. Poter and Baker theoretically obtained the optimal sizes of sub-sets for a single task processed over two systems [3]. Trietsch and Baker presented a linear programming method for single task problems [4]. Heuristic Johnson's rule [5] and bottleneck minimal idleness [6] were implemented. During the past decades, more complex scheduling problems have been studied and meta-heuristic methods have been used frequently. For SSSS with same size sub-sets, genetic algorithm (GA) [1, 8], ant colony optimization (ACO) and threshold accepting (TA) [2], discrete particles swam optimization (DPSO) [10]. Discrete artificial bee colony algorithm (DABC) [12] and estimation of distribution algorithm (EDA) [14] have been inquired.

Fruit fly optimization algorithm (FOA) is a novel optimization approach, which is inspired by the knowledge from the food finding behavior of the fruit flies [17]. Recently the fruit flies optimization algorithm has been applied for different problem including linear generation mechanism [21], PID controller tuning [17], power load forecasting [18] , web auction logistics services [19] and financial distress [20].

The FOA is easily implemented and has potential to solve the complex scheduling problems. To best of our knowledge, the FOA has been applied to solve the SSSS yet. Thus, in this paper we shall intend amended FOA (aFOA) for the SSSS. We encode the result as to vectors to delineate the dividing of tasks and the sequence of sub-sets at a time. Within search model of the FOA, we assume two main search processes including neighborhood-based search and global cooperation-based search to considerably balance the

development and exploration on the basis of numerical testing result and comparisons demonstrate the effectiveness of intended aFOA for SSSS.

The paper is organized as, firstly the SSSS is described in section II and basic of FOA is introduced in section III. Then, aFOA is presented in section IV and numerical testing and comparisons are provided in section V. Finally, we end the paper with few conclusions and future work.

II SET-STREAMING STREAM-SHOP SCHEDULING

The SSSS discussed in this paper is to minimize the completion time for an n tasks and m systems set streaming stream scheduling with same-size sub-sets. The problem is described as:-

Each task y ∈ {1, 2, 3,……., n} has D_y identical units that will be divide into S_y sub-sets with same size, where S_y is a variable to be optimized. Let the processing time of each unit of task y on system x be T(x,y). The sub-sets of different tasks can be blending and will be sequentially processed on m systems and the sub-sets sequence is t5he same on each system. If two consecutive sub-sets have different task-types, then the second sub-set consumes a set –up time of S_1 (x,y) on its arrival system on the other hand if they have the same task-type, the second sub-set consumes a time of S_2 (x,y). Suppose that the release time of all tasks is zero and the buffer space is infinite. Besides, sub-set transit time is included in the processing time. The objective of scheduling into minimize the completion time.

III FRUIT FLY OPTIMIZATION ALGORITHM

The fruit fly optimization algorithm (FOA) is a new swarm intelligence algorithm, which was intended by Pan [19] in 2011. It is a kind of interactive evolutionary computation method. FOA can reach the global optimum by imitating the food finding behavior of fruit fly. Fruit fly is superior to other species in vision and osphresis (as exemplify in Figure 1). The food finding process of fruit fly is as follows: it firstly smells the food source with its osphresis organ and flies towards that location; then, after it gets close to the food location, its sensitive vision is also used for finding food and other fruit flies flocking location, then it flies towards that direction. The FOA has been applied to several fields including traffic incidents [15], export trade forecasting [16] and the design of analog filters [13]. The procedures of fruit fly group's food finding behavior is exemplify in Figure 2 [18].

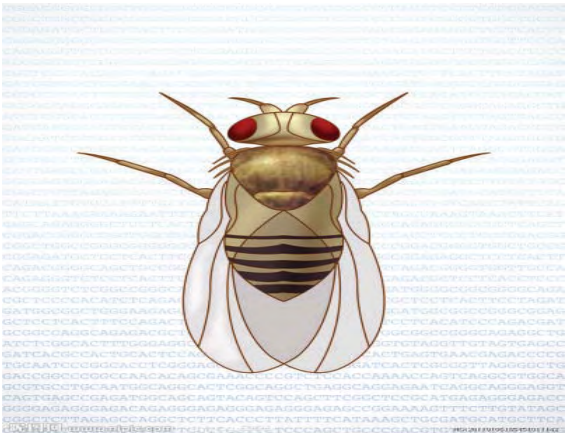

Figure 1 Fruit Fly.

