

# [Big valley landscape exploitation biology essay](https://assignbuster.com/big-valley-landscape-exploitation-biology-essay/)

[](https://assignbuster.com/)[Science](https://assignbuster.com/essay-subjects/science/), [Biology](https://assignbuster.com/essay-subjects/science/biology/)

Li-Pei Wong Chin Soon ChongChi Yung PuanMalcolm Yoke Hean LowSchool of Computer Engineering Singapore Institute of Manufacturing TechnologyNanyang Technological University 71 Nanyang DriveNanyang Avenue, SINGAPORE 639798 SINGAPORE 638075

## ABSTRACT

Scheduling is a crucial activity in semiconductor manufacturingindustry. Effective scheduling in its operations leadsto improvement in the efficiency and utilization of itsequipment. Job Shop Scheduling is an NP-hard problemwhich is closely related to some of the scheduling activitiesin this industry. This paper presents an improved Bee ColonyOptimization algorithm with Big Valley landscape exploitationas a biologically inspired approach to solve theJob Shop Scheduling problem. Experimental results comparingour proposed algorithm with Shifting BottleneckHeuristic, Tabu Search Algorithm and Bee Colony Algorithmwith Neighborhood Search on Taillard JSSP benchmarkshow that it is comparable to these approaches.

## 1 INTRODUCTION

Semiconductor manufacturing industry is a complex yetdynamic business. Some major activities in the semiconductorproduction are wafer fabrication, wafer probe, product assembly and final testing. These activities arehighly capital intensive and need to be performed in an unpredictableenvironment, as the activities are sensitive todisruption factors such as frequent facilities maintenance, rework, machine downtime etc. To compete in a versatileenvironment where the product life cycle is considerablyshort, semiconductor manufacturers are trying differentmethods to improve productivity and minimize the cycletime of their products. Solutions to such problems play animportant role in ensuring that scarce resources are allocatedeffectively to competing activities, so as to maximizetheir utilization and efficiency. Job Shop Scheduling Problem (JSSP) is closely relatedto activities in semiconductor manufacturing industry suchas part routing, part processing operations and coordinationof part handling as discussed by Gupta and Sivakumar(2006), and Cavalieri et al. (1999). It is NP-hard in nature(Lenstra et al. 1977). In a typical JSSP, a sequential job allocationon resources (machines) that optimizes a particularobjective function is to be determined. While many algorithmsexist to solve the JSSP (Blazewicz et al. 1996, Lee et al. 1997), the Bee Colony Optimization (BCO) algorithmhas recently been adapted (Chong et al. 2006, Chonget al. 2007). The bee inspired algorithms are generalizedfrom the foraging behaviors of bees where waggle dance isused as a communication medium to attract other bees to afood source. When the behaviors are applied algorithmicallyon complex and dynamic problems, the algorithm appearsto be self-organized, flexible and robust in discoveringsolutions to the problems (Bonabeau and Meyer 2001). The bee inspired algorithms have also been attemptedin various areas including the dynamic server allocation inInternet hosting center (Nakrani and Tovey 2004), hexgame playing program (Rijswijck 2007), Traveling SalesmanProblem (Lucic and Teodorovic 2002, Lucic andTeodorovic 2003, Wong et al. 2008), and TelecommunicationNetwork Routing (Wedde et al. 2004). A survey thatdiscusses bee inspired algorithms and their applications tosome generalized assignment problems can be found inBaykosoglu et al. (2007). In this paper, a BCO algorithm with Big Valley landscape(BCBV) is presented. Besides the foraging behaviors, bees in the proposed algorithm are equipped with theability to explore the search space which appears in a BigValley structure as discussed in the works by Reeves(1999), Nowicki and Smutnicki (2005), and Boese et al.(2008). An effective search around the Big Valley structurewill help in locating the best solution in the space. TheBCBV algorithm is tested on Taillard JSSP benchmark andcompared against the Shifting Bottleneck Heuristic (SBP), Tabu Search Algorithm (TSA), and Bee Colony Algorithmwith Neighborhood Search (BCNS) (see Sections 5. 1 and5. 2 for details). This paper starts with a discussion on JSSP in Section2. Section 3 explains the Big Valley landscape structure. Adiscussion on the BCBV algorithm is presented in Section978-1-4244-2708-6/08/$25. 00 ©2008 IEEE 2050Proceedings of the 2008 Winter Simulation ConferenceS. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds. Wong, Puan, Low, and Chong4. Experiments and results are presented in Section 5. Finally, this paper ends with a conclusion.

