International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 9 (2017) pp. 173-180

Performance Optimization of the Paper Mill using Opposition based Shuffled frog-leaping algorithm

Tarun K. Sharma
Amity School of Engineering and Technology,
Amity University Rajasthan, Jaipur, India
taruniitr1@gmail.com

***Abstract*: Shuffled frog-leaping algorithm (SFLA) is recently introduced memetic algorithm inspired by foraging behavior of frogs. SFLA partially follows particle swarm optimization in local search process and shuffled complex evolution algorithm in performing global search. The key concept about such algorithms is to gain an edge over traditional or deterministic mathematical techniques to achieve comparatively better solutions to the multimodal or multifaceted optimization problems. SFLA embeds the features of both particle swarm optimization (PSO) and shuffled complex evolution (SCE) algorithm. In this study SFLA named as O-SFLA is proposed. In general structure of SFLA, the frogs are divided into memeplexes based on their fitness values where they forage for food. In this study the opposition based learning concept is embedded into the memeplexes before the frog initiates foraging. The proposal is validated on performance optimization of the Paper Mill.**

***Keywords*:** Shuffled frog-leaping algorithm, SFLA, Optimization, Convergence, Opposition based learning, paper & pulp**.**

I. Introduction

The solution(s) for the complex optimization problems, where nontraditional or deterministic methods are not able, can be searched using nature or biological inspired optimization algorithms. Nature or biologically inspired optimization algorithms are widely acceptable algorithms to solve the problems, where analytical methods fail. SFLA, is one of the recently addition to the biologically inspired algorithm.

The problem of optimization arises in every sphere of human life. As the level of problem increases in terms of complexity, dimension, limited knowledge about the problem domain, feasible region and a like, it becomes difficult to solve such multifaceted problems using traditional optimization methods (e.g. gradient search methods etc.). SFLA, introduced in 2003 by Eusuff and Lansey [1][2], is a recent member of memetic algorithms family. SFLA, like other memetic algorithms, enthused by natural foraging behavior of species. In SFLA these natural species are frogs. Since introduction, SFLA and its variants have been successfully applied to solve optimization problems lies in the domain of engineering and management. The same can be witnessed from the literature. SFLA embeds the features of both particle swarm optimization (PSO) and shuffled complex evolution (SCE) algorithm. PSO helps in performing local search while SCE helps in global search. In SFLA, like other memetic algorithms, colony of frogs is initialized randomly and in second step the colony is divided into sub colonies or memeplexes.

SFLA performs both exploration by dividing the randomly initialized population of frogs into number of memeplexes and exploitation by performing evolution process in each memeplexes. Although SFLA emerges as successful optimizer however it also suffers in global convergence velocity. In this study a modification in the memeplexes is introduced by embedding the concept of opposition based learning. In general structure of SFLA, the frogs are divided into memeplexes based on their fitness values where they forage for food. In this study the opposition based learning concept is embedded into each memeplexes before the frog initiates foraging. After each iteration when information exchange process among memeplexes is performed, OBL is again introduced to improvise global search. The proposal is named as O-SFLA. This modification not only enhances local search mechanism of SFLA but also improves the diversity. The performance of the O-SFLA is validated on Performance Optimization of the Paper Mill.

The present study is structured as follows: SFLA is discussed in Section 2 followed by the proposed scheme in Section 3. Section 4 briefs paper making system in paper mill. Experimental settings are presented in Section 5. Optimization results are given in Section 6 and finally the conclusions drawn and future work are given in Section 7.

II. Outline of SFL Algorithm

SFLA, proposed by Eusuff and Lansey in 2003, was initially designed to solve discrete optimization problems. SFLA like other nature inspired algorithms takes its inspiration from a group of frogs foraging for food. During this, frogs interact individual and exchange information. SFLA has proved that it can perform better as compared to genetic algorithm, ant colony optimization and the Particle swarm optimization algorithms [3]. SFLA partially follows particle swarm optimization in local search process and shuffled complex evolution algorithm in performing global search. SFLA as an optimizer model has also proved its competence in finding optimal solutions to many intricate optimization problems [4] – [11]. In SFLA, population of feasible solutions is represented by a set of frogs. This population is divided in equal number of groups named as memeplexes. In each memeplex the frogs are from different culture where they perform local search and the position of worst frog (based on

function value) is updated in order to optimize the position. After a fixed number of generations the shuffling of information, through evolution process, among the memeplexes takes place. This progression of local search and information exchange through shuffling is continued until the fixed termination criterion.
SFLA process is discussed in four steps as follows:

A. *Initialization Process in SFLA*

Equation (1) is used to initialize the random population (F) of frogs that is bounded by lower (lb) and upper bounds (ub):

$$x_{ij} = lb_j + rand(0,1) \times (ub_j - lb_j) \quad (1)$$

where $i = 1, 2, 3, 4,..., F$; $j = 1, 2,3, 4 ..., D$ (dimension) and *rand* (0,1) is a random number which is uniformly distributed between 0 and 1.