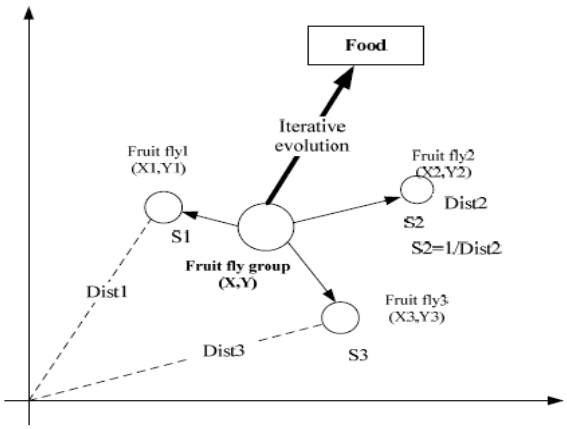

Figure 2 Iterative Scrounging Process of Fruit Fly.

According to food finding characteristics of fruit fly swarm, the whole procedure of the original FOA can be divided into several step as follows:-

Step 1 Start:

Start the algorithm to reach the maximum number of generation.

Step 2 Parameter Initializations:

The main parameters of FOA are maximum iteration number, the population size, the random initialization fruit fly swarm location range and random fly direction and distance zone of fruit fly. Randomly initialize the location of the fruit fly swarm.

Step 3 Population Initializations:

Randomly assign each and every fruit fly a direction and distance for their movement to search for food using osphresis to generate a new population.

Step 4 Population Evaluations:

Evaluate all the new individuals.

Step 5 Selection Operations:

Identify the best fruit fly with the maximum smell concentration value (i.e. the best objective), and then the fruit fly group flies towards the best location utilizing vision.

Step 6 End:

End the algorithm if the maximum number of generation is reached; otherwise, go back to step 3.

IV aFOA FOR SSSS

The design of aFOA will be described in detail. First, we will describe the encoding system; second, we introduce the population initialization way; third, we design two main search process including the neighborhood-based search and global cooperation-based search. Finally, we exemplify the procedure of aFOA.

IV (a) Encoding System:

In the aFOA, each fly is a result of the SSSS, which is delineated by two A-bits integer vectors, where A is maximum number of sub-sets of all the tasks. Let, there are 20 tasks and each task can be divide into 1~10 sub-sets, therefore A will be equal to 200.

The result consists of A units and delineates a sequence of A sub-sets. The first vector delineates the task-type and second delineates the size of each sub-set. Formula to calculate the size of sub-set x of task y is given below:

$$\begin{cases} Q_{xy} = D_y \mid S_y \quad x=1,2,3\ldots\ldots..,S_{y-1}. \\ Q_{xSx} = D_{y--} \sum_{y-1}^{Sy-1} Q_{XY.} \end{cases}$$

Above encoding system is exemplified in Figure 3. Generally, the sub-set number is less than A. If no real sub-set exists in unit, the task type of that unit would be set to -1 while the size of the unit is set to 0.

2	1	2	2	3	1	3	-1
2	4	2	3	3	5	2	0

Figure 3 Encoding System.

IV (b) POPULATION INITIALIZATION

Following rules are developed to initialize the population. First rule is to generate blending results. For these results, sub-sets of different tasks can be blending with each other. The second rule is to generate un-blending results, in which sub-sets of the same tasks should be produced consecutively. These rules are applied randomly and thus we get an initial population with NP fruit flies, each of which is delineated by 2*A integer vectors.