## 2 JOB SHOP SCHEDULING PROBLEM (JSSP)

As presented by Adams et al. (1988), JSSP is defined by aset J of n jobs, J = {1, 2, …, n}. These jobs are to be processedon a set of m machines, M = {1, 2, …, m}. O is a setof operations, O = {0, O11, …, O1m, On1, …, Onm, z} whereOij denotes the j-th operation of job Ji. 0 and z denotes twofabricated operations which represent the " first" and " ultimate" operation. Thus, | O| = (n\*m)+2. Each operation Oij isassociated with tij and \_ij which denote its earliest start timeand processing time respectively. Apart from these, the followingconstraints have to be fulfilled: \_ Each job Ji in set J is composed of a set Ai whichconsists of ordered pairs of operations, constrainedby the precedence relations in (1). i j ij ij ij i j i t \_ t \_ \_ O O \_ A \_ \_ , ( , ) ( 1) ( 1) \_ (1)\_ Each machine Mr in set M is composed of a set Erwhich describes the set of all pairs of operationsto be performed on machine r. Each operation Oijwill be processed for \_ij without interruption andeach machine can handle at most one operation ata time. These constraints are shown in (2). O O E r Mt tt tij kl rkl ij ijij kl kl

## \_ \_ \_ \_

## \_

## \_

## \_ \_

## \_ \_

## ( , ) ,

## \_

## \_

(2)\_ For every operation in O, tij must be greater thanor equal to 0. This constraint guarantees the completionof all jobs as shown in (3). t O O ij ij \_ 0, \_ \_ (3)Although there are many metrics that can be consideredas the objective function of JSSP, makespan (Cmax) isthe most common and will be the focus in this paper. Cmaxis defined as the longest duration for which all operationsof all jobs are completed.

## 2. 1 Disjunctive Graph Representation, Critical

## Path and Its Block Decomposition

A common representation for JSSP is the disjunctivegraph. A disjunctive graph is a collection of nodes (vertices)and edges (arcs). Nodes are linked by the edges of thegraph. Hence, a disjunctive graph G is defined as\_ \_ Conj Disj G \_ O, E \_ E (4)where O is the set of operations as defined in Section 2. EConj is a set of directed conjunctive edges representing theprecedence constraints of each job as described in (1). EDisjis a set of bi-directional disjunctive edges representing thecapacity constraints of each machine as described in (2). These disjunctive edges link all the operations that need tobe handled by a particular machine. An example of the disjunctivegraph based on a 3-job x 3-machine JSSP in Table1 is shown in Figure 1. Each row of the table represents apre-defined machine precedence order for each job withthe processing time in parentheses. A solution can be obtainedby converting bi-directional disjunctive edges to becomedirected edges. To make sure that a feasible solutionis obtained, the conversion is performed such that no cycleexists in the graph. This will eventually turn the disjunctivegraph to a directed graph as shown in Figure 2. Table 1. An Example of 3-job x 3-machine JSSP. Job Machine (Processing time)1 2 (3) 1 (13) 3 (6)2 1 (8) 2 (4) 3 (12)3 3 (10) 2 (5) 1 (5)Legend: DirectedConjunctive EdgeBi-directionalDisjunctive Edge\_ij - Processing time of OijOij Oij - j-th operation of job i\_ijO1213O218O224O2312O3110O32500z0O113O335O136Figure 1: A Disjunctive Graph for 3-job x 3-machine JSSPInstance in Table 1. Legend: DirectedConjunctive EdgeDirectedDisjunctive Edge\_ij - Processing time of OijOij Oij - j-th operation of job i\_ijO1213O218O224O2312O3110O32500z0O113O335O136Figure 2: A Directed Graph (Feasible Solution) for 3-job x3-machine JSSP Instance in Table 1. M1M2M3Oij = j -th operation of job iO31 O23 O13O21 O33 O12O32 O22 O11timeFigure 3: Gantt Chart for the Directed Graph in Figure 2. 2051Wong, Puan, Low, and ChongThe longest path (or critical path) in the directed graphprovides the makespan for the JSSP. The critical path inFigure 2 is given by: 0 \_ O31 \_ O32 \_ O22 \_ O11 \_ O12\_ O13 \_ z. The sum of all the processing times is 0 + 10 +5 + 4 + 3 + 13 + 6 + 0 = 41. To better illustrate the criticalpath found in Figure 2, a Gantt chart is shown in Figure 3where the operations in the critical path are shaded. A critical path consists of a set of operations whichcannot be delayed so that all the jobs will be completed ontime. The critical path can be decomposed into a set of rblocks, B = {b1, b2, … br} as described by Grabowski et al.(1986), and Nowicki and Smutnicki (1996). Each blockcontains an order set of operations/nodes which are processedon the same machine bk = {dk1, dk2, …, dki} whered\_O. At the same time, two consecutive blocks must containoperations/nodes which are processed on different machines. The following example illustrates the block decompositionof the critical path found in Figure 3. As stated earlier, one of the critical paths for the 3-jobx 3-machine JSSP instance in Table 1 is given by {O31, O32, O22 , O11 , O12 , O13 }. By performing block decomposition, four blocks are identified where r = 4, B = {b1, b2 , b3, b4}. The content of each block is as follows: b1 ={O31,}, b2 = {O32, O22, O11}, b3 = {O12}, b4 = {O13}. Thisblock decomposition is the central idea in the implementationof the neighborhood operator which will be discussedin Section 4. 3. In the subsequent section, the Big Valley landscapestructure will be discussed. The discussion will includehow the landscape looks and its characteristics.

