*Proceedings of the 2006 Winter Simulation Conference*
*L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.*

# A BEE COLONY OPTIMIZATION ALGORITHM TO JOB SHOP SCHEDULING

Chin Soon Chong

Singapore Institute of
Manufacturing Technology
71 Nanyang Drive
SINGAPORE 638075

Malcolm Yoke Hean Low

School of Computer Engineering
Nanyang Technological University
Nanyang Avenue
SINGAPORE 639798

Appa Iyer Sivakumar

School of Mechanical and Production Engineering
Nanyang Technological University
Nanyang Avenue
SINGAPORE 639798

Kheng Leng Gay

School of Electrical and Electronics Engineering
Nanyang Technological University
Nanyang Avenue
SINGAPORE 639798

## ABSTRACT

In the face of globalization and rapidly shrinking product life cycle, manufacturing companies are trying different means to improve productivity through management of machine utilization and product cycle-time. Job shop scheduling is an important task for manufacturing industry in terms of improving machine utilization and reducing cycle-time. However, job shop scheduling is inherently a NP-hard problem with no easy solution. This paper describes a novel approach that uses the honey bees foraging model to solve the job shop scheduling problem. Experimental results comparing the proposed honey bee colony approach with existing approaches such as ant colony and tabu search will be presented.

## 1 INTRODUCTION

Intense competition of global market has resulted in challenging manufacturing environment with lower product costs, shorter product life cycles and more product variety. The conflicting objectives of maintaining low inventory level to reduce costs, and quick response to customer demand to remain competitive calls for an effective scheduling algorithm for production shop floor. In this respect, there have been extensive studies of scheduling algorithms and heuristics in both static and dynamic job shops for decades by researchers and practitioners (Gere 1966, Blackstone et. al. 1982, Rajendran and Holthaus 1999, Jain and Meeran 1999)

A scheduling problem can be characterized by a set of jobs, each with one or more operations. The operations of a job are to be performed in a specified sequence on specific machines. The objective of scheduling is to determine the job schedules that minimize (or maximize) a measure (or multiple measures) of performance (Rajendran and Holthaus 1999). Due to factorial explosion of possible solutions, job shop scheduling problems are considered to be a member of a large class of intractable numerical problems known as NP-hard (Jain and Meeran 1999). The commonly used performance measures that are related to job shop scheduling include machine utilization, cycle time, throughput rate and inventory level. Of these measures, utilization of manufacturing resources is of vital importance to any manufacturing enterprise in the global competition of today. Improving resource utilization leads to better throughput rate and lower product cost. An alternative measure of resource utilization is makespan of a schedule, which is often studied by research community in job shop scheduling problems.

Solution techniques for shop scheduling problems range from simple priority dispatching rules such as FIFO (first in first out) and SPT (shortest processing time) to more elaborate techniques such as Branch and Bound (Brucker et. al. 1994), tabu search (Nowicki and Smutnicki 1996), shifting bottleneck algorithms (Balas and Vazacopoulos 1998), and ant colony (Blum and Sampels 2004). Meta-heuristics such as tabu search and shifting bottleneck procedure have been very successful. These approaches excel in solution quality as well as in computation time. Other meta-heuristics that work well when computation time is unconstrained are evolutionary computation approaches such as ant colony.

This work aims to explore an evolutionary computation approach, which is based on nectar collection in honey bee colonies, to job shop scheduling problems. This research is inspired by the work done by Nakrani and Tovey (2004), on using a new honey bee algorithm for dynamic allocation of internet servers. In their algorithm, servers and HTTP request queues in an Internet server colony are modeled as foraging bees and flower patches respectively. The experimental results show that the algorithm performs reasonably well in the dynamic allocation problem. Based on similar idea of honey bee colonies and the behavior of forager bees, which is characterized by decentralized and elementary interactions, we adapt the algorithm to job shop scheduling problem.

This paper first describes how honey bee colonies deploy forager bees to collect nectar amongst diverse flower patches in section 2. Subsequently in section 3, job shop scheduling problem is discussed. In section 4, the mapping of job shop scheduling meta-heuristic to honey bees forager deployment is given. Subsequently, the implementation details are discussed in section 5. This is then followed by a comparative study on the performance of the honey bee approach on benchmark problems in section 6. The paper finally ends with conclusions and future works in Section 7.