IV (c) NEIGHBORHOOD-BASED SEARCH

In neighborhood-based search stage, namely smell vision based search process, NN neighbor are randomly generated around each fruit fly a_y, $y \in \{1,2,3,\cdots\cdots NP\}$ based on three kinds of neighborhood structures. The first kind of neighborhood structure is the blending insert, which is exemplified in Figure 4. A sub-set is randomly selected and then it is inserted into another size. This may cause an additional set-up time $S_1(x,y)$.

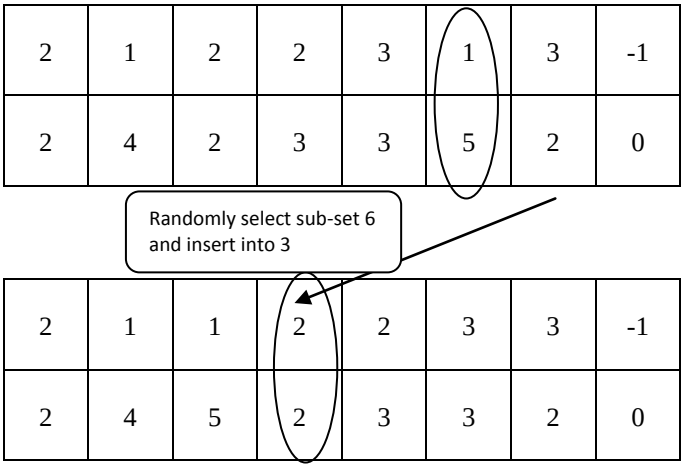

Figure 4 An Exemplify of Blending Insert.

Second kind of neighborhood structure is un-blending insert, which is exemplified in Figure 5. Among the whole sequence, some sub-sets consecutively produced may have the same task types. They share a set-up time (S_1 (x,y) together and we consider these sub-sets as whole called a "sub-task". The sequence of sub-sets could be considered as a sequence of sub-sets with a smaller number then sub-sets. To carry out the insert operator, a sub-task is selected and then it is inserted next to another sub-task. This insert may not cause an additional setup time.

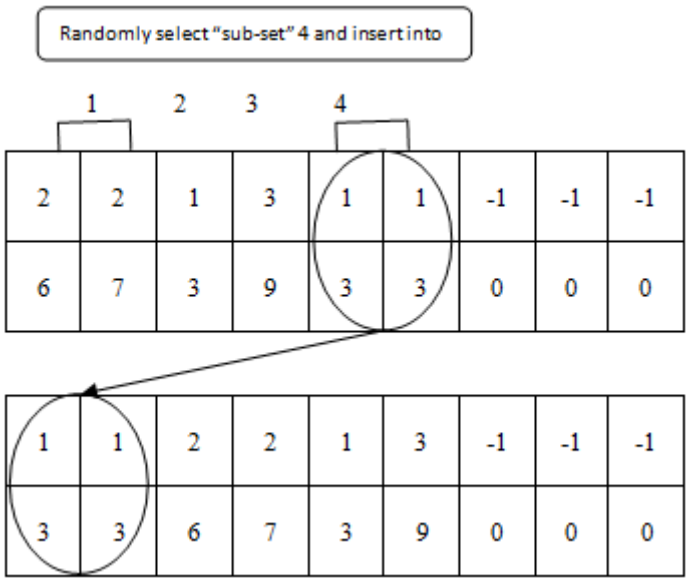

Figure 5 An Exemplify of Un-blending Insert.

Third kind of neighborhood structure is sub-set number-varied search which exemplified in Figure 6 and 7. First a task is randomly selected and then the number of its sub-sets increase and decrease by one.

ALGORITHM FOR INSERT SUB-SET

Step 1: Select task randomly.

Step 2: If this task number of sub-set reaches maximum, then it ends, else go to next step.

Step 3: Select the sub-set of the task randomly.

Step 4: Insert a new sub-set of the task and calculate the new size of each sub-set.