## 3 THE BIG VALLEY LANDSCAPE STRUCTURE

A landscape can be described as a structure of the neighborhoodgenerated by a heuristic operator used to traversethe search space of the problem in view of the objectivefunction. Reeves (1999) suggested that when a heuristicsearch approach is applied to a combinatorial optimizationproblem that defines a unique search space, a " landscape" will be created. In addition, it is found that different landscapeswill be created by different search operators used inthe search space exploration. A landscape consists of manylocal optima or false peaks which will be changing withrespect to the heuristic search operators. The existence ofthese local optima or false peaks often obstruct the searchfrom locating global optimum as illustrated in Figure 4. However, these landscape structures can assist thesearch of global optimum instead of obstructing it. Onesuch landscape structure is the " Big Valley" structure observedby Boese et al. (2008) in the 2-opt operator forTraveling Salesman Problem (TSP). Similar landscapestructure has since been extended to flow shop schedulingproblem (FSP) by Reeves (1999), and also in JSSP byNowicki and Smutnicki (2005). In the Big Valley landscape structure, local optimatend to exist close to one another in clusters, with eachcluster centered on the global optimum forming a valleystructure as illustrated in Figure 4. The formation of theBig Valley landscape has strong implication for how heuristicsearch should be performed. The Big Valley structuresuggests that the determination of new start points forsearch should be based on previous local optimum ratherthan based on a random point in the search space. This isbecause good candidate solutions are often found to beclose to other good solutions. By exploring and exploitingthe areas near to these local optima effectively, the searchwill be directed towards the global optimum eventually. global minimumclusters of local minimumobjectivefunction ridge of localoptimashoulder of localoptimastatecurrent spacestateinitialstateFigure 4: A Landscape with the Big Valley Feature.

## 3. 1 Big Valley Landscape Analysis

To investigate the Big Valley landscape structure, a plotfrom a sample of 1500 solutions collected through a run ofBCBV algorithm on the ta\_01 (15-job x 15-machine) problemis generated as shown in Figure 5. ta\_01 is one of thedatasets in Taillard JSSP benchmark which will be discussedin Section 5. 1. The solutions are plotted using thepercentage makespan difference, \_ (see Section 5. 1) versustheir heuristic distance (see Section 4. 4. 1) from a fixed localoptimum. In Figure 5, solutions are concentrated in four separateclusters at different distances from a fixed local optimum. The clusters are formed by the neighborhood operator (seeSection 4. 3. 2) which is integrated in BCBV. Figure 5 alsoshows that the clusters are some heuristic distance awayfrom one another. Search should be performed intenselywithin clusters to find new local optima with bettermakespan since they are most probable to occur close toone another. The clustering approach by neighborhood operatorand intensive searching around different clusterswhich are certain heuristic distance apart is implemented inBCBV algorithm. Details about the analysis on the BigValley landscape can be obtained in Reeves (1999), andNowicki and Smutnicki (1996). 2052Wong, Puan, Low, and ChongFigure 5: \_ versus Distance from a Local Optimum.