## 2 HONEY BEE COLONY

Colonies of social insects such as ants and bees have instinct ability known as swarm intelligence (Nakrani and Tovey 2004, Teodorovic and Dell'orco, 2005). This highly organized behavior enables the colonies of insects to solve problems beyond capability of individual members by functioning collectively and interacting primitively amongst members of the group. In a honey bee colony for example, this behavior allows honey bees to explore the environment in search of flower patches (food sources) and then indicate the food source to the other bees of the colony when they return to the hive. Such a colony is characterized by self-organization, adaptiveness and robustness.

Seeley (1995) proposed a behavioral model of self-organization for a colony of honey bees. In the model, foraging bees visiting flower patches return to the hive with nectar as well as a profitability rating of respective patches. The collected nectar provides feedback on the current status of nectar flow into the hive. The profitability rating is a function of nectar quality, nectar bounty and distance from the hive. The feedback sets a response threshold for an enlisting signal which is known as waggle dance, the length of which is dependent on both the response threshold and the profitability rating. The waggle dance is performed on the dance floor where individual foragers can observe. The foragers can randomly select a dance to observe and follow from which they can learn the location of the flower patch and leave the hive to forage. This self-organized model enables proportionate feedback on goodness of food sources.

## 3 JOB SHOP SCHEDULING

Job shop scheduling is concerned with finding a sequential allocation of competing resources that optimizes a particular objective function. The deterministic job shop scheduling problem is the most general of the classical scheduling problems (Jain and Meeran 1999). The problem consists of a finite set $J$ of $n$ jobs to be processed on a finite set $M$ of $m$ machines. Each job $J_i$ must be processed on every machine and consists of a chain of $m_i$ operations $O_{i1}$, $O_{i2}$,…,$O_{im}$, which have to be scheduled in a pre-determined given order. $O_{ij}$ is the $j^{th}$ operation of job $J_i$ which has to be processed on a machine $M_x$ for a processing time period of $\tau_{ij}$ without interruption and preemption. Each machine can process only one job and each job can be processed by only one machine at a time. The longest duration in which all operations of all jobs are completed is referred to as the makespan $C_{max}$.

More specifically, let $A_i$ be the set of ordered pairs of operations constrained by the precedence relations for each job $J_i$. For each machine $M_x$, the set $E_x$ describes the set of all pairs of operations to be performed on the machine. For each operation $O_{ij}$, let its earliest possible process start time be $t_{ij}$. Hence, the job shop scheduling problem can be modeled as:

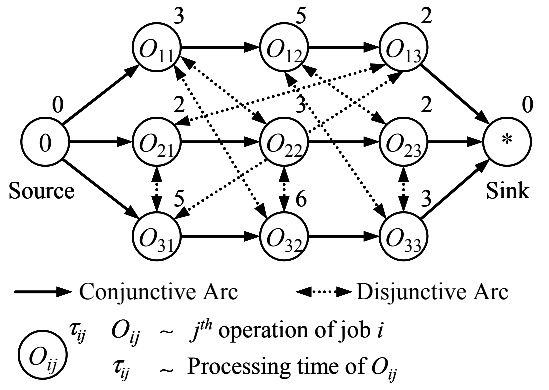
$$t_{i(j+1)} - t_{ij} \geq \tau_{ij} \qquad \forall (O_{ij}, O_{i(j+1)}) \in A_i,$$
$$t_{ij} - t_{kl} \geq \tau_{kl} \;\; or \;\; t_{kl} - t_{ij} \geq \tau_{ij} \qquad \forall (O_{ij}, O_{kl}) \in E_x,$$

Although many objective functions can be considered in job shop scheduling problems, makespan is the principal criterion for research community and is able to capture the fundamental computational difficulty which exists implicitly in determining an optimal schedule. Further, makespan minimization problem is well defined and is simple to understand. It is used in our work as a proving ground for the proposed honey bee colony algorithm for job shop scheduling problems.

A common representation for job shop scheduling problem is the disjunctive graph. In the graph, there is a node for each operation. There are also two additional nodes, known as source and sink to represent the initial and final operations respectively. The positive weight of each node is equivalent to the processing time of the corresponding operation. The starting and completion times of the source and sink represent the starting and finishing times of the job shop problem respectively. The source is connected to the initial operation of each job whereas the final operation of each job is connected to the sink.