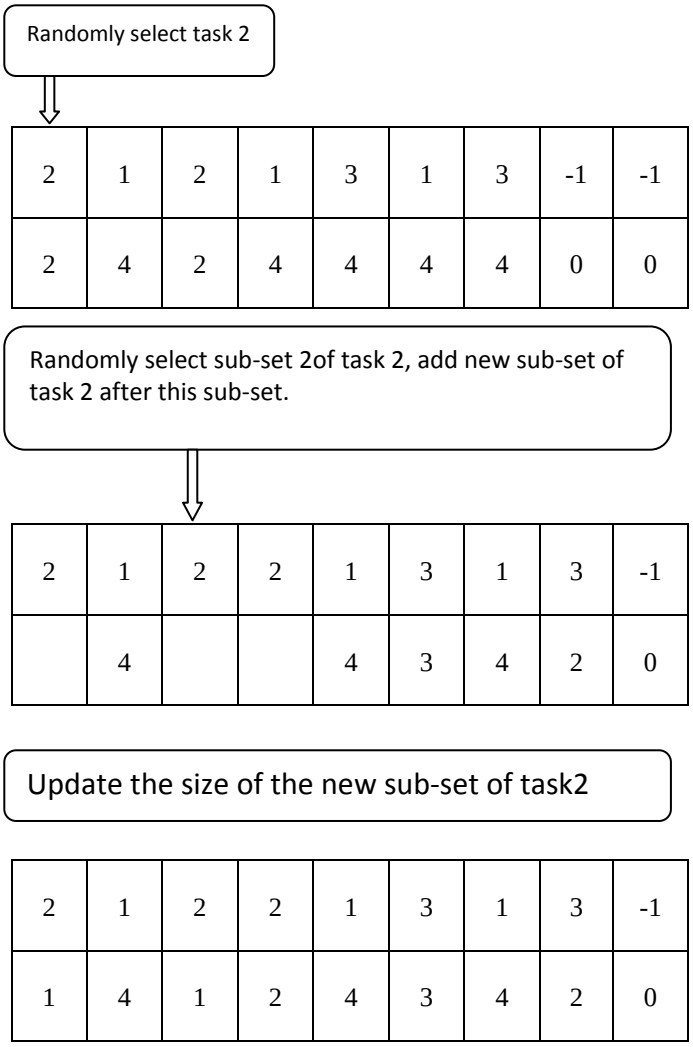

Figure 6 An Exemplify of Increasing Number Search.

ALGORITHM FOR DELETE SUB-SET

Step 1: Select the task randomly.

Step 2: If the number of sub-set reaches minimum, then ends, else go to next step.

Step 3: Randomly select a sub-set of the task.

Step 4: Remove the sub-set and calculate the new size of each sub-set.