## 4 BCO WITH BIG VALLEY LANDSCAPE

This section first describes the natural foraging model of atypical bee colony. Next, an algorithmic framework ofBCBV is presented. It is followed by an overview of theforaging model used in constructing feasible solutions forJSSP, and the waggle dance model used in finding goodsolutions. Finally, a discussion on the Big Valley landscapeexploitation in BCO is provided.

## 4. 1 Bee Colony

The foraging behavior in a bee colony remains mysteriousfor many years until von Frisch (1974) translated the languageembedded in bee waggle dances. Waggle dance operatesas a communication tool among bees. Through thewaggle dance, bees could describe the distance, direction, and description of the food source to other bees. Distanceis conveyed by the type and duration of the waggle dance. Suppose a bee found a rich food source, a figure-eightpattern is shown in the dance. This figure-eight dance consistsof a straight waggle run followed by a turn to the rightback to the starting point, and then another straight wagglerun followed by a turn to the left and back to the startingpoint again. Via these informative dances, the bee has actuallyinformed its hive mates about the direction and distanceof the food source. von Frisch (1974) also suggestedthat bees could describe the type of flower which is richestto forage through the use of pollen. Once a bee has associateditself with the particular scent of pollen from a waggledance, it will ignore flowers with other scents at the indicatedlocation. Such a mechanism achieves a kind of shorttermmemory to differentiate and identify food sources andwill be explored through the addition of a Taboo list in theBCO model in this paper. Further discussions about waggledance can be found in Dyer (2002), and Biesmeijer andSeeley (2005).

## 4. 2 BCBV Algorithm

The BCBV algorithm uses a similar model based on bees’foraging behaviors as shown in Figure 6. The BCBV startswith a set of feasible schedules generated by dispatchingrules, which will be discussed in Section 4. 3. 1. A combinationof foraging and performing waggle dance constitutesone cycle (or iteration). The foraging model will be discussedin Sections 4. 3. 2 and 4. 3. 3. The waggle dancemodel, which includes how a bee observes, selects and performsa waggle dance, is implemented using a linked listWL and will be discussed in Section 4. 4. The BCBV algorithm is executed for Nmax iterationsand the best solution found during the searching processwill be presented as the final schedule at the end of a run. procedure BCBVCmax\_best \_ \_Niter \_ 0GenerateInitialSolution( ) [see section 4. 3. 1]while Niter <> Nmax dofor each forager bee fi doif WL <>{ }fi. ObserveNSelectDance( ) [see section 4. 4. 2 & 4. 4. 3]

## end if

fi. Cmax\_ fi. Forage( ) [see section 4. 3. 2 & 4. 3. 3]if fi. Cmax < Cmax\_bestCmax\_best \_ fi. Cmaxfi. PerformWaggleDance( ) [see section 4. 4. 1]

## end if

## end for

Niter\_ Niter+1

## end while

end procedure BCBVFigure 6: Algorithmic Framework of BCBV.

## 4. 3 Foraging Model with Neighborhood Search

The foraging model starts with a group of bees constructinga set of feasible solutions by using a set of dispatchingrules for JSSP with the use of disjunctive graph representationas mentioned in Section 2.