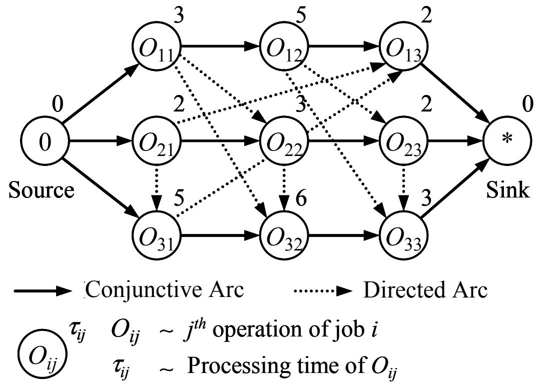
A set of directed conjunctive arcs is used to represent the precedence constraints for each job. A set of disjunctive arcs is used to represent capacity constraints to ensure that no two operations processed by the same machine can be executed simultaneously. An example of a 3 x 3 disjunctive graph is shown in Figure 1. The conjunctives arcs are shown by solid lines while the dotted lines represent disjunctive arcs. The set of operations on each machine is given by: $M_1 = \{O_{11}, O_{22}, O_{32}\}$, $M_2 = \{O_{12}, O_{23}, O_{33}\}$ and $M_3 = \{O_{13}, O_{21}, O_{31}\}$

Solutions to a job shop scheduling problem can be obtained by directing the disjunctive arcs of the disjunctive graph according to the machine permutations (i.e. by making bidirectional arcs to become unidirectional arcs). The makespan of a solution is the length of the longest directed path in the graph. The length of the path is given by the sum of the processing times of the operations on that path. A feasible solution to the earlier job shop scheduling problem is presented in Figure 2.



3
5
2
0
2
3
2
0
5
6
3
Source
Sink
Conjunctive Arc
Disjunctive Arc
$O_{ij}$ ~ $j^{th}$ operation of job $i$
$\tau_{ij}$ ~ Processing time of $O_{ij}$

Figure 1: A Disjunctive Graph Representation of a 3x3 Instance



3
5
2
0
2
3
2
0
5
6
3
Source
Sink
Conjunctive Arc
Directed Arc
$O_{ij}$ ~ $j^{th}$ operation of job $i$
$\tau_{ij}$ ~ Processing time of $O_{ij}$

Figure 2: A Feasible Solution

To better visualize, a Gantt chart for the solution is shown in Figure 3. The longest path is given by the set of operations $\{O_{23}, O_{33}, O_{31}, O_{32}\}$, which give a makespan of 16.

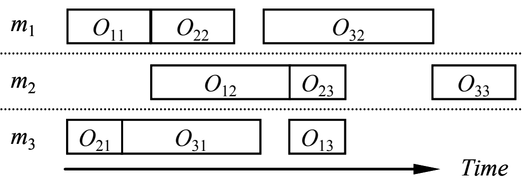


Figure 3: Gantt Chart of the Feasible Solution

## 4 HONEY BEE COLONY ALGORITHMS

This section details algorithms to perform job shop scheduling inspired by the behavior of honey bee colony. The challenge is to adapt the self-organization behavior of the colony for solving job shop scheduling problems. There are two major characteristics of the bee colony in searching for food sources: waggle dance and forage (or nectar exploration). We will discuss in separate sub-sections on how we map these characteristics of a bee colony to job shop scheduling.

### 4.1 Waggle Dance

A forager $f_i$ on return to the hive from nectar exploration will attempt with probability $p$ to perform waggle dance on the dance floor with duration D = $d_iA$, where $d_i$ changes with profitability rating while $A$ denotes waggle dance scaling factor. Further, it will also attempt with probability $r_i$ to observe and follow a randomly selected dance. The probability $r_i$ is dynamic and also changes with profitability rating. If a forager chooses to follow a selected dance, it will use the 'path' taken by the forager performing the dance to guide its direction for flower patches. We term the path as '**preferred path**'. The path for a forager is a series of landmarks from a source (hive) to a destination (nectar).