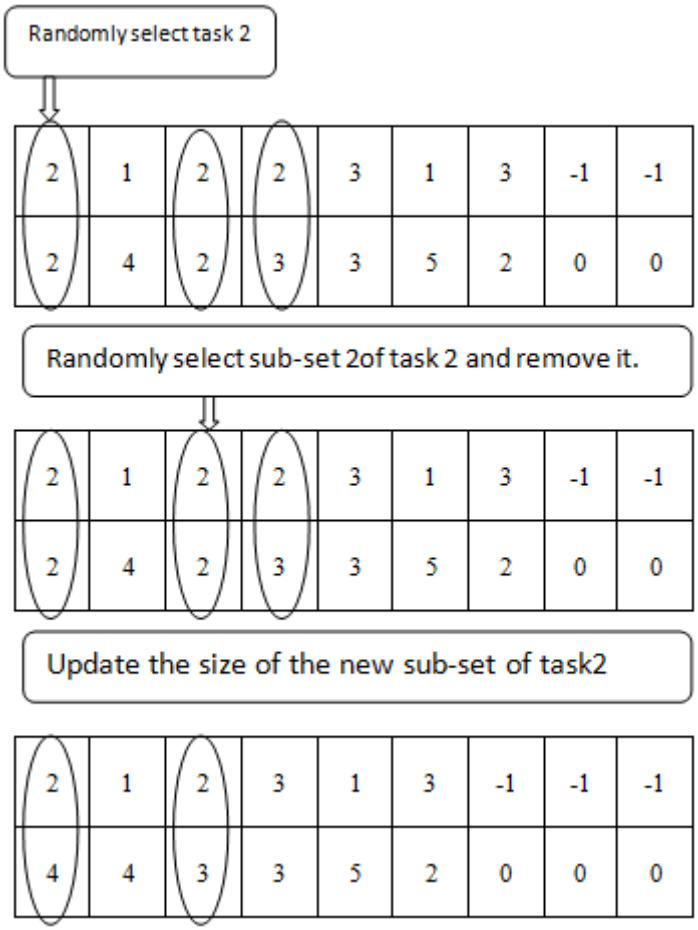

Figure 7 An Exemplify of Decreasing Number Search.

Third kind of neighborhood structure may cause an additional setup time S_2 (x,y), where as it cannot cause an additional setup time S_1 (x,y).

Above three kinds of neighborhood structure will be selected with probability 0.4, 0.4 and 0.2 respectively, according to the experience of some prior experiments. After NN new individuals are produced based on the three kinds of neighborhood structure, the individual x_y finds its best local neighbor using its sensitive vision. If the best neighbor is better, then it flies towards the best neighbor. Otherwise, it remains unchanged.

Once the whole population completes NG times of neighborhood-based search, it will move to the following global cooperation-based search. In the neighborhood-based search, each fruit fly constantly searches new results around itself and there is no communication taken place between different individuals.

IV (d) GLOBAL COOPERATION-BASED SEARCH

To enhance the global exploration, the fruit fly with better performance will guide the flies with worse performance towards a suitable flying direction. The detailed procedure is given below:-

Step 1: Sort the individuals among the population from the best to the worst.

Step 2: Divide the population in two halves. Each individual in the half with poor fitness performs crossover with its corresponding fly in the other half with good fitness. For instance, the group has 10 fruit flies, the first fly has the best fitness and the last one has worst. Then crossover the 5^{th} fly with the first and 6^{th} with second one and so on.

Step 3: After performing the crossover operator, if the ne individual is better than the original one, then its replace the original one otherwise it keeps the original one unchanged.

Above crossover is a task based operator. An example exemplified in Figure 8. The new fly will inherit the sites of sub-sets of one task randomly selected from bad fly while it inherit the site of sub-sets of all the other tasks from the excellent fly. In each a way, a group of poor flies may fly towards the better ones. Thus, the search may have the potential to concentrate on the location with better quality among the global area.

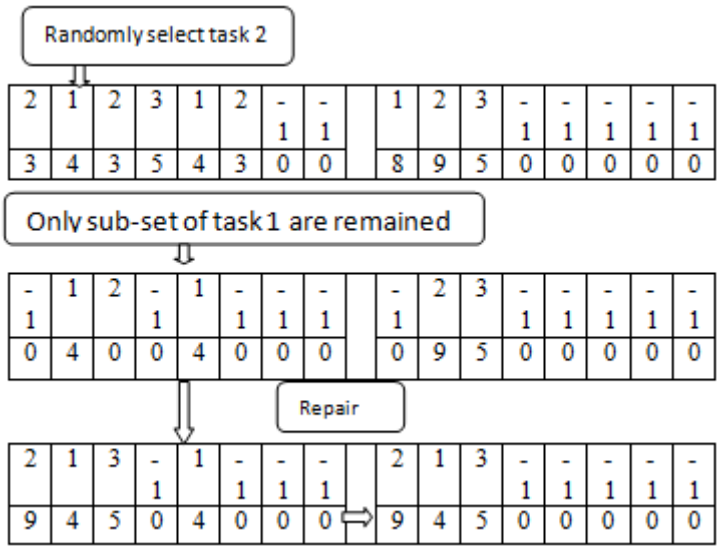

Figure 8 Task-based Crossover Operators.

