

A HYBRID PARALLEL GENETIC ALGORITHM FOR SOLVING JOB-SHOP
SCHEDULING PROBLEMS

GAN TECK HUI

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Master of Engineering (Electrical)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

JULY 2005

UNIVERSITI TEKNOLOGI MALAYSIA

BORANG PENGESAHAN STATUS TESIS^u

JUDUL: A Hybrid Parallel Genetic Algorithm For Solving Job Shop
Scheduling Problems

SESI PENGAJIAN: 2004/2005

Saya GAN TECK HUI
(HURUF BESAR)

mengaku membenarkan tesis (PSM/Sarjana/Doktor Falsafah)* ini disimpan di Perpustakaan Universiti Teknologi Malaysia dengan syarat-syarat kegunaan seperti berikut:

1. Tesis adalah hakmilik Universiti Teknologi Malaysia.
2. Perpustakaan Universiti Teknologi Malaysia dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (4)

SULIT

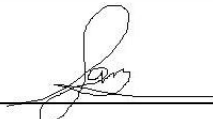
(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

TERHAD

(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)

TIDAK TERHAD

Disahkan oleh



(TANDATANGAN PENULIS)



(TANDATANGAN PENYELIA)

Alamat Tetap:

707-C, Jalan Perak, Taman Megah
75450, Bukit Beruang
Melaka

Prof. Dr. Marzuki Khalid

Nama Penyelia

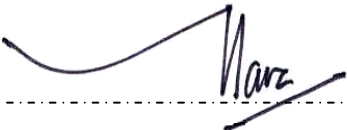
Tarikh: 18 - 07- 2005

Tarikh: 18 - 07- 2005

CATATAN: * Potong yang tidak berkenaan.
** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali sebab dan tempoh tesis ini perlu


^u Tesis dimaksudkan sebagai tesis bagi Ijazah Doktor Falsafah dan Sarjana secara penyelidikan, atau disertasi bagi pengajian secara kerja kursus dan penyelidikan, atau Laporan Projek Sarjana Muda (PSM).

“We hereby declare that we have read this thesis and in our opinion this thesis is sufficient in terms of scope and quality for the award of the degree of Master of Engineering (Electrical)”

Signature : 

Name of Supervisor I : .Prof. Dr. Marzuki Khalid.....

Date : .18 – 07 - 2005.....

Signature : 

Name of Supervisor II : .Prof. Madya. Dr. Rubiyah Yusof..

Date : ..18 – 07 - 2005.....

BAHAGIAN A – Pengesahan Kerjasama*

Adalah disahkan bahawa projek penyelidikan tesis ini telah dilaksanakan melalui kerjasama antara _____ dengan _____

Disahkan oleh:

Tandatangan : _____ Tarikh : _____

Nama : _____

Jawatan : _____

(Cop rasmi)

** Jika penyediaan tesis/projek melibatkan kerjasama.*

BAHAGIAN B – Untuk Kegunaan Pejabat Sekolah Pengajian Siswazah

Tesis ini telah diperiksa dan diakui oleh:

Nama dan Alamat Pemeriksa Luar :..Prof. Dr. Abdul Razak bin Hamdan.....
..Jabatan Sains & Pengurusan System,.....
..Fakulti Teknologi & Pengurusan System
..UKM, 43600 UKM, Bangi.....
..Selangor Darul Ehsan.....

Nama dan Alamat Pemeriksa Dalam :..Dr. Hj. Mohamed Khalil bin Hj.Mohd Hani.
..Fakulti Kejuruteraan Elektrik, UTM.....
..81310 UTM Johor, Malaysia.....

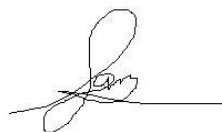
Nama Penyelia Lain (jika ada) :.....
.....
.....

Disahkan oleh Penolong Pendaftar di SPS:

Tandatangan : Tarikh :.....

Nama :.....

I declare that this thesis entitled “*A Hybrid Parallel Genetic Algorithm for Solving Job-Shop Scheduling Problems*” is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.



Signature :
Name : ..Gan Teck Hui.....
Date : ..18 – 07 - 2005.....

To My Beloved
Father, mother, brother and sisters

ACKNOWLEDGEMENT

In the preparation of this thesis, many people have contributed towards my understanding and thoughts. I wish to express my deepest gratitude to my thesis supervisor, Professor Dr. Marzuki Khalid and co-supervisor, Assoc. Prof. Dr. Rubiyah Yusof for their encouragement and guidance. I am also thankful to Mr. Tay Cheng San and Mr. Haruki Inoue for their valuable suggestions in this thesis.

Last but not least, I would like to express my appreciation toward all my family members, Ms. Yeo Lee Ling and all my colleagues who have provided assistance and showed their support when I needed them most.

ABSTRACT

The effort of searching an optimal solution for scheduling problems is important for real-world industrial applications especially for mission-time critical systems. In such an environment, an optimal result should be obtained within a relatively reasonable time. Genetic algorithms (GA) have long been applied to the many scheduling problems especially for solving job shop scheduling problems. However there are several problems when implementing GA into job shop scheduling problem (JSSP) such as slow and premature convergence. GA on one single personal computer is very time consuming while harder problems need bigger population and this has translate directly into higher computational costs. Therefore, this research proposes a different approach in solving JSSP using GA. Instead of conventional approach of using “serial” GA on a single PC, this research looks into the possibility of using parallel and distributed computing techniques and parallel genetic algorithm (PGA) in solving the same problems. Island model is chosen in this research to suit the implementation of distributed computing environment. The proposed PGA is a combination of both Asynchronous Colony Genetic Algorithm (ACGA) and Autonomous Immigration Genetic Algorithm (AIGA). This hybrid PGA is originally applied to symmetric multiprocessor machine (SMP) by Haruki Inoue. This type of modeling of PGA is based on the biologically reported observation that isolated environments, such as islands, often produce animal species that are more specifically adapted to the peculiarities of their environments than corresponding areas of wider surfaces. This theory also led to the hypothesis that several competing subpopulations could be more search-effective than a wider one in which all the members are held together. Each “island” is a different type of serial GA which represents one PC. Each island is using a combination of different crossover operator (GOX, Giffler and Thompson (GT) Crossover) and selection method (random selection, roulette wheel selection) into different types of serial GA. Every “island” has a certain convergence point where the improvement of the population fitness is almost stagnant. Therefore to solve this kind of undesired situation; a more sophisticated idea is used in this coarse grained PGA. This model of parallelization introduces a migration operator that is used to send some individual from one “island” to another “island”. The individual can migrate to any other “island” at their own judgment. Communication overhead during migration process is reduced by implementing a global mailbox method. Results show that PGA with a combination of different types of GA that creates a partially isolated environment for each sub demes, allowing a wider and more effective search, has outperformed PGA with a single type of GA. This research does not consider network topology in the scope as the system is linked via office network. To further improve the robustness of the system in the future, functions such as process migration and parallel input/output - by migrating intensive I/O processes to file servers (rather than the traditional way of bringing data to the processes) should be developed.

ABSTRAK

Usaha untuk mendapatkan penyelesaian yang optima untuk masalah penjadualan (*scheduling*) adalah sangat penting di dalam aplikasi dunia industri sebenar terutamanya sistem misi kritikal. Dalam situasi tersebut, keputusan yang optima sepatutnya dapat diperolehi dalam masa yang singkat. Algoritma Genetik (GA) telah lama diaplikasikan dalam banyak penyelesaian masalah penjadualan terutamanya untuk menyelesaikan masalah JSSP (*job shop scheduling problems*). Walau bagaimanapun, masih terdapat banyak kelemahan apabila mengimplikasikan GA dalam JSSP seperti penumpuan (*convergence*) yang perlahan. Operasi GA menggunakan komputer peribadi tunggal memakan masa yang lama manakala masalah lebih sukar memerlukan populasi yang lebih kompleks. Oleh itu, penyelidikan ini mencadangkan satu pendekatan yang berbeza dalam menyelesaikan JSSP dengan menggunakan GA. Tesis ini menyiasat kemungkinan menyelesaikan masalah JSSP dengan teknik pemprosesan selari and PGA (*parallel genetic algorithms*). Model pulau (*island model*) telah dipilih sebagai model utama yang bersesuaian dengan teknik pemprosesan selari dalam tesis ini. PGA yang dicadangkan adalah merupakan gabungan dari ACGA (*asynchronous colony genetic algorithm*) dan AIGA (*autonomous immigration genetic algorithm*). PGA hibrid ini pernah diimplikasikan oleh Haruki Inoue pada SMP (*symmetric multiprocessor machine*). Model PGA ini adalah berdasarkan kepada pemerhatian biologi yang mengatakan bahawa dalam suasana terasing, pulau biasanya menghasilkan spesis haiwan yang mempunyai ciri-ciri tertentu yang bersesuaian dengan suasana pulau tersebut. Teori yang satu lagi mengatakan bahawa beberapa populasi yang kecil dan bersaing adalah lebih baik daripada satu populasi yang besar dalam proses mencari penyelesaian. Setiap “pulau” yang dicadangkan dalam penyelidikan ini menggunakan jenis GA yang berbeza. Setiap GA terdiri daripada gabungan operator pengabungan (*crossover*), cara pemilihan (*selection*) dan lain-lain yang berbeza. Tiap-tiap “pulau” mempunyai titik penumpuan di mana nilai *fitness* populasi menjadi statik. Oleh itu, untuk menyelesaikan masalah seperti ini, salah satu idea yang lebih canggih ialah menggunakan model PGA yang dicadangkan. Model PGA ini memperkenalkan satu operator penghijrahan yang digunakan untuk menghantar beberapa individu dari satu “pulau” ke “pulau” yang lain. Individu ini akan berhijrah ke “pulau” lain bergantung kepada keputusan masing-masing. Kos komunikasi semasa migrasi telah dikurangkan dengan satu teknik yang dipanggil “peti surat global” (*global mail box method*) Pengujian dengan konfigurasi PGA seperti dikatakan tadi telah mempercepatkan proses GA dan memendekkan masa yang diperlukan untuk memperolehi satu penyelesaian yang baik. Prestasi PGA dalam menyelesaikan JSSP adalah lebih baik berbanding PGA dengan hanya satu jenis GA. Untuk meningkatkan keupayaan system ini, fungsi seperti migrasi proses dan masukan/keluaran selari melalui migrasi masukan/keluaran intentif ke “server” fail harus dibangunkan.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
1	INTRODUCTION	
	1.1 Introduction	1
	1.2 Literature Review	3
	1.3 Objectives of Research	9
	1.4 Scope of Research	9
	1.5 Thesis Layout	9
2	GENETIC ALGORITHMS	
	2.1 What are Genetic Algorithms	11
	2.2 Basic Structure of a Genetic Algorithm	13
	2.3 How Do Genetic Algorithms Work?	15
	2.3.1 Initialization	15
	2.3.2 Selection	16
	2.3.3 Reproduction	18
	2.3.4 Crossover Operator	18
	2.3.5 Inversion	20
	2.3.6 Mutation	20
	2.3.7 Evaluation	21
	2.4 A Numerical Demonstration of a Simple Genetic Algorithms Example	22
	2.5 Advance Techniques in Genetic Algorithms	26
	2.5.1 Hybridization	27
	2.5.2 Fitness Technique	28

	2.5.2.1 Linear Scaling	28
	2.5.2.2 Windowing	29
	2.5.2.3 Linear Normalization	29
3	JOB SHOP SCHEDULING PROBLEMS	
	3.1 General Introduction to Scheduling	31
	3.2 Classifications of Scheduling Problems	33
	3.2.1 Single Machine Shop	33
	3.2.2 Parallel Machine Shop	34
	3.2.3 Flow Shop Scheduling	34
	3.2.4 Job Shop Scheduling	35
4	PARALLEL GENETIC ALGORITHMS	
	4.1 Introduction to Parallel Genetic Algorithms	40
	4.2 Parallel Computers	41
	4.2.1 Types of parallel computers	41
	4.2.1.1 Shared Memory Multiprocessor System	42
	4.2.1.2 Message-Passing Multicomputer	43
	4.2.1.3 Distributed Shared Memory Computer	45
	4.2.2 Programming Model	46
	4.3 Classification of PGAs	48
	4.3.1 Global Parallelization	48
	4.3.2 Coarse Grained PGAs	50
	4.3.3 Fine Grained PGAs	55
	4.3.4 Hybrid Algorithms	58
	4.4 Underlying Problems	59
	4.4.1 When should migration happen? (A migration timescale)	59
	4.4.2 How often should migration happen? (Migration rate)	59

4.4.3	What is the best topology?	60
5	IMPLEMENTATION OF SEQUENTIAL GENETIC ALGORITHMS TO JOB SHOP SCHEDULING PROBLEMS	
5.1	Introduction	61
5.2	Assumption in JSSP	61
5.3	Applying Sequential Genetic Algorithms to JSSP	62
5.3.1	Representation	62
5.3.2	Fitness	63
5.3.3	Building a Schedule	65
5.3.3.1	A Simple Example of Building a Schedule	68
5.3.4	Selection Method	70
5.3.5	Crossover Operator	71
5.3.5.1	Generalized Order Crossover (GOX)	71
5.3.5.2	Giffler and Thompson Algorithm-based Crossover (GT)	73
5.3.6	Random Number Generator	74
5.4	Types of Sequential GA Developed	74
5.4.1	Conventional Genetic Algorithm	75
5.4.1.1	Combination 1	76
5.4.1.2	Combination 2	76
5.4.1.3	Combination 3	77
5.4.2	Micro Genetic Algorithm	78
5.5	Experimental Setup	81
5.5.1	Calibration Test I – Mutation Rate for Conventional GA	81
5.5.2	Calibration Test II – Mutation Rate for Micro GA	83
5.5.3	Experiments on JSSP – Results and Discussions	85

6	IMPLEMENTATION OF PARALLEL GENETIC ALGORITHMS TO JOB SHOP SCHEDULING PROBLEMS	
6.1	Applying PGA to JSSP	91
6.2	Asynchronous Colony Genetic Algorithm (ACGA)	92
6.3	Autonomous Immigration Genetic Algorithm (AIGA)	93
6.4	Hybrid PGA Model	95
6.5	Migration in Parallel Genetic Algorithms	98
6.6	Software Applied in Proposal PGA Implementation	98
6.7	Creating the Global Mailbox	99
6.8	Spawning	101
6.9	Message Passing Communication	101
6.10	Monitoring Program	102
6.11	PGA implementation – Experiments, Results and Discussions	103
6.11.1	Experiment 1 – Effect of Migration Rates	104
6.11.2	Experiment 2 – Comparing the performance on PGA and Sequential GA	107
6.11.3	Experiment 3 - Number of Running Sub-GAs	110
6.11.4	Experiment 4- PGA with multiple combination of sub-GAs	111
7	CONCLUSION AND FUTURE WORKS	
7.1	Conclusion	113
7.2	Future Works	114
	REFERENCES	116
	Appendix A	124

LIST OF TABLES

TABLE NO.	TITLE	PAGE
2.1	Corresponding Term between Natural and Artificial Terminology.	13
2.2	List of generated individuals	23
2.3	Evaluation results	23
2.4	Reproduction results	24
2.5	Reproduction results after mutation	26
2.6	Comparison of fitness values from the various techniques	30
3.1	Example of 3x3 JSSP	36
5.1	Basic structure of gene representation	63
5.2	Example of a 3 x 2 chromosome	65
5.3	List of dispatching rules	68
5.4	2x3 JSSP example	69
5.5	List of schedulable operation	69
5.6	List of dispatching rules	69
5.7	Conventional GA – combination 1	76
5.8	Conventional GA – combination 2	77
5.9	Conventional GA – combination 3	77

5.10	Micro-GA – combination 1	80
5.11	Micro-GA – combination 2	80
5.12	Micro-GA – combination 3	80
5.13	Results for different combination of GA operator on conventional GA	86
5.14	Results for different combination of GA operator on Micro-GA	87
6.1	Results for multiple migration rate	105

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
1.1	Example of Disjunctive Graph Representation	4
2.1	Pseudo code of a simple genetic algorithm	13
3.1	Single Machine Shop	33
3.2	Parallel Machine Shop	34
3.3	Gantt chart for Machines	37
3.4	Gantt chart for Jobs	38
3.5	Venn Diagram	39
4.1	Conventional computer having a single processor and memory	42
4.2	Traditional shared memory multiprocessor model	43
4.3	Message passing multiprocessor model	44
4.4	Shared memory multiprocessor implementation	46
4.5	MPMD structure	47
5.1	Flow chart of random schedule builder	66
5.2	Flow chart of schedule builder with heuristic	67
5.2	Parent 1 after step 1	71
5.4	Parent 2 after step 1	71

5.5	Parent 2 after step 2	71
5.6	Parent 1 marked for delete after step 3	72
5.7	Offspring produced after step 4	72
5.8	Offspring repaired after step 5	72
5.9	Parent 2 in case 2	72
5.10	Parent 1 marked for delete in case 2	72
5.11	Offspring produced in case 2	72
5.12	Flow chart of a conventional GA	75
5.13	Flow chart of implemented Micro-GA	79
5.14	Calibration test I result for conventional GA – combination 1	82
5.15	Calibration test I result for conventional GA – combination 2	82
5.16	Calibration test I result for conventional GA – combination 3	83
5.17	Calibration test II result for micro-GA – combination 1	84
5.18	Calibration test II result for micro-GA – combination 2	84
5.19	Calibration test II result for micro-GA – combination 3	85
5.20	Differences between best and average results for combination (1) [conventional GA]	88
5.21	Differences between best and average results for combination (2) [conventional GA]	88
5.22	Differences between best and average results for combination (3) [conventional GA]	89
5.23	Differences between best and average results for combination (1) [micro-GA]	89

5.24	Differences between best and average results for combination (2) [micro-GA]	90
5.25	Differences between best and average results for combination (3) [micro-GA]	90
6.1	Illustration of ACGA	93
6.2	AIGA structure	94
6.3	Hybrid PGA	96
6.4	Flow chart of hybrid PGA	97
6.5	Receive mail from global mailbox.	100
6.6	Delete and send mail to global mailbox	100
6.7	Retrieve information on the whole mailbox	101
6.8	Example of pvm_spawn function	101
6.9	Sending initialization process.	102
6.10	Example of receiving and unpacking the string function.	102
6.11	Process monitoring screen shot	103
6.12	Graphical monitoring program screen shot	103
6.13	Results of multiple migration rate on data sets	106
6.14	Experiments results on Ft10	107
6.15	Experiment results on La02	108
6.16	Experiment results on La03	109
6.17	Experiment results on La05	109
6.18	Experiment results of having multiple GAs on a single PC	110
6.19	Comparing results for PGA on Ft10	111
6.20	Comparing results for PGA on La02	112

6.21	Comparing results for PGA on La03	112
------	-----------------------------------	-----

LIST OF SYMBOLS

ACGA	-	Asynchronous colony genetic algorithm
AIGA	-	Autonomous immigration genetic algorithm
DNA	-	Deoxyribonucleic acid
EDD	-	Earliest due date
FCFS	-	First come first serve
FIFO	-	First in first out
GA	-	Genetic algorithms
GOX	-	Generalized order crossover
GT	-	Giffler and Thompson based crossover
GTA	-	Genetic tree algorithms
JSS	-	Job shop scheduling
JSSP	-	Job shop scheduling problems
LAN	-	Local area network
LOR	-	Least operation remaining
LPT	-	Longest processing time
LWR	-	Least work remaining
MIMD	-	Multiple inputs multiple data
MOR	-	Most operation remaining
MPI	-	Message passing interface
MPMD	-	Multiple programs multiple data
MT	-	Mersenne Twister
MWR	-	Most work remaining
NP	-	Non-polynomial
OR	-	Operation research

PC	-	Personal computer
PGA	-	Parallel genetic algorithms
PPSN	-	Parallel Problem Solving from Nature workshop
PVM	-	Parallel virtual machine
RAM	-	Random access memory
RNA	-	Ribonucleic acid
SMP	-	Symmetrical multiprocessors machine
SPMD	-	Single program multiple data
SPT	-	Shortest processing time
C_{\max}	-	Makespan
f_{avg}	-	Fitness average
f_{\max}	-	Fitness maximum
f_{\min}	-	Fitness minimum

CHAPTER 1

INTRODUCTION

1.1 Introduction

When the manufacturing world grows more sophisticated, the problems involved seem to be getting harder to solve as well. Many applications appear on the market as the need to use material and human resources more efficiently and effectively arise. The job shop scheduling problem (JSSP) is a notoriously difficult NP-hard combinatorial optimization problem. Similar to the process of the manufacturing world, it is consequently used as one benchmark for determining the effectiveness of local search algorithms. Each word in job shop scheduling has its own definition. Job is a piece of work that goes through a series of operations. Shop is a place for manufacturing or repairing of goods or machinery and scheduling is defined as a decision process aiming to deduce the order of processing. In short, job shop scheduling means an activity to allocate share resources over time to competing activities.

In today competitive environment where speed and efficiency are the main factors, conventional search algorithms are no longer adequate in solving manufacturing problems such as flow-shop scheduling and job-shop scheduling effectively. Traditional methods lack intelligence, robustness and flexibility to solve these problems. As a result, researches are looking into the possibility of applying artificial intelligence (AI) techniques for such problems. Two prominent fields arose, connectionism (neural networking, parallel processing) and evolutionary computing. Both of these major fields are more well-known as artificial intelligence. An AI system usually is capable of doing three things: 1) stores knowledge; 2) applies the

knowledge stored to solve problems; and 3) acquires new knowledge through experience (Haykin, 1994). AI can be categorized into many areas such as fuzzy logic, artificial neural networks, expert system, genetic algorithms, chaos theory, natural language system, etc. However, this thesis will only consider the area of evolutionary computing which is more well-known as genetic algorithms and genetic programming.

Genetic algorithms are among the techniques that is getting a lot of attention from researchers in many fields of engineering and manufacturing especially in the area of scheduling. Genetic algorithms are basically algorithms based on natural biological evolution. The architecture of systems that implement genetic algorithms (GA) is more capable of adapting to a wide range of problems. Generally, a GA functions by generating a large set of possible solutions to a given problem. It then evaluates each of the solutions, and decides on a "fitness level" for each solution set. These solutions then breed new solutions. The parent solutions that were more "fit" are more likely to reproduce, while those that were less "fit" are more unlikely to do so. In essence, solutions are evolved over time.

Genetic algorithms have been implemented successfully for many scheduling problems such as timetabling, one machine scheduling, job shop scheduling and so on. GA is preferred because of their capability of producing a good solution without searching through the whole search space by brute force like traditional method which might take a long time to reach an optimal solution with current computer power.

In some scheduling problems, especially in manufacturing industries, time taken to reach an optimal solution is a crucial factor. Therefore, it is essential to find ways of decreasing the processing time in finding the solution. One of the ways of achieving this requirement has led to the exploration of parallel genetic algorithms. Parallel genetic algorithm is another popular area of genetic algorithms. When we speak of parallel genetic algorithms, naturally it involves parallel computers as a mean of implementing the algorithms. Many types of parallel architectures have been introduced and employed by researchers. While some of the works use explicitly

parallelization of the genetic algorithms (much like the parallelization of any algorithm), some introduce changes to the architectural composition the way genetic algorithms function. While parallelizing the genetic operations (e.g. crossover, mutation, evaluation) over the population is an example to the efforts of the first kind, population migration is a good example to the second kind. (Onur, 2002).

1.2 Literature Review

From our review, many methods have been used to solve JSSP. For example, mathematical programming has long been applied extensively to job shop scheduling problems. Problems are normally formulated using integer programming, mixed-integer programming, linear programming and dynamic programming. However, the use of these approaches has been limited because scheduling problems belong to the class of NP-complete problems.

According to Balas (1969) and Roy et al (1964) the JSSP can be represented with a disjunctive graph. The disjunctive graph $G = (N, A, E)$ is defined as follows: N contains nodes representing all operations, A contains arcs connecting consecutive operations of the same job, and E contains disjunctive arcs connecting operations to be processed by the same machine. A disjunctive arc can be settled by either of its two possible orientations. The construction of a schedule will settle the orientation of all disjunctive arcs so as to determine the sequences of operation on same machines. Once a sequence is determined for a machine, the disjunctive arcs connecting operations to be processed by the machine will be replaced by the usual (oriented) precedence arrow, or conjunctive arc. Figure 1.1 illustrates the disjunctive graph for a three-job three machine instance, where each job consists of three operations.

Since the job shop scheduling problem is to find the order of the operations on each machine that is, to settle the orientation of the disjunctive arcs such that the resulting graph is acyclic (there are no precedence conflicts between operations) and that the length of the maximum weight path between the start and end nodes is

minimal. The length of a maximum weight (or longest) path determines the makespan. However, when come to flexibility and practicability, this method has some disadvantages as the data in each node of the arc is fix and cannot be change thus making it not flexible enough for real world application.

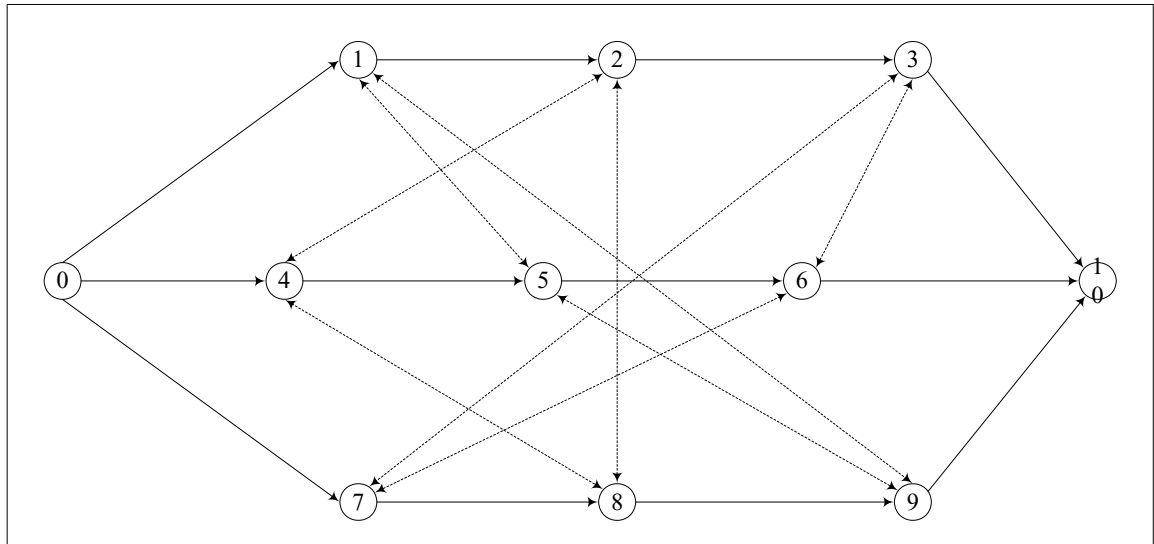


Figure 1.1 Example of disjunctive graph representation

Using heuristic rules in solving JSSP is not new. The heuristic procedures for a job-shop problem can be roughly classified into two classes: one pass heuristic and multi-pass heuristic. Base on priority dispatching rules, one pass heuristic builds up a single complete solution by fixing one operation at a time in the schedule. This type of conventional heuristic is fast and usually is used to find solutions that are not too difficult. While multi-pass heuristic is simply the combination of multiple one pass heuristics in order to obtain better schedules at some extra computational cost.

On the other hand, dispatching rules have been applied consistently to scheduling problems. They are procedures designed to provide good solutions to complex problems in real-time. The terms such as dispatching rule, scheduling rule, sequencing rule, or heuristic are often used synonymously in this area of research (J. Blackstone et al, 1982). Dispatching rules are named according to their performance criteria. D. Wu (1987) categorized dispatching rules into 3 classes. Class 1 contains simple priority rules, which are based on information related to the jobs. Class 2 consists of combinations of rules from class one. The particular rule that is

implemented can now depend on the situation that exists on the shop floor. Class 3 contains rules that are commonly referred to as Weight Priority Indexes. This heuristic is applied in this thesis.

The algorithms of Giffler and Thompson (1960) can be considered as basic of all priority rules based heuristics. Giffler and Thompson have proposed algorithms for active and non-delay schedules generation. The algorithms are based on a tree-structured approach. The nodes in the tree correspond to partial schedules, the arcs represent the possible choices and the leaves of the tree are the set of enumerated schedules. The algorithm essentially identifies all processing conflicts when a partial schedule first generated and an enumeration procedure is used to resolve the conflicts in all possible ways at each consecutive stage. By contrast, heuristic resolve there conflicts with priority dispatching rules; that is they specify a priority rule for selecting one operation among the conflicting operations. At each stage, a list of schedulable operations is generated. The list of operations is determined from the precedence structure. The valid operations (schedulable operations) are the operations with immediately scheduled predecessors which can be simply determined from the precedence structure. An extensive summary and discussion is being published by Panwalkar and Iskander (1977) and Haupt (1989).

Randomized heuristic as what it sounds like is based on the idea of randomly selecting a dispatching rule from a family of heuristic at each stage. At each selection of an operation, this algorithm will choose a dispatching rule randomly throughout an entire schedule generation. This algorithm is an early attempt to provide more accurate solution (Baker, 1974). Morton and Pentico (1993) proposed a guided random approach that used an excellent heuristic to explore the problem and provide good guidance as to where to search.

For NP-hard problem such as JSSP, neighborhood search methods are very popular. Neighborhood search methods have been proven to provide good solutions and have the potential to be enhanced when combined with other heuristics. One of the first neighborhood procedures was developed by L. Wilkerson and J. Irwin (1971). This method iteratively added small changes (“perturbations”) to an initial

schedule. Having the similar concept with hill climbing, these techniques continue to perturb and evaluate schedules until the termination condition is reached. Popular techniques such as Tabu search, simulated annealing, and genetic algorithms belong to this family.

The concept of Tabu search (F. Glover, 1996) is to explore the search space of all feasible scheduling solutions by a sequence of moves. Similar to gradient-based techniques, this method performs its search by moving from one schedule to another schedule, evaluating all candidates and selecting the best available. Those moves that trap the search at a local optimum, or lead to cycling (repeating part of the search) are classified as tabu (forbidden). These moves are put on the Tabu List that built up from the history of moves used during the search. Exploration of new search space is forced by these tabu moves until the old solution area (e.g., local optimum) is left behind. Another essential element in tabu search is that of freeing the search by a short term memory function that provides “strategic forgetting”. Adaptive Memory Programming (AMP) is the more advanced framework of the Tabu search methods. Tabu search methods have been applied successfully to scheduling problems and as solvers of mixed integer programming problems. Tabu search has difficulty in satisfying JSSP problem with high constraints and large neighborhood.

Simulated annealing is based on the analogy to the physical process of cooling and recrystallization of metals. Simulated annealing has three key elements: current state of the thermodynamic system, energy equation and ground state. These three elements are analogous to current scheduling solution, objective function, and the global optimum respectively. In addition to the global energy J , there is a global temperature T , which is lowered as the iterations progress. Using this analogy, the technique randomly generates new schedules by sampling the probability distribution of the system [S. Kirkpatrick et al, 1983]. Since increases of energy can be accepted, the algorithm is able to escape local minima. Simulated annealing has been applied effectively to scheduling problems. A. Vakharia and Y. Chang (1990) developed a scheduling system based on simulated annealing for manufacturing cells.

Fuzzy set theory being useful in modeling and solving problem with uncertainty has been used to solve job shop scheduling problem with uncertain processing time, constraint and setup time. The uncertainties in the problems are represented by fuzzy memberships that are described by using the concept of an interval of confidence. Fuzzy logic techniques are usually integrated with other methodologies (e.g., search procedures, constraint relaxation) in solving problems.

J. Krucky (1994) solved the problem of minimizing setup times of mix product production line using fuzzy logic. The heuristic that incorporated fuzzy logic into the algorithm assists in setup time minimization process by clustering assemblies into families of products that share the same setup by balancing a product's placement time between multiple-high-speed placement process steps. On the other hand, Y. Tsujimura et al (1993) developed a hybrid system that uses fuzzy set theory to model the processing times of a flow shop scheduling facility. These processing times are represented by Triangular Fuzzy Numbers (TFNs). Each job is defined by a lower bound and an upper bound TFNs. Minimization of makespan is done with a branch and bound procedure.

Reactive scheduling is generally defined as the ability to revise or repair a complete schedule that has been "overtaken" by events such as rush orders, excessive delays, and broken resources on the shop floor (K. Kempf, 1995). Scheduling system using the reactive repair waits until an event has occurred before it attempts to recover from that event. Proactive adjustment requires a capability to monitor the system continuously, predict the future evolution of the system, do an emergency planning for likely events, and generate new schedules, all during the execution time of the current schedule. R. Wysk et al (1986) , W. Davis and A. Jones (1988) are three researchers that study knowledge of this category. Approaches that are more recent utilize artificial intelligence and knowledge-based methodologies (S. Smith, 1995). Still most of the AI approaches propose a quasi-deterministic view of the system, i.e., a stochastic system featuring implicit and/or explicit causal rules. The problem formulation used does not recognize the physical environment of the shop floor domain where interference not only leads to readjustment of schedules but also imposes physical actions to minimize them.

T. Starkweather et al (1993) were the first researchers to implement genetic algorithms to a dual-criteria job shop scheduling problem in a real production facility. Both of the criteria were the minimization of inventory and the minimization of waiting time for an order to be selected. These criteria are inversely related (The smaller the inventory, the longer the wait, larger the inventory, the shorter the wait).

A symbolic coding was used for each member (chromosome) of the population to represent the production or shipping optimization problem. The Genetic Algorithm used to solve this problem was based on a modification to the blind recombinant operator described above. This recombination operator emphasized information about the relative order of the elements in the permutation, because this impacts both inventory and waiting time. A single evaluation function (a weighted sum of the two criteria) was utilized to rank each member of the population. That ranking was based on an on-line simulation of the plant operations. This approach generates schedules that produced inventory levels and waiting times that were acceptable to the plant manager. In addition, the integration of the genetic algorithm with the on-line simulation made it possible to react to system dynamics. These applications have emphasized the utilization of genetic algorithms as a "solo" technique. This has limited the level of complexity of the problems solved and their success. Recent research publications have demonstrated the sensitivity of genetic algorithms to the initial population. When the initial population is generated randomly, genetic algorithms are shown to be less efficient than the annealing-type algorithms, but better than the heuristic methods alone. However, if the initial population is generated by a heuristic, the genetic algorithms become as good as, or better than the annealing-type algorithms. In addition, integration with other search procedures (e.g., tabu search) has enhanced the capabilities of both. This result is not surprising, as it is consistent with results from non-linear optimization.

1.3 Objectives of Research

The objective of this research is to propose a new hybrid parallel genetic algorithm (PGA) for application in job shop scheduling problems. This objective is broken down to the following sub-objectives:

1. To investigate the performance of the proposed parallel genetic algorithm and compare with existing techniques.
2. To compare the performance of sequential GA and PGA under different combination of GA operators.
3. Reduce the communication overheads of the parallel computing (message passing) that contribute to more effective use of parallel computer power.

1.4 Scope of Research

The scope of research includes the following:

1. A comparative study of 6 combinations of GA operators using 20 benchmark data sets.
2. Investigate the performance of the 