For job shop scheduling, the profitability rating should be related to the objective function, which in our case, is makespan. Let $Pf_i$ denote the profitability rating for a forager, it is given by:

$$Pf_i = \frac{1}{C^i_{\max}}$$

where,

$C^i_{\max}$ = makespan of the schedule generated by a forager $f_i$

The bee colony's average profitability rating, $Pf_{colony}$ is given by:

$$Pf_{colony} = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{C^j_{\max}}$$

where,

$n$ = number of waggle dance at time $t$ (we only consider those bees that dance when computing profitability rating)

$C^j_{\max}$ = makespan of the schedule generated by a forage $f_j$ performing waggle dance

The dance duration, $d_i$ is given by:

$$d_i = \frac{Pf_i}{Pf_{colony}}$$

The probability $r_i$ of following a path is adjusted according the profitability ratings of a forager and the colony based on the lookup table 1 (adopted from Nakrani and Tovey 2004). Essentially, a forager is more likely to randomly observe and follow a waggle dance on the dance floor if its profitability rating is low as compared to the colony's.

Table 1: Lookup Table for Adjusting Probability of Following a Waggle Dance

| **Profitability Rating** | $r_i$ |
|---|---|
| $Pf_i < 0.9Pf_{colony}$ | 0.60 |
| $0.9Pf_{colony} \le Pf_i < 0.95Pf_{colony}$ | 0.20 |
| $0.95Pf_{colony} \le Pf_i < 1.15Pf_{colony}$ | 0.02 |
| $1.15Pf_{colony} \le Pf_i$ | 0.00 |

## 4.2 Forage (Nectar Exploration)

For foraging algorithm, a population of l foragers is defined in the colony. These foragers cyclically construct solutions to the job shop scheduling problems. The foragers move along branches from one node to another node in the disjunctive graph and so construct paths representing solutions. A forager must visit every node once and only once in the graph, starting from initial node (i.e. source) and finishing at final node (i.e. sink), so as to construct a complete solution. When a forager is at a specific node, it can only move to next node that is defined in a list of presently allowed nodes, imposed by precedence constraints of operations. A forager chooses the next node from the list according to the state transition rule:

$$P_{ij}(t) = \frac{[\rho_{ij}(t)]^{\alpha} \cdot \left[\frac{1}{d_{ij}}\right]^{\beta}}{\sum_{j \in allowed\_nodes} [\rho_{ij}(t)]^{\alpha} \cdot \left[\frac{1}{d_{ij}}\right]^{\beta}}$$

where,

$\rho_{ij}$ = rating of the edge between node$_i$ and node$_j$

$d_{ij}$ = heuristic distance between node$_i$ and node$_j$

$P_{ij}$ = probability to branch from node$_i$ and node$_j$

The rating $\rho_{ij}$ of the edge (directed) between node$_i$ and node$_j$ is given by:

$$\rho_{ij} = \begin{cases} \alpha \\ \dfrac{1 - m\alpha}{k - m} \end{cases}$$

where,

$\alpha$ = value assigned to the preferred path, $\alpha < 1.0$

$k$ = number of allowed nodes

$m$ = number of preferred path, $m$ = 1 or 0

Based on the expression, it should be noted that for the first nectar exploration expedition by the foragers, $\rho_{ij}$ will be assigned the same value for all allowed nodes (since $m = 0$).

The parameters α and β tune the relative importance in probability of the 'weight' in edges found in the preferred path versus the heuristic distance. According to this rule, edges that are found in the preferred path and that are shorter will have a higher probability to be chosen for the solution. The heuristic distance is the processing time of the operation associated with node$_j$. When a forager completes a full path, the edges it has traveled and the makespan of the resulting solution will be kept for the waggle dance when it returns to the hive.

## 4.3 Algorithmic Framework

A combination of forage and waggle dance algorithms constitutes one cycle (or iteration) in this evolutionary computation approach. This computation will run for a specific number of iterations $N_{max}$. The best solution during the iteration process will be presented as final schedule at the end of run. The algorithmic framework of the scheduling algorithm is presented in Algorithm 1.

ALGORITHM 1: Algorithmic framework for job shop scheduling problem

```
for i = 1 to $N_{max}$
for j = 1 to l
   Forage
       Save best solution
       Waggle dance
end for
end for
```

## 5 IMPLEMENTATION DETAILS

The honey bee colony algorithms are developed using Java on Windows XP platform. A list of elite solutions is used to denote foragers that are currently performing waggle dance on the dance floor. The duration of a waggle dance is linked to the number of iterations that an elite solution is allowed to stay in the list. Each elite solution contains forager's path, its makespan, maximum number of iterations allowed and the iteration number ($i = 1$ to $N_{max}$) when a solution is added into the list. After every foraging cycle, the list is updated to remove elite solutions that have exceeded the maximum number of iterations.