## 4. 3. 1 Generate Initial Feasible Solution

The dispatching rules that are applied in the BCBV algorithminclude Shortest Processing Time, Longest ProcessingTime, Most Work Remaining, Least Work Remaining, Work in Next Queue, Last In First Out, First In First out, Shortest Processing Time+Work in Next Queue, ShortestProcessing Time+Most Work Remaining and RandomDispatching Rules. In the Random Dispatching Rules, beesare allowed to randomly pick one of the listed rules to constructa feasible solution. These rules are applied in a roundrobin fashion for all the bees. Implementation details of thelisted rules can be found in the work by Yamada (2003). 6810121416180 5 10 15 20 25 30 35 40 45Distance From Local Optimum

## \_ (%)

2053Wong, Puan, Low, and ChongThe set of feasible solutions generated by the dispatchingrules will be used to generate other feasible solutions. Other than this, bees may also generate new solution fromtheir own solution found in previous iteration or based onthe dance (solution) that they followed. These three sets offeasible solutions will serve as a foundation to produceother solutions for the rest of the algorithm via a neighborhoodoperator. The mechanism of the operator will be explainedin the subsequent section.

## 4. 3. 2 Neighborhood Operator

Each feasible solution, Sx, has it own critical path whichcan be identified via the disjunctive graph. To produce aset of moves based on Sx, a neighborhood operator C(Sx), which is based on the block structure described in Section2, is defined in (5): C S C b b B jrx j j j \_ \_ \_ \_ ( ) ( ), 1 \_ (5)where Cj is defined as in (6), (7), and (8):

## \_\_ \_\_

## \_ \_ \_

## \_

## \_ \_

## \_ \_

otherwised d b and BC b i i

## ,

, , 1 1( ) 1( 1) 1 11 1(6)

## \_\_ \_\_ \_\_

## \_\_ \_\_

## \_

## \_

## \_

## \_

## \_ \_

## \_ \_

## \_

## \_

otherwised d b and B jd d d d b and B jC b j j jj j j i ji jj j

## ,

, , 2, , , , 3( ) 1 21 2 ( 1)for j= 2, …, r-1 (7)

## \_\_ \_\_

## \_ \_ \_

## \_

## \_ \_

## \_

otherwised d b and BC b r r rr r ,, , 1 1( ) 1 2 (8)The generation of C(Sx) is adapted from the work byNowicki and Smutnicki (1996). (6) suggests that swappingis allowed on the last two operations in the first block. (7)suggests that swapping is allowed on the first two (and lasttwo) in every intermediate blocks (b2, …, br-1). (8) suggeststhat the swapping is allowed on the first two operations inthe last block. Once the neighborhood operation ends, C(Sx) contains a set of moves that can be used to generatenew feasible solutions. Of all the moves in C(Sx), movesthat are able to generate better Cmax when compared to thecurrent Cmax (improving moves) are placed into the set MI , and moves that have been recently visited (Taboo moves)are placed in the set MT. A bee will then decide whichmove to select based on the strategies that will be discussedin Section 4. 3. 3.

## 4. 3. 3 Move Selection Strategies

Recall that bees could remember scent and avoid searchingpatches of flowers with different scents through the pollenidentification ability (refer to Section 4. 1). To aid thesebees in making a better decision in selecting a move, asimilar short-term memory is given to the artificial bees inthe model. This short-term memory is developed via theuse of a Taboo list. Each time a move is selected, its inversemove will be added to the Taboo list. Should the inversemove be encountered in the next search, its selectionis avoided if possible. The aim of the Taboo list is to directthe search process away from recently visited solutions sothat more of the unexplored search space can be reached. Hence, more solutions can be evaluated. Besides Taboolist, another strategy is introduced to aid the bees to select abetter move. Using this strategy, a bee will select at randoma move from MI should one exist, else it will select atrandom a non-improving move. The combination of the neighborhood operator, Taboolist and move selection strategy form the foraging modelshown in Figure 7. It corresponds to the Forage() methodin Figure 6. Step 1: Find a set of moves from a feasible solution, Sx by usingoperator C (Sx). Step 2: Identify MI and MT from C (Sx). Step 3: If \_ \_ \_ I T M MRandomly pick a move from MI – MT. Go to Step 7. Step 4: If \_ \_ \_ I T M MRandomly pick a move from MI \_ MT. Go to Step 7. Step 5: If MT \_ \_Randomly pick a move from MT . Go to Step 7. Step 6: Random pick a move from C(Sx). Step 7: Add the inverse of selected move to MTStep 8: Return the selected move. Figure 7: Algorithm for Forage().