B. *Process of sorting and distribution of frogs in memeplexes*

Fitness value of each frog is calculated and sorted. Later based upon their fitness values, the frogs are distributed into memeplexes (m) of prefixed size (n). The distribution of frogs is performed in such way that the frog having best fitness value (optimal) belongs to first memeplex, accordingly rest of the frogs are distributed in to other memeplexes.

C. *Local search mechanism*

The frogs having best (X_b) , worst (X_w) and global (Xg) fitness function values are identified. Later the frog with worst fitness function value is modified. Then the frog having the worst fitness function value is considered to improve through evolutionary process in each cycle. Mathematically, the process of updating the worst frog position is done using the following equation:
Frog position update in i^{th} iteration is given by:

$$S_t = rand(0,1) \times (X_b - X_w) \quad (2)$$

The new position of Worst frog's will be:

$$X_w = current\ \ position(X_w + D_i) \quad (3)$$

$$-S_{\max} \le S_t \le S_{\max} \quad (4)$$

where t indicates the number of generations from 1 to N_{gen} (fixed number of evolution generation in any memeplex); Frog movement and the permissible movement (maximum) is represented by S_t and S_{max} respectively. If the position of the frog gets enhanced through the above discussed process then the worst frog position is replaced by the newly generated value. Otherwise, the process discussed in equation (2) and (3) are continued w.r.t the global best position (i.e. X_b is replaced by X_g). A new solution is initiated randomly using equation (1), in case of no significant change in worst frog position. This is a iterative evolution process that continues until specific or fixed number of generations (N_{gen}).

D. *Process of Shuffling*

In order to exchange the information, the frogs from different memeplexes are shuffled. The frogs from each subset are again shuffled and sorted to complete the round of evolution. This is an iterative process.

Searching process in SFLA is illustrated with the help of figure 1 and SFL Algorithm is presented in figure 2.

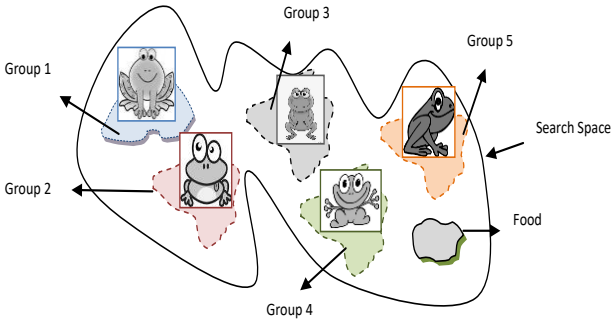

Figure 1. Illustration of searching process in SFLA locally and then globally.

Algorithm – Basic SFLA

Step 1: Initial population of frog size F is generated and evaluated (fitness value)

Step 2: Distribute the population of frog in m memeplexes where each memeplex contains n fog members such that $F = m$ x n.

Step 3: Start Evolutionary process

Step 4: Set $i = 1$

Step 5: while $i \le i_{max}$

Step 6: Identify the global best (x_g) position of frog in the population.

Step 7: Identify the worst (x_{worst}) and best (x_{best}) frog in each memeplex.

Step 8: Apply eq. (2) & (3) to generate new frog position and evaluate.

Step 9: If $f(x_{new}) < f(x_{worst})$

Step10: Set $x_{worst} = x_{new}$ and go to Step 13.

Step 11: $x_{best} = x_{global}$. Repeat Step 8 & Step 9.

Step 12: Random position is generated and replaced with x_{worst}.

Step 13: $i = i + 1$

Step 14: End While

Step 15: Population of frog is shuffled.

Step 16: If fixed termination criterion (N_{gen}) is achieved then exit else go to Step 2.

Figure 2. SFL Algorithm

III. Proposal: O – SFL Algorithm

SFLA has three phases, initialization of frog positions, local search process in each memeplex and information exchange i.e. shuffling process. SFLA performs both exploration as well as exploitation. However because of poor exploration capabilities SFLA sometimes get trapped in local optima that

results in poor convergence. In this study OBL is introduced in the structure of basic SFLA to eliminate its limitations.