In our implementation, the path is stored in a list. This list contains edges that connect two operations together. The edges are directional, connecting two consecutive operations. For the solution in Figure 3, the edges are $\{O, O_{11}\}$, $\{O_{11}, O_{21}\}$, $\{O_{21}, O_{31}\}$, $\{O_{31}, O_{22}\}$, $\{O_{22}, O_{12}\}$, $\{O_{12}, O_{32}\}$, $\{O_{32}, O_{23}\}$, $\{O_{23}, O_{13}\}$, $\{O_{13}, O_{33}\}$, and $\{O_{33}, *\}$. $O$ and * are source and sink respectively.

The forage algorithm is related to operation scheduling heuristic. There are two alternative approaches: operation centric or machine centric. In operation centric approach, a list of currently eligible operations that can be scheduled is always maintained during scheduling process. To be eligible, an operation's preceding operation (of a job) must have been scheduled. Each operation in the list is checked against the most recently scheduled operation (on the same machine) to identify if the 'edge' between the two operations (the most recently scheduled operation and the operation under consideration from the list) is found in the preferred path. Higher rating $\rho_{ij}$ will be assigned to the operation with edge found in the path. The operation scheduling starts with a list of the first operations of all jobs, and proceeds with operation selection algorithm outlined in section 4.2. When an operation is scheduled, it will be removed from the list and its succeeding operation (if any) will be added into the list.

For machine centric approach, a discrete-event simulation based technique is used to perform operation scheduling. An event list of events, which are in sorted order of increasing time, is maintained during scheduling process. At time $t = 0$, events relating to machine-ready status are inserted into the list. Events in the list are removed and executed one by one according to the event time. In case of tie for events having the same time, an event will be randomly picked. For the machine that is associated with the selected event, a list of currently eligible operations will be identified. Each operation in the list is checked against the most recently scheduled operation on the same machine to identify if the 'edge' between the two operations is found in the preferred path. Higher rating $\rho_{ij}$ will be assigned to the operation if the edge is found in the path. An operation is selected among the eligible ones according to algorithm described in section 4.2. When an operation $O_{ik}$ is scheduled on the machine, a new event of machine-ready status for the machine with time $t + \tau_{ik}$ will be inserted into the event list (if there is still pending operation(s) to be scheduled on the machine). The whole procedure repeats until the event list becomes empty.

## 6 EXPERIMENTAL EVALUATION

In this section we describe the benchmark problems, benchmark algorithms and present experimental results comparing the performance of honey bee algorithm with ant colony and tabu search algorithms.

### 6.1 Problem Instances

The performance of the honey bee colony scheduling approach is studied by evaluating them on the following 82 job shop problem instances (Ganesan et. al. 2004):

- 3 problems from Fisher & Thompson (1963), referred as ftp06, ftp10 and ftp20
- 40 problems from Lawrence (1984), referred as la01 – la40
- 20 problems due to Storer et al. (1992), referred as swv01 – swv20
- 10 problems used by Applegate and Cook (1991), referred as orb1 – orb10
- 4 problems used by Yamada and Nakano (1992), referred as yn1 – yn4
- 5 problems formulated by Adams et al. (1988), referred as abz5 – abz9

The sizes of these problems range from 6 to 50 jobs and 5 to 20 machines. Larger sizes of shop problems are not considered in the study as past results have concluded that the size of the shop does not affect the relative performance of dispatching rules, and valid conclusions could be drawn from experiment with relatively small shops (Nanot 1963, Buffa 1968).

### 6.2 Benchmark Algorithms

To compare and evaluate the performance of the proposed bee colony algorithm, we have included two other meta-heuristics in our experimental study. The first is an ant colony algorithm based on the work done by Dorigo et al. (1996). The second algorithm is a tabu search algorithm developed by Nowicki and Smutnicki (1996).

Ant algorithms are a class of meta-heuristic search algorithms that have been successfully applied to solving job shop scheduling problems. Ant algorithms are biologically inspired from the behavior of colonies of real ants on how they forage for food. Ants communicate with one another through indirect means by making modifications to the concentration of highly volatile chemicals called pheromones in their immediate environment. Since both ant algorithms and bee algorithms are based on self-organization of colonies to forage food sources, we reason it is orderly to compare the performance of both algorithms.