## 4. 4 Waggle Dance with Exploitation of Big Valley

## Landscape

Upon returning to the hive after foraging, a bee will needto perform waggle dance in order to convey informationabout its discovery to other hive mates. At any one time, abee will perform waggle dance if its Cmax is smaller thanthe current best Cmax among all the bees. Note that in ourimplementation, the solutions representing the dances bythe bees are accumulated in an unbounded list, WL. Accumulationof these dances (solutions) over time will create alandscape which consists of multiple local optima as discussedin Section 3. To manage the landscape effectively, approaches stated in Sections 4. 4. 1 to 4. 4. 3 are proposed.

## 4. 4. 1 Dance Accumulation Strategy

Since WL is unbounded, it might accumulate too manydances such that a unique landscape could not be identified. To maintain a finite set of unique peaks (local optima), a replacement approach is introduced. This approachcompares the similarity of the newly generated solution2054Wong, Puan, Low, and Chong(denoted as Sp) with each solution in WL (denoted by Sqwhere Sq\_WL) using a distance metric D. The role of D is to compare the similarity between twosolutions. It measures the number of bit difference betweenthe directed disjunctive arcs representation of two solutionsas defined in (9). The EDisj of both Sp and Sq are representedas a total-ordering bitmap which consists of a string ofbooleans.

## \_ \_

## ( )

## ( ), ( )

Disj pDisj p Disj qE Sdiff E S E SD \_ (9)The solution Sq that is within a distance of replacementthreshold DR from Sp will be replaced, if one exists. If thereare two or more solutions in the list that meet the threshold, all of them will be replaced accordingly. Otherwise, Sp willbe appended to WL. Step 1: If Sp. Cmax < Cmax\_bestSearch through WL and accumulate Sp in WL by replacingsolutions which are within the replacementthreshold, DR. Sp. CtElite \_ 0. Step 2: For each entry Sq in WLIf Sq. CtElite > NAttemptWL. remove(Sq)Figure 8: Algorithm for PerformWaggleDance(). To make sure that every solution in WL is searched intensivelyby bees, a counter, CtElite, is associated to eachsolution. CtElite is incremented whenever the associated solution(dance) is used (followed) by another bee. When asolution is used for NAttempt (a predefined value) times, theentry will be removed from the list. By using the abovestrategy, exploitation of different peaks in the Big Valleylandscape is preserved. Figure 8 shows the algorithm forthe dance accumulation strategy which corresponds to thePerformWaggleDance() method in Figure 6.

## 4. 4. 2 Dance Observation Strategy

Before a bee leaves its hive to start the foraging process, itdecides if it will observe and follow a dance shown by previousdancer with a probability of Pfollow. In the BCO algorithmsuggested by Chong et al. (2006), Pfollow is adjusteddynamically via a profitability lookup table. A differentstrategy is adopted in BCBV. Pfollow is adjusted based onthe profitability rating of a bee, Pfi and the average profitabilityof the waggle dance list, PfWL. Pfi and PfWL are definedin (10), and (11) respectively: i i CPfmax\_ 1 (10)\_\_ \_ nWL i i n CPf 1max1 1 (11)where Cimax is the makespan of the schedule generated by aforager fi (Chong et al., 2007) and n is the number of dancingbee. Pfollow is determined by (12) where the 0. 9 and 0. 6are obtained after a series of parameter tuning tests.

## \_ \_ \_

## \_

## \_

otherwisePf PfP i WLfollow 0. 00, 0. 60, 0. 90\* (12)

## 4. 4. 3 Dance Selection Strategy

After a bee has decided to observe and follow a dance, thenext step is to select a dance from WL. Some selectionstrategies tested in the development of the BCBV algorithmare: \_ Most Recently Added (MRA) – the most recentlyadded solution to WL is selected. \_ Round Robin (RR) – the solution is selected in around robin manner, one after another in succession. \_ Roulette Wheel (ROUL) – the solution is selectedbased on the Cimax. A more profitable dance will havea higher chance to be selected. \_ Random Improving Move (RIM) – the solutionthat is better than the current makespan in WL is selectedin an unbiased manner. These strategies are aimed at exploring the Big Valleylandscape more effectively. As initial experiments showthat RR gives the best results on a set of benchmark dataset, it is adopted in the BCBV model in this paper. Figure 9shows the algorithm which corresponds to the ObserveNSelectDance() method in Figure 6. Step 1: Pfollow \_ 0. 00Step 2: Sx \_ solution found at previous iteration. Step 3: If Pfi < 0. 90 \* PfWLPfollow \_0. 60Step 4: Generate a random number r \_ [0, 1]. Step 5: If r < PfollowSx \_ Pick a solution from WL using a dance selectionstrategy. Step 6: Return Sx. Figure 9: Algorithm for ObserveNSelectDance().