OBL

This concept was introduced by Tizhoosh [12]. The opposite positions of frogs can be calculated using Equation (5):

$$x_{ijOpp} = \alpha_j - \beta_j - x_{ij} \quad (5)$$

where the upper and lower range of search process are represented by α and β , Frogs by i = 1, 2,..., F; and problem's dimension by j = 1, 2, ..., D.

Embedded OBL in O – SFLA

In the present study OBL has been employed in local search process of SFLA to improve positions of frogs within memeplex. In each memeplex opposite positions are generated using OBL (shown in figure 3) then elite n positions are selected based on fitness values ($f(n_{pop}) \cap f(n_{OBL\text{-}pop})$). Further, in order to enhance diversity and convergence one extra step to improve the performance of memeplex is included in the structure of basic SFLA (Step14) in figure 4. The proposed algorithm is discussed in figure 4.

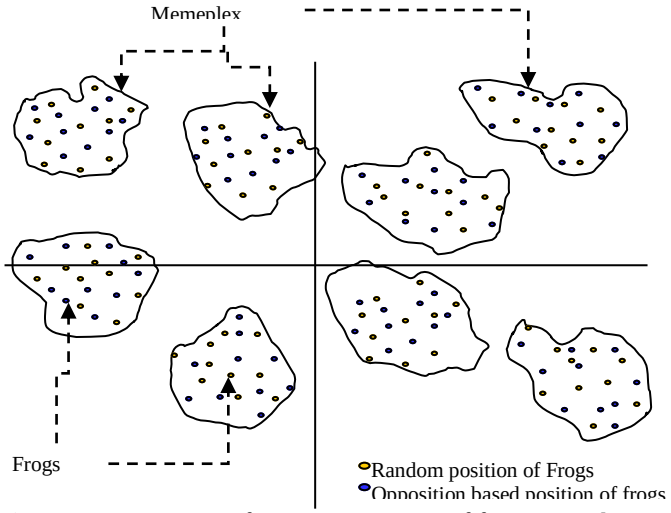

Figure 3. Generation of opposite position of frogs in each memeple.

Proposed Algorithm: O – SFLA

Step 1: Initial population of frog size N is generated and evaluated (fitness value)

Step 2: Distribute the population of frog in m memeplexes where each memeplex contains n_{pop} fog members such that $F = m \times n$.

Step 3: Start Evolutionary process

Step 4: Set $i = 1$

Step 5: while $i \leq i_{max}$

Step 6: Identify the global best (x_g) position of frog in the population.

Step 7: Apply OBL to generate opposition positions of frogs ($n_{OBL\text{-}pop}$) in each memeplex.

Step 8: Select the elite n frogs positions from a set of $n_{pop} \cup n_{OBL\text{-}pop}$ based on fitness values.

Step 9: Identify the worst (x_{worst}) and best (x_{best}) frog in each memeplex.

Step10: Apply eq. (2) & (3) to generate new frog position and evaluate.

Step11: **If** $f(x_{new}) < f(x_{worst})$

Step12: Set $x_{worst} = x_{new}$ and go to Step 16 .

Step13: $x_{best} = x_{global}$. Repeat Step 11 & Step 12.

Step14: Apply OBL to generate opposite point of x_{worst} i.e. $x_{OBL\text{-}worst.}$ If $f(x_{OBL\text{-}worst}) < f(x_{worst})$ go to Step 16.

Step15: Random position is generated and replaced with x_{worst}.

Step16: $i = i + 1$

Step17: End While

Step18: Population of frog is shuffled.

Step19: If fixed termination criterion (N_{gen}) is achieved then exit else go to Step 2.

Figure 4. O - SFL Algorithm

IV. A brief description of paper making system

A paper mill/industry is a complex engineering system that comprises of various functional units installed in series or in parallel. Some of the functional units are Chipping, Screening, bleaching, chemical recovery process, beating & refining, stock preparation, dryer, pressing, paper cutting etc (figure 5). Paper making process is one of the vital processes in any paper mill which is responsible for quality of the paper. The paper making process is demonstrated in figure 6 [13]. Initially wooden chips are screened before processing in chemical recovery cycle. Then the produced pulp is bleached using various sequences/stages to remove toxins. Then some chemical treatment is performed in order to improve brightness. During the process pulp passes through various stages like running of pulp over wire mat which are running on rollers then vacuum pumps are used to suck excessive water from the pulp. Later in press section, the wet pulp is processed through dryer (heated rollers) and press unit till it get converts into dry paper. The transition diagram is depicted in figure 6.