Tabu search is a meta-heuristic based on a local search technique which attempts to exploit the solution space beyond local optimality. It is a simple technique that attempts to guide a search process away from solutions that appear to duplicate or resemble previously achieved solutions. tabu search is an iterative approach improving on existing solutions and thus requires an initial solution to be constructed by other scheduling techniques. A good neighborhood structure plays an important role in tabu search, and it is one of the reasons why tabu search algorithm developed by Nowicki and Smutnicki (1996) is so successful. Their algorithm is one of the most efficient tabu search implementations for the performance criteria of makespan (Blazewicz et. al. 1996). By considering tabu search heuristic in our study, we can thus benchmark the proposed bee algorithm to one of the best in class.

## 6.3 Experiments and Results

Since both the bee and ant algorithms rely on random distribution function to construct solutions, a total of 5 replication runs have been performed for each job shop problem to obtain average results. Further, we have performed fine tuning on the parameters for the algorithms, and the final settings of major parameters are presented in Table 2.

Our initial experimentation indicates that the machine centric approach to operation scheduling always lead to better solutions comparing to operation centric approach. We therefore adopt machine centric approach in our experiments.

Table 3 summarizes the results on the relative performance of makespan in terms of percentage for bee colony, ant colony and tabu search algorithms. The results show the average, minimum and maximum percentage differences from the best known makespan for the 82 job shop problems. The second last row records the number of best solutions achieved among the three heuristics. The last row exhibits the relative execution time for the three heuristics.

From the results, it is obvious that tabu search outperforms other two heuristics. Tabu search records the closest results to the best known solutions and has the most number of best solutions. Further, it also manages to achieve best results in the shortest execution time. These spectacular results are attributed to the efficient critical block neighborhoods. Moreover, a tabu list that keeps track of the most recent tabu moves prevents the search algorithm to be locked in local minimums.

Table 2: Parameter Settings for the Algorithms

| Parameter | Bee colony | Ant colony | Tabu search |
|---|---|---|---|
| Maximum number of iterations, $N_{max}$ | 1000 | 1000 | 500 |
| Population size, $l$ | Number of jobs | Number of jobs | |
| Alpha, $\alpha$ | 1.0 | 1.0 | |
| Beta, $\beta$ | 1.0 | 1.0 | |
| Rating, $\rho_{ij}$ | 0.9 | | |
| Scaling factor, $A$ | 100 | | |
| Evaporation coefficient, $\rho$ | | 0.01 | |
| Maximum number of elite solution | 20 | | 20 |
| Maximum size of tabu list | | | 8 |

Table 3: Relative Performance of Bee Colony, Ant Colony and Tabu Search Heuristics

| Relative Improvement | Bee colony | Ant colony | Tabu search |
|---|---|---|---|
| Mean (%) | 11.08 | 11.45 | 5.17 |
| Minimum (%) | 0 | 0 | 0 |
| Maximum (%) | 38.09 | 38.70 | 24.23 |
| Number of best solutions | 14 | 13 | 27 |
| Relative execution time | 1.09x | 1.19x | 1x |

Comparing the performance of peers, bee colony and ant colony heuristics, bee algorithm performs slightly better than ant algorithm. Bee algorithm achieves better mean and maximum percentages as well as higher number of best solutions in comparison to ant algorithm. The time taken to solve the 82 job shop problems for both heuristics is approximately equal with bee colony being slightly faster. Evidently, both heuristics under perform tabu search primarily due to: 1) solutions are always constructed from scratch, instead of the more efficient operation swaps in tabu search; 2) no clear scheme to escape from being locked into local minimums.

## 7 CONCLUSIONS AND FUTURE WORKS

We have implemented a job shop scheduling algorithm based on self-organization of honey bee colony for solving job shop scheduling problems. It is found that the performance of the algorithm is comparable to ant colony algorithms, but gaps behind the efficient tabu search heuristics. Since the bee algorithm is our first implementation, we believe there is much room for improvement.

We intend to test the bee algorithms on semiconductor manufacturing scheduling problems. One of the works we intend to pursue is to deploy the algorithms in a distributed computing environment using software agents. Prior work has already been carried out using agents in symbiotic simulation of semiconductor assembly and test operation (Low et. al. 2005). In comparison to ant colony algorithms, it will be relatively easier to treat each bee forager as an agent in the bee colony algorithms since the issue of share state in maintaining a global pheromone table in ant algorithms does not occur.