IV (e) aFOA SEARCH MODEL

The search model of aFOA is exemplified in figure 9 for aiming to resolving the distributed system SSSS. In addition to the population initialization with following rules, the aFOA performs search process by using both neighborhood-based and global cooperation-based search.

ALGORITHM FOR aFOA

Step 1 Start:

Start the algorithm to resolve the set-streaming stream-shop schedules.

Step 2 Parameter Initializations:

The main parameters of aFOA are maximum iteration number, the population size.

Step 3 Population Initializations:

Randomly initialize the population size is NP. Randomly produce solution, half blending and un-blending.

Step 4 Neighborhood Generations and Find Best Neighborhood:

NN neighborhoods are randomly generated. NN neighborhoods are produced by three kinds of neighborhood search. Find out the best neighbor.

Step 5 Replacements of Neighbor:

If the best neighbor is better than the fly then replace the fly with the best neighbor and go to next step, else go to next step without replacing.

Step 6 Local Neighborhood Loop Search:

If the loop termination is reached then sort the population else go back to step 4.

Step 7 Crossovers:

In global cooperation search each flies in the poor half crossover it with the corresponding one. If new fly is better than poor fly then replace the poor fly with new one and go to next step, else go to next step without replacing.

Step 8 Termination Criterions:

If termination standard is reached then provide result, else go back to step 3.

Step 9 End:

End the algorithm with solution of set-streaming stream-shop schedules.

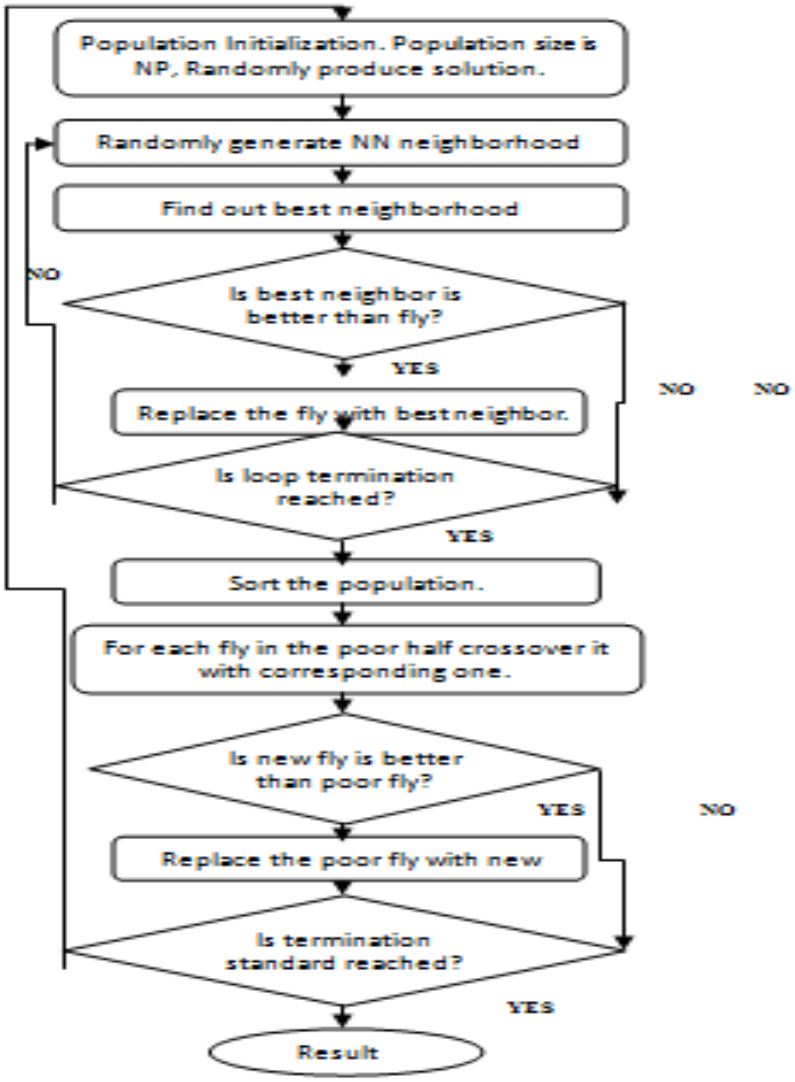

Figure 9 Flow Chart of aFOA.