There are four basic subsystems in paper making process. In this study they are named as G_1, G_2, G_3 and G_4. A brief description of each subsystem has been discussed below [14]:

- G_1: In this subsystem wire mat units, aligned in series, are used to collect the suspended fiber that is deposited on the top of wire mat. Vacuum pumps are used to extract water from the pulp. This subsystem also helps in controlling the width of the paper sheet. Failure of this subsystem results in failure of a whole system.
- G_2: This subsystem comprises of synthetic belt that provides supports to the fiber running all the way through press and dryer processes. Failure of this subsystem results in failure of a whole system.
- G_3: In this subsystem, set of rollers are installed in series to support the synthetic belt and wire mat unit to spin on the pulp smoothly. Failure of even a single process results in failure of a whole system.
- G_4: This fourth subsystem comprises of 6 vacuum pumps installed in parallel. These vacuum pumps are used to suck water from the pulp all the way through wire mat. At a time four pumps remain functional, where as 2 are idle and in reserve. If more than two vacuum pumps get fail simultaneously, whole system gets fail.

Conjecture and taxonomy:

Following are the details of conjecture and taxonomy used in mounting mathematical (probabilistic) model for paper making process in paper mill.

Conjecture:

1) Repair/failure rates are statistically independent and constant over a time.
2) The performance of the repaired unit for a specific period is same as new unit.
3) There is no leading time for the repair.
4) All units either standby or active, posses the same capacity and nature.
5) Exponential distribution is followed by system repair/failure.
6) Replacement or repair (may be both) are included in service.
7) Efficiency or the capacity of the system may reduce.
8) No simultaneous failure among system whereas it is quite possible to have in subsystems in a system

Taxonomy

G1, G2. G3. G4 – are the four subsystems in good working conditions/states.

g1, g2, g3, g4 – presents bad or failed working conditions/states of four subsystems.

λ_{28-31} – represents the constant mean failure rates of four subsystems (G1 – G4)

$\lambda 32$ – represents steam supply failure rate

μ_{28-31} – represents constant mean repair rates of four subsystems (g1 – g4)

$\mu 32$ – presents steam supply repair rate

Q = represents the state probability

$Q_0(t)$ – presents the state probability that the system at a time t, is performing with its full capacity.

$Q_i(t)$ – represents the system is in *i* state at time t.

$Q'_i(t)$ – First order derivative of the state probabilities

To optimize the paper making performance, the mathematical model is referred from the []. The states 0, 4 and 8 presents that the system is processing with full capacity where as the states 1, 2, 3, 5, 6, 7, 9, 10, 11 and 12 & 0SF. 1SF, 2SF,…, 12SF presents the failed states.

$$Q'_0(t) + \sum_{i=28}^{32} \lambda_i Q_0(t) = \sum_{i=28}^{31} \mu_i Q_{i-27}(t) + \mu_{32} \sum_{j=0}^{3} Q_{jSF}(t) + \mu_{32} Q_{12SF(t)} \quad (6)$$

$$Q'_4(t) + \sum_{i=28}^{32} \lambda_i Q_4(t) + \mu_{31} Q_4(t) = \sum_{i=28}^{31} \mu_i Q_{i-23}(t) + \lambda_{31} Q_0(t) + \sum_{j=4}^{7} Q_{jSF}(t) \quad (7)$$

$$Q'_8(t) + \sum_{i=28}^{32} \lambda_i Q_8(t) + \mu_{31} Q_8(t) = \sum_{i=28}^{31} \mu_i Q_{i-19}(t) + \lambda_{31} Q_4(t) + \mu_{32} \sum_{j=8}^{11} Q_{jSF}(t) \quad (8)$$

$$Q'_i(t) + \lambda_{32} Q_i(t) + \mu_{28} Q_i(t) = \lambda_{28} Q_j(t) \quad (9)$$

where i = 1, 5, 9 and j = 0, 4, 8

$$Q'_i(t) + \lambda_{32} Q_i(t) + \mu_{29} Q_i(t) = \lambda_{29} Q_j(t) \quad (10)$$

where i = 2, 6, 10 and j = 0, 4, 8

$$Q'_i(t) + \lambda_{32} Q_i(t) + \mu_{30} Q_i(t) = \lambda_{30} Q_j(t) \quad (11)$$

where i = 3, 7, 11 and j = 0, 4, 8

$$Q'_{iSF}(t) + \mu_{32} \sum_{i=0}^{12} Q_{iSF}(t) = \sum_{i=0}^{12} \lambda_{32} Q_j(t) \quad (12)$$

where i & j = 0,1,2,…,12

Initial conditions i.e. at time t = 0, $Q_i(t)$ = 1 for i = 0 and $Q_i(t)$ = 0 for $i \neq 0$