## 5 EXPERIMENTS AND RESULTS

In this section, the benchmark problems, benchmark algorithmsand some experimental results will be presented. The experiments are conducted on a 16-node Linux clusterwhere each node has two Dual Core Xeon 3. 0GHz processorswith 4GB RAM.

## 5. 1 Benchmark Problems

The Taillard JSSP dataset used in this paper is obtainedfrom the library of JSSP sample instances maintained byTaillard, E. (http://mistic. heig-vd. ch/taillard/). Our experimentsfocus on the first 50 problem instances (ta\_01-ta\_50)for which only 19 optimal solutions have been found todate. The sizes of these problems range from 15 to 30 jobs2055Wong, Puan, Low, and Chongand 15 to 20 machines. The performance measure used isthe percentage difference in makespan \_ of a solutionfound, Cmax, from the known optimum/upper bound, Coptimal, calculated as in (13): \_ = \_\_ \_ \_ optimal optimal C C / C max \_ (13)

## 5. 2 Benchmark Algorithms

To evaluate the performance of the proposed bee colonyalgorithm, we include three other meta-heuristics in ourexperimental study. The first is the TSA by Nowicki andSmutnicki (1996) which is considered to be one of the bestin the class of optimization algorithms for the JSSP. Thesecond is the SBP for the JSSP proposed by Adams et al.(1988). The third is the BCNS by Chong et al. (2007).

## 5. 3 Parameter Settings

As TSA and SBP are deterministic algorithms, results forthese two algorithms are taken after one run. In contrast, the experimental results for BCNS and BCBV are the averagesover five runs due to their stochastic characteristic. Table 2: Parameter Setting for the Algorithms.

## Parameter TSA SBP BCNS BCBV

Nmax \_T \_T 2000 2000Population, l n. a. n. a. No. ofJobs10Alpha, \_ n. a. n. a. 1. 0 n. a. Beta, \_ n. a. n. a. 1. 0 n. a. Rating, \_ij n. a. n. a. 0. 99 n. a. Waggle dance scalingfactor, An. a. n. a. 100 n. a. Probability to performwaggle dance, pn. a. n. a. 0. 001 n. a. Max. no. of Elite Solution20 n. a. 20 \_Max size of Taboo List 8 n. a. n. a. 15Waggle dance replacementthreshold, DRn. a. n. a. n. a. 0. 15Waggle dance recruitmentcounter, NAttemptn. a. n. a. n. a. 50\_T denotes the experiments are allowed to run until termination. Table 3: Impact of Different DR Settings in BCBV. DR \_(%)

## Min Max Mean

0. 05 4. 88 12. 17 9. 110. 15 3. 84 11. 13 8. 020. 25 4. 54 11. 36 8. 380. 35 4. 49 11. 14 8. 920. 45 4. 41 13. 17 8. 86Due to the different natures of the four algorithms, it isdifficult to allocate the exact same amount of computationtime and resource to each of them. For a meaningful comparison, both TSA and SBP were allowed to run until termination. Both BCNS and BCBV were allowed to run for2000 iterations. The parameter settings for all four algorithmsare presented in Table 2. The parameter settings areobtained after a series of tuning experiments. Table 3 shows the impacts of different DR settings onBCBV on a set of five representative problems from Taillarddataset. They include ta\_04, ta\_13, ta\_29, ta\_40, ta\_41. The average \_ as well as the maximum and minimum\_ of the five problems are shown in the table. DR controls the number of distinct peaks (local optima)in the landscape stored in WL. When DR is set toolow (e. g. 0. 05), too many solutions are stored in WL leadingto slow convergence of the search. When DR is set toohigh (e. g. 0. 45), only a few solutions are stored in WL leadingto insufficient diversification in the search. Setting DRto 0. 15 gives the best result for the five problem instances.

## 5. 4 Experimental Results

Table 4 summarizes the performance results of the four algorithmsinvestigated. Besides the average, minimum andmaximum \_, the number of best solutions found amongstthe four algorithms on the 50 problem instances is also reported. From the results presented in Table 4, TSA recordsthe closest results to the optimal and find the best resultsfor 37 out of 50 problem instances. In comparison withTSA, BCBV gave similar performances for both the meanand maximum \_. BCBV also obtained a lower \_ and foundmore best solutions compared to BCNS and SBP. Whileboth bee heuristics underperform TSA, they offer a comparableperformance after 2000 iterations and their performancecan be improved when more iterations are used. Fromthe CPU time (speed) perspective, our experimental resultsshow that BCBV is 2. 6 times slower than TSA. Table 4: Performance of TSA, SBP, BCNS and BCBV on50 Problem Instances in Taillard JSSP Benchmark.

## Category TSA SBP BCNS BCBV

Mean \_ (%) 6. 93 11. 77 9. 16 8. 43Min \_ (%) 1. 04 5. 29 2. 84 4. 01Max \_ (%) 13. 66 20. 63 16. 22 13. 93Best Solutions Found 37 0 2 11

## 5. 5 Long Running Behaviors of BCBV

To see the long running performance of BCBV algorithmagainst BCNS and TSA, the averaged \_ over five problemsversus the number of iterations is plotted in Figure 10. Thesame set of five problems as mentioned in Section 5. 3 isused. A snapshot is taken at every 1000-th iteration wherethe average \_ for each algorithm is calculated. To investigate whether TSA, BCNS and BCBV have asimilar starting point, a snapshot on the average \_ is takenat the initial stage. Results show that \_ is 24. 10% for TSA, 2056Wong, Puan, Low, and Chong22. 02% for BCNS, and 21. 14% for BCBV, which showsthat these three algorithms have similar starting points. 5101520251 11 21 31 41 51 61 71 81 91Iterations ('000)

## \_ (%)

BCNS TSA BCBVFigure 10: Long Running Behaviors of TSA, BCNS andBCBV. The TSA algorithm converges very quickly to find itsbest solution, leveling off at approximately 31000 iterationsand showing no performance improvement thereafter. This is due to the termination of the algorithm by its stoppingcondition when its elite solution list becomes empty. The graph for TSA is extended beyond its terminationpoint for comparison with BCBV and BCNS. The BCNSalgorithm converges much slower (in terms of number ofiterations) and takes approximately 85000 iterations toreach the performance level of the TSA algorithm. Convergingslower than TSA, the BCBV algorithm starts tooutperform TSA at approximately 13000-th iteration andits solutions continued to improve before leveling off atapproximately 37000-th iteration. The trend of BCBV andBCNS suggests that by taking such an evolutionary approach, the quality of the solutions obtained may improvefurther given a longer running time. This characteristic isimportant if the objective is to obtain very high quality solutionsgiven ample computational facilities and time.

## 6 CONCLUSION

A Bee Colony Optimization algorithm with Big Valleylandscape exploitation has been proposed to solve JSSPproblem. The algorithm is based on the foraging behaviorof honey bees where the waggle dance is used as a communicationmedium among bees. To have an effective exploitationand exploration on the landscape structures, accumulationand selection strategies are introduced on thesolution list WL with waggle dances that represents peaksin a landscape. Experimental results on the Taillard JSSPbenchmark show that BCBV is able to achieve comparableperformance to TSA, SBP and BCNS using small numberof iterations. Given ample computation time, BCBV is ableto deliver performance better than TSA, SBP and BCNS. For future works, we will continue to explore otherfeatures of the Big Valley landscape as well as parametertuning to improve the performance of the BCBV algorithm. We will also incorporate other bee related featuressuch as the direction of waggle dance and scout bees in orderto make the algorithm more comprehensive.

## ACKNOWLEDGMENT

The authors would like to thank Science University of Malaysiaand Ministry of Higher Education of Malaysia forthe scholarship awarded to Li-Pei Wong to pursue hisPh. D. in Nanyang Technological University, Singapore.