V PROVES AND COMPARISIONS

As best of our knowledge, there is no benchmark instance for the distributed system SSSS. In this paper, two set of instance are generated randomly, including 3 instances with 9-Tasks, 9-Systems and 3 instances with 40-Tasks and 2-Systems. The associated data are randomly discrete integer with consistent distribution. To be particular:

$D_y \in U[45, 90]$, $T(x, y) \in U[1, 20]$ and

$S_1(x, y) \in U[1,5]$. In [8] $S_y \in U[1,8]$ and $S_2(x, y)$ is not considered. Here, we assume $S_2(x, y) = 3$. To make a fair comparison, the entire algorithm take the equal maximum CPU time limit of 100 * m * n ms as the stopping standard.

Parameter of aFOA are set as NP=60, NN=30 and NG=30. For each instance, 30 independent replications are carried out. We code the algorithm in java and run it on a system.

V (a) COMPARISONS OF aFOA WITH GA

We compare the aFOA with the GA [8]. The comparative results are listed in Table 2 for all the instances with two different scales.

The comparative results with the 9*9 small-scale instances and 40*20 large-scale instances are listed in Table 1, where Best and Worst represent the best and the worst results among the 30 replications, Avg represents the average of the 30 replication results, and SD represents the standard deviation of the results.

It can be seen that the proposed algorithm is more powerful in resolving the SSSS. In fact, the aFOA performs better than the GA. From [8], it is known that the genetic algorithm is two-level GA, where the upper-level GA optimizes the number of sub-lots for each product determines sub-lots sizes and the lower-level GA optimizes the sub-lot intermingling scheduling. The evaluation of individuals of the upper-lever GA is based on

the optimization process of the lower-level GA. In contrast, the encoding system of our aFOA allows more flexible and pliable neighborhood-based search process to achieve better and more effective results.

n*m	aFOA				GA			
	Best	Worst	Avg	SD	Best	Worst	Avg	SD
9*9	6629	6676	6648.1	14.6	8467	8558	8495.3	26.4
9*9	7664	7703	7667.9	11.7	9703	9815	9776.5	32.7
9*9	8467	8558	8495.3	26.4	9716	9916	9762.2	47.05
40*20	28148	28275	28184.1	46.5	26516	26944	26730.1	151.4
40*20	28528	28676	28576.8	54.1	28930	29379	29122	158.9
40*20	29116	29386	29245.6	104	29824	300431	30061.5	180.7

Table 1 Comparison of aFOA with GA on SSSS.

VI CONCLUSIONS

The major contribution of this paper is to resolving set-streaming stream-shop scheduling. To enhance the effectiveness of search algorithm, we designed a special encoding system and two kinds of search processes, including the multiple neighborhoods based search and the global cooperation search. Two rules were also applied for initializing the population. With the results of proves and comparisons, the effectiveness of the aFOA was demonstrated in resolving the SSSS with same-size sub-sets. Besides, more complex scheduling such as the SSSS with variable-size sub-sets will be challenging work. The future work is to design the self-adaptive FOA where some knowledge in using neighborhoods and adjusting parameters may be utilized.

ACKNOWLEDGMENTS

The Authors would like to acknowledge the cooperation of all the supporting members. We would like to thank the anonymous reviewers for their helpful comments that helped improving the quality of the paper.