The above equations are solved, considering steady state behavior. The steady state behavior in a plant/mill presents a failure free execution system usually for a long period. It can also be said that the performance of a system is independent of time. Therefore steady state availability of each unit in a system may be evaluated by substituting $\frac{d}{dt}=0$ and $t\to\infty$ in the equations (6) – (12) discussed above. When these equations (6 to 12) are solved recursively following are obtained:

$Q_1=K_1Q_0$; $Q_2=K_2Q_0$; $Q_3=K_3Q_0$; $Q_4= LQ_0$; $Q_5= K_1LQ_0$; $Q_6= K_2LQ_0$; $Q_7= K_3LQ_0$; $Q_8= NLQ_0$; $Q_9= K_1NLQ_0$; $Q_{10}= K_2NLQ_0$; $Q_{11}= K_3NLQ_0$; $Q_{12}= MNLQ_0$; $Q_{0SF}=BQ_0$; $Q_{1SF}=BK_1Q_0$; $Q_{2SF}=BK_2Q_0$; $Q_{3SF}=BK_3Q_0$; $Q_{4SF}= BLQ_0$; $Q_{5SF}= BK_1LQ_0$; $Q_{6SF}= BK_2LQ_0$; $Q_{7SF}=BK_3LQ_0$; $Q_{8SF}= BNLQ_0$; $Q_{9SF}= BK_1NLQ_0$; $Q_{10SF}= BK_2NLQ_0$; $Q_{11SF}= BK_3NLQ_0$; $Q_{12SF}= BMNLQ_0$

Where

$$K_1=\frac{\lambda_{28}}{\lambda_{32}+\mu_{28}};K_2=\frac{\lambda_{29}}{\lambda_{32}+\mu_{29}};K_1=\frac{\lambda_{30}}{\lambda_{32}+\mu_{30}}$$

$$L=\frac{\lambda_{31}}{\lambda_{31}+\mu_{31}-(N*\mu_{31})}$$

$$N=\frac{\lambda_{31}}{\{\lambda_{31}+\mu_{31}-(M*\mu_{31})\}};M=\frac{\lambda_{31}}{(\lambda_{32}+\mu_{31})};B=\frac{\lambda_{32}}{\mu_{32}}$$

Also, the sum of probabilities is 1 i.e. $\sum_{i=0}^{12}P_i+\sum_{j=0}^{12}P_{jSF}=1$, following expression is obtained:

$Q_0+K_1Q_0+K_2Q_0+K_3Q_0+LQ_0+K_1LQ_0+K_2LQ_0+K_3LQ_0+NLQ_0+K_1NLQ_0+K_2NLQ_0+K_3NLQ_0+MNLQ_0+BQ_0+BK_1Q_0+BK_2Q_0+BK_3Q_0+BLQ_0+BK_1LQ_0+BK_2LQ_0+BK_3LQ_0+BNLQ_0+BK_1NLQ_0+BK_2NLQ_0+BK_3NLQ_0+BMNLQ_0=1$

The above Kan further be normalized as:

$P_0[(1+K_1+K_2+K_3)+L(1+K_1+K_2+K_3)+NL(1+K_1+K_2+K_3+M)+B(1+K_1+K_2+K_3)+BL(1+K_1+K_2+K_3)+BNL(1+K_1+K_2+K_3+M)]=1$

$P_0(X_1+LX_1+NLX_2+BX_1+BLX_1+BNLX_2)=1$

$P_0[X1(1+L+B+BL)+X_2(NL+BNL)]=1$

$P_0(X_1Z_1+X_2Z_2)=1$

$P_0=1/(X_1Z_1+X_2Z_2)$ (13)

where $X_1=1+K_1+K_2+K_3$; $X_2=1+K_1+K_2+K_3+M$;

$Z_1=((1+L+B+(B*L))$; $Z_2=((N*L)+(B*N*L))$

Now the states 0, 4 and 8 presents the system (paper making) is full working or steady state availability (S_{AV}). The system performance is evaluated in terms of availability. The system can be represented as a sum of full working states as follows:

$$S_{AV}=Q_0+Q_4+Q_8$$

$$S_{AV}=Q_0+LQ_0+NLQ_0$$

$$S_{AV}=Q_0(1+L+NL)$$

$$S_{AV}=\left[\frac{(1+L+(NL)}{((X_1Z_1)+(X_2Z_2))}\right] \quad (14)$$

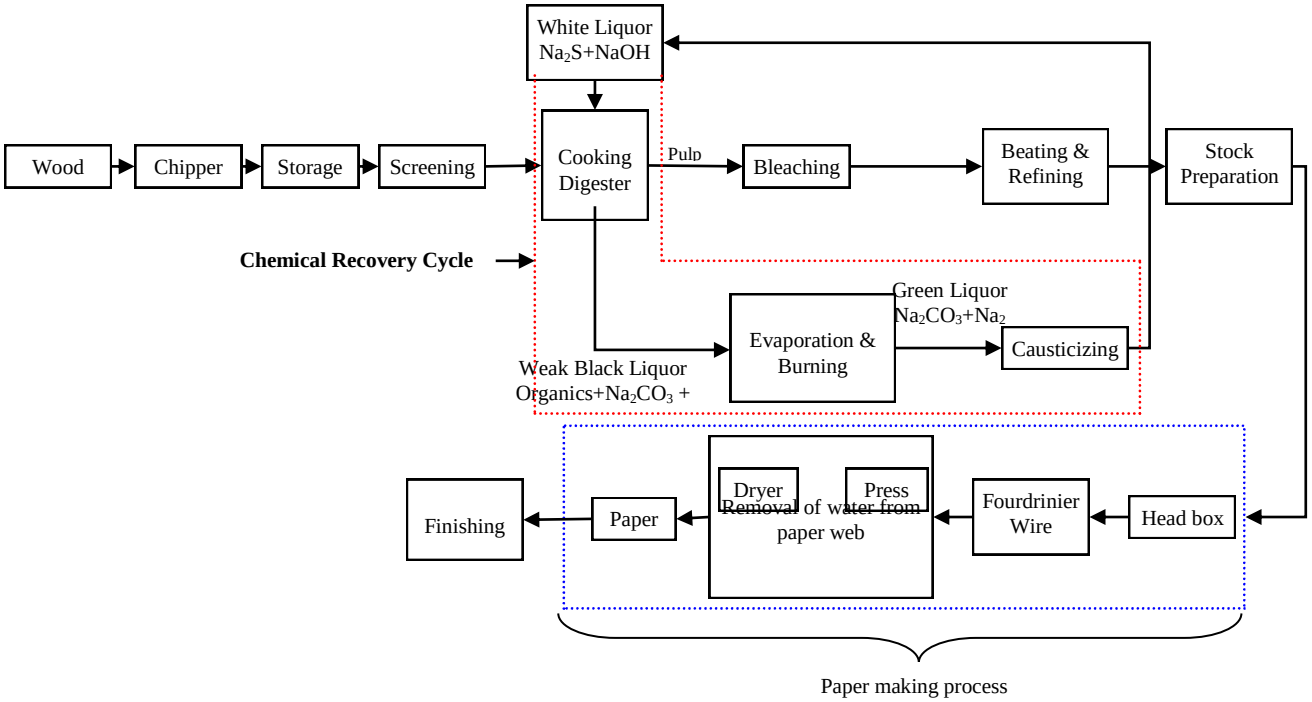

Figure 5. Graphical representation of paper making process

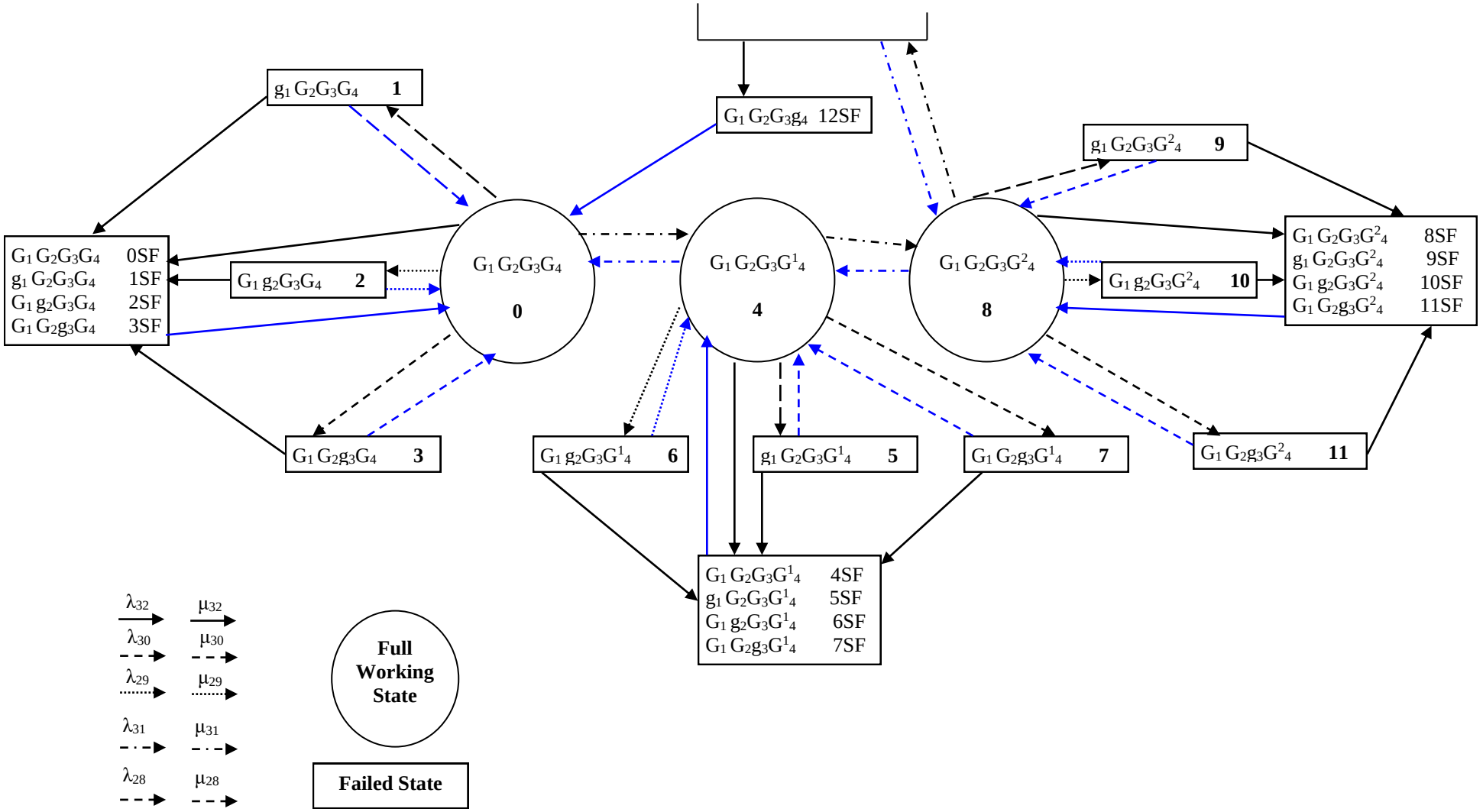

Figure 6. Transition diagram for Paper making process

V. Experimental Settings

The population of frog is fixed to 200 with 20 memeplexes. There will be 10 frogs in each memeplexes performing local search. 10 generations are performed in each memeplexes to explore the local region. The maximum movement step is fixed to 0.5 times of the search area. 25 runs are performed to evaluate mean function value and standard deviation. The simulations are performed on Dev C++ with following machine configurations: Intel(R) CPU T1350@1.86GHz with 1 GB of RAM.

VI. Results: Optimization of performance

In paper making system the optimization performance is highly influenced by both parameters in any subsystem i.e. repair (μ) and failure (λ). In this study O – SFLA is implemented to coordinate these parameters which help in stable state performance. In this case there are five repair and five failure parameters. The maximum and minimum limits of these parameters are discussed below.

Parametric restraints for repair (μ) and failure (λ) rate are presented below:

$0.001 \leq \lambda_{28} \leq 0.005$; $0.10 \leq \mu_{28} \leq 0.50$; $0.001 \leq \lambda_{29} \leq 0.005$; $0.10 \leq \mu_{29} \leq 0.50$; $0.003 \leq \lambda_{30} \leq 0.006$; $0.10 \leq \mu_{30} \leq 0.50$; $0.02 \leq \lambda_{31} \leq 0.10$; $0.10 \leq \mu_{31} \leq 0.50$; $0.02 \leq \lambda_{32} \leq 0.010$; $0.10 \leq \mu_{32} \leq 0.50$

Repair (μ) and failure (λ) parameters are optimized to desired availability level (S_{AV}). Maximum value of S_{AV} corresponds to best value of repair and failure parameters. The optimal simulated results using SFLA and O – SFLA for repair (μ) and failure (λ) parameters are presented in the Table 4. It can be noticed from the Table 1 that the system performance or the availability of the system is 92.59% using O – SFLA where as 91.85% is achieved using SFLA.

Table 1. Comparison of results.

Parameters	O – SFLA	SFLA
λ_{28}	0.0017	0.0048
λ_{29}	0.0024	0.1274
λ_{30}	0.0043	0.0049
λ_{31}	0.0727	0.1567
λ_{32}	0.0574	0.0023
μ_{28}	0.4039	0.2257
μ_{29}	0.1574	0.1001
μ_{30}	0.2347	0.1000
μ_{31}	0.1046	0.0804
μ_{32}	0.1553	0.3861
(S_{AV})	**0.9259**	**0.9185**

VII. Conclusions

In this study a novel variant of SFLA that embeds OBL is proposed and named as O – SFLA. The concept is embedded into the memeplexes before the frog initiates foraging and if the position of the worst frog does not improves then again OBL comes into picture to generate opposite point of worst frog position. The proposal is investigated on 6 benchmark functions and its performance is tested by comparing it with state-of-art algorithms. The efficiency of the proposal is also validated using non-parametric test analysis. The results prove the efficiency of the proposal. Further the proposed algorithm is implemented to optimize the performance of paper making system in paper mill which is a complex engineering system that comprises of various functional units installed in series or in parallel. The simulated result presents that the performance can be optimized up to 92.59% with the best values of repair and failure. In future the proposal will be implemented on a real world multi optimization problem.

Acknowledgment

Authors are thankful to the Editor-in Chief and the editorial board for their immense support. We also declare that there is no conflict of interest.

References

[1] M. M. Eusuff, K. E Lansey. "Optimization of water distribution network design using the shuffled frog leaping algorithm", *Journal of Water Resource Planning and Management*, 129(3), pp. 210-225, 2003.

[2] M. M. Eusuff, K. E Lansey. "Water distribution network design using the shuffled frog leaping algorithm". *In Proceedings of the World Water and Environmental Resources Congress 2001*, pp.1-8, 2001.

[3] M. M. Eusuff, K. E Lansey, F. Pasha. "Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization", *Engineering Optimization*, 38(2), pp. 129-154, 2006.

[4] E. Elbeltagi, T. Hegazy, D. Grierson. 2007. "A modified shuffled frog-leaping optimization algorithm: applications to project management", *Structure and Infrastructure Engineering: Maintenance, Management, Life-Cycl*, 3(1), pp. 53-60, 2007.

[5] E. Elbeltagi, T. Hegazy, D. Grierson. "Comparison among five evolutionary-based optimization algorithms", *Advanced Engineering Informatics*, 19(1), pp. 43-53, 2005.

[6] S. Sharma, T. K. Sharma, M. Pant, J. Rajpurohit, B. Naruka. "Accelerated Shuffled frog-leaping algorithm". *In Proceedings of the Soft Computing for Problem Solving* (*SocProS*), pp. 185-193, 2014.

[7] N. Bhagyashri, T. K. Sharma, M. Pant, S. Sharma, J. Rajpurohit. "Differential Shuffled Frog Leaping Algorithm". *In Proceedings of the Soft Computing for Problem Solving* (*SocProS*), pp. 249-257, 2014.

[8] S. Sharma, T. K. Sharma, M. Pant, J. Rajpurohit, B. Naruka. "Centroid Mutation Embedded Shuffled Frog-Leaping Algorithm". *In Proceedings of the Information and Communication Technologies (ICICT)*, pp. 127–134, 2014.

[9] Q. Duan, V. K. Gupta, S. Sorooshian. "A shuffled complex evolution approach for effective and efficient global minimization", *Optimization Theory and Application*, 76(3), pp. 501-521, 1993.

[10] T. K.Sharma, M. Pant. "Shuffled Artificial Bee Colony Algorithm", *Soft Computing*, (DOI: 10.1007/s00500-016-2166-2), 2016

[11] T. K.Sharma, M. Pant. "Identification of noise in multi noise plant using enhanced version of shuffled frog leaping algorithm", *International Journal of Systems*

Assurance Engineering and Management, Springer (DOI: 10.1007/s13198-016-0466-7), 2016.

[12] H.R. Tizhoosh. "Opposition-based learning: a new scheme for machine intelligence", *In Proceedings of the. International Conference on Computational. Intelligence. Modeling, Control and Automation*, pp. 695–701, 2005.

[13] T. K.Sharma, M. Pant, M. Singh."Nature-inspired metaheuristic techniques as powerful optimizers in the paper industry", *Materials and Manufacturing Processes*, 28(7), pp. 788-802, 2013.

[14] R. Khanduja, P.C. Tewari, R.S.Chauhan, D. Kumar. "Mathematical Modeling and Performance Optimization for the Paper Making System of a Paper Plant", *Jordan Journal of Mechanical and Industrial Engineering*, 4(4), pp. 487 – 494, 2010.

Author Biographies

Tarun K Sharma is presently affiliated with Amity University Rajasthan, Jaipur. He did his PhD from IIT Roorkee, India in Computational Intelligence. His research area includes design, analysis of hybrid swarm intelligence algorithms and their applications in versatile domain. He has published several papers in referred International Journals and conferences of repute. He is Program chair of International Conference on Soft Computing: Theories and Applications (SoCTA2017).