Further works can obviously be done to enhance the bee algorithms. A research area to focus on is to incorporate local search heuristics based on an effective critical block neighborhood structure on every schedule generated by forage heuristic. Another area of research is to consider penalizing edges (lower rating $\rho_{ij}$) that are in opposite directions to the edges in the preferred path. As operation centric and machine centric scheduling methods have their respective strengths and weaknesses, it is proposed that both methods can be used jointly in the honey bee colony algorithm.

## REFERENCES

Abbas, H.A. and J. Teo. A true annealing approach to the marriage in honey–bees optimization algorithm. in The *Inaugural Workshop on Artificial Life (AL'01).* 2001. Adelaide, Australia.

Adams, J., Balas, E. and Zawack, D., "The shifting bottleneck procedure for job shop scheduling," *Management Science*, Vol. 34, No. 1 (1988), pp. 391-401.

Applegate, D. and Cook, W., "A computational study of the job shop scheduling problem," *ORSA Journal on Computing,* Vol. 3, No. 2 (1991), pp. 149-156.

Balas, E. and Vazacopoulos, A., "Guided local search with shifting bottleneck for job shop scheduling," *Management Science*, Vol. 44, No. 2 (1998), pp. 262-275.

Blazewicz, J., Domschke, W. and Pesch, E., "The job shop scheduling problem: conventional and new solution techniques," *European Journal of Operational Research*, Vol. 93, No. 1 (1996), pp. 1-33.

Blum, C. and M. Sampels, An ant colony optimization algorithm for shop scheduling problems. *Journal of Mathematical Modelling and Algorithms*, 2004. 3(3): p. 285 - 308.

Brucker, P., Jurisch, B. and Sievers, B., "A branch and bound algorithm for the job shop scheduling problem," *Discrete Applied Mathematics*, Vol 49, No. 1 (1994) pp. 109-127.

Buffa, E. S., *Operations Management: Problems and Models*, John Wiley & Sons (New York, 1968) pp. 454-460.

Chong, C.S., Using simulation based approach to improve on the mean cycle time performance of dispatching rules. in *Proceedings of the 2005 Winter Simulation Conference*. 2005.

Dorigo, M., Maniezzo, Vittorio, Colorni, Alberto, Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 1996. 26(1): p. 29-41.

Fisher, H. and Thompson G. L., "Probabilistic learning combination of local job shop scheduling rules," *Industrial Scheduling,* (1963), pp. 225-251.

Ganesan, V. K., Sivakumar A. I., and Srinivasan G, "Hierarchical minimization of completion time variance and makespan in jobshops," *Computers & Operations Research*, (In press: date of accept: 03-Aug-2004).

Gere, W. S., Jr., "Heuristics in jobshop scheduling," *Management Science,* Vol. 13, No. 1 (1966), pp. 167-175.

Jain. A. S. and Meeran. S., "Deterministic job shop scheduling: past, present and future," *European Journal of Operational Research*, Vol. 113, No. 2 (1999), pp. 390-434.

Lawrence, S. "Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (supplement)," Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburg, 1984.

Low, Y. H., Lye, K. W., Lendermann, P., Turner, S. J., Leo, S. and Chin, R., "An agent-based approach for managing symbiotic simulation of semiconductor assembly and test operations," in *Proceedings of the 2005 International Conference on Autonomous Agent and Multiagent Systems (AAMAS)*, July 25-29, 2005, Utrecht, The Netherlands.

Nakrani, S. and C. Tovey, On honey bees and dynamic allocation in an internet server colony. *Adaptive Behavior,* 2004. 12(3-4): p. 223-240.

Nanot, Y. R. "An experimental investigation and comparative evaluation of priority disciplines in jobshop like queuing networks," Management Science Research Project, UCLA, Research report, No. 87 (1963).

Nowicki, E. and Smutnicki, C., "A fast taboo search algorithm for the job shop problem," *Management Science,* Vol. 42, No. 6 (1996), pp. 797-813.

Rajendran, C. and Holthaus, O., "A comparative study of dispatching rules in dynamic flowshops and jobshops," *European Journal of Operational Research* Vol. 116, No. 1 (1999), pp. 156-170.

Seeley, T. D., *The Wisdom of the Hive*, Publication Harward University Press.

Storer, R. H., Wu, S. D. and Vaccari, R., "New search spaces for sequencing problems with application to job shop scheduling," *Management Science*, Vol. 38, No. 1 (1992), pp 1495-1509.

Sumpter, D.J.T. and S.C. Pratt, A modelling framework for understanding social insect foraging. *Behavioral Ecology and Sociobiology*, 2003. 53: p. 131-144.

Teodorovic, D and Dell'orco, M, Bee colony optimization - A cooperative learning approach to complex transportation problems, *Advanced OR and AI Methods in Transportation,* 2005, pp. 51-60

Yamada, T. and Nakano, R. A genetic algorithm applicable to large-scale job-shop problems. In Manner, R., Manderick, B. (Eds.), *Proceedings of the Second International Workshop on Parallel Problem Solving from Nature (PPSN'2)*, Brussels, Belgium, 1992. pp. 281-290

## AUTHOR BIOGRAPHIES

**CHIN SOON CHONG** obtained his degree in Electrical and Electronics Engineering from the City University of London, UK. He joined Singapore Institute of Manufacturing Technology (SIMTech), and is currently in the Planning Operation Management Group. He obtained his Master of Engineering in Computer Integrated Manufacturing from Nanyang Technological University, Singapore. He has been involved in simulation, scheduling and optimization related projects in logistic and manufacturing IT domains. The projects include cargo container operation simulation, printing process simulation, manufacturing cycle-time modeling, scheduling and optimization for MNCs. His current research interest includes simulation, planning, scheduling, optimization in the area of manufacturing, logistic, and supply chain. He can be reached via email at <cschong@simtech.a-star.edu.sg>.

**Dr. MALCOLM YOKE HEAN LOW** is an Assistant Professor in the School of Computer Engineering at the Nanyang Technological University (NTU), Singapore. Prior to this he was with the Singapore Institute of Manufacturing Technology, Singapore (SIMTech). He received his Bachelor and Master of Applied Science in Computer Engineering from NTU in 1997 and 1999 respectively. He was awarded a Gintic (now SIMTech) Postgraduate Scholarship in 1999. In 2002, he received his D.Phil. degree in Computer Science from Oxford University. His current research interest is in the application of parallel/distributed simulation, grid computing and agent technology for the modeling, simulation, analysis and optimization of complex systems. His e-mail and web address are <yhlow@ntu.edu.sg> and <www.ntu.edu.sg/home/yhlow/>.

**Professor ROBERT GAY** obtained his B. Eng, M. Eng and PhD degrees at the University of Sheffield, England, in 1965, 1967 and 1970 respectively. He was awarded the Grouped Scholarship in Engineering and Metallurgy by the University of Sheffield from 1967 to 1970. Since obtaining his PhD, he has been involved in Education and R&D working in institutions such as Singapore University (1972-1979), Rutherford and Appleton Laboratory (England, 1979-1982), NTU (1982-1995 and 1999-present) and Singapore Institute of Manufacturing Technology (1989-1999). He has also been actively involved in promoting innovation in Singapore through work in various committees: Science Quiz (MOE), Science Centre Board, National CAD/CAM (NCB), Tan Kah Kee Young Inventors Award (TKK Foundation & NSTB), National IT Plan (NCB), Technopreneurship incubation center (ITE), Commercenet Singapore, Singapore Computer Society. Currently Professor Gay is in the School of EEE at NTU and also Director and CEO of the ASP Centre. He has more than a hundred publications in journals, conference proceedings and books. His email and web addresses are <eklgay@ntu.edu.sg> and <www.ntu.edu.sg/eee/icis/cv/robertgay.html>.

**Associate Professor APPA IYER SIVAKUMAR** (Senior member IIE) is an Associate Professor in the School of Mechanical and Production Engineering at Nanyang Technological University (NTU), Singapore and a Fellow of Singapore Massachusetts Institute of Technology (MIT) Alliance (SMA). Prior to this he was at Singapore Institute of Manufacturing Technology, Singapore (SIMTech). His research interests are in the area of simulation-based optimization of manufacturing performance, supply chain, and dynamic schedule optimization. Prior to joining SIMTech in 1993, he held various management positions including technical manager and project manager for nine years at Lucas Systems and Engineering and Lucas Automotive, UK. He received a Bachelors of Engineering from University of Bradford, UK and a PhD in Manufacturing Systems Engineering from University of Bradford, UK. He has been the technical chair and co-edited the proceedings of the 3rd and 4th International Conference on Computer Integrated Manufacturing (ICCIM '95 and ICCIM'97), Singapore. His email and web addresses are msiva@ntu.edu.sg and <www.ntu.edu.sg/home/MSiva/>.