REFERENCES

[1] Marimuthu S, Ponnambalam SG, Jawahar. Evolutionary algorithms for scheduling m-machine flow shop with lot streaming. Robotics and Computer-Integrated Manufacturing, 2008, 24(1): 125-139.
[2] Marimuthu S, Ponnambalam SG, Jawahar. Threshold accepting and ant-colony optimization algorithm for scheduling m-machine flow shop with lot-streaming. Journal of Materials Processing Technology, 2009, 209(2): 1026-1041.
[3] Potts CN, Baker KR. Flow shop scheduling with lot streaming. Operations Research Letters, 1989, 8(6): 297-303.
[4] Trietsch D, Baker KR, Basic techniques for lot streaming, Operational Research, 1993, 41: 1065-1076.
[5] Vickson RG. Optimal lot streaming for multiple products in a two-machine flow shop. European Journal of Operational Research, 1995, 38(1-2): 57-67.
[6] Kalir AA, Sarin SC. Constructing near optimal schedules for the flow-shop lot streaming problem with sublot-attached setups. Journal of Combinatorial Optimization, 2003, 7(1): 23-44.
[7] Yoon SH, Ventura JA. Minimizing the mean weighted absolute deviation from due dates in lot-streaming flow shop scheduling. Computer & Operations Research, 2002, 29(10): 1301-1315.
[8] Wang J, Zhou H. Integrated optimization of lot streaming and sublot-intermingling scheduling for a kind of flow shop. Journal of System Simulation, 2008, 20(4): 1011-1015.
[9] Sang H. A discrete differential evolution algorithm for lot-streaming flow shop scheduling problems. Sixth International Conference on Natural Computation (ICNC), 2010, 10-13.
[10] Tseng CT, Liao CJ. A discrete particle swarm optimization for lot-streaming flow shop scheduling problem. European Journal of Operational Research, 2008, 191(2): 360-373.
[11] Pan QK, Wang L, Gao L, Li J. An effective shuffled frog-leaping algorithm for lot-streaming flow shop scheduling problem. International Journal of Advanced Manufacturing Technology, 2011, 52: 699-713.
[12] Pan QK, Suganthan PN, Tasgetiren MF, Chua TJ. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. Information Sciences, 2011, 181(12): 2455-2469.
[13] Xiao, Z.A. Design of analog filter based on fruit fly optimization algorithm. J Hubei Univ. Educ.2012, 29, 26–29.
[14] Pan QK, Ruiz R. An estimation of distribution algorithm for lot-streaming flow shop problems with setup times. Omega - International Journal of Management Science, 2012, 40: 166-180.
[15] Shi, D.Y.; Lu, J.; Lu, L.J. A judge model of the impact of lane closure incident on individual vehicles on freeways based on RFID technology and FOA-GRNN method. J.Wuhan Univ.Technol. 2012, 34, 63–68.
[16] Xu, Z.H.; Wang, F.L.; Sun, D.D.; Wang, J.Q. A forecast of export trades based on the FOA-RBF neural network [in Chinese]. Math. Pract. Theor. 2012, 42, 16–21.
[17] Han J, Wang P, Yang X. Tuning of PID controller based on fruit fly optimization algorithm. Internaional Conference on Mechatronics and Automation (ICMA), 2012, 409-413.
[18] Li H, Guo S, Li C, Sun J. A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm. Knowledge-Based Systems, 2013, 37: 378-387.

[19] Lin SM. Analysis of service satisfaction in web auction logistics service using a combination of fruit fly optimization algorithm and general regression neural network. Neural Computing & Applications, 2013, 22(3-4): 783-791.
[20] Pan WT. A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowledge-Based Systems, 2012, 26(2): 69–74.
[21] Dan Shan, GuoHua Cao, and Hongjiang Dong. LGMS-FOA: An improved fruit fly optimization Algorithm for solving optimization problems. Hindawi Publishing Corporation Mathematical Problems in Engineering. Volume 2013, Article Id 108768, 9 pages.

First Author is currently employed at University institute of information Technology (UIIT) in Himachal Pardesh University (HPU) Shimla-5. He acquired M. Tech. CSE from Arni School of Technology and Master of Computer Application from Arni School of Computer Science in Arni University (HP). His special interests include Artificial Intelligence, Neural Network, and Distributed System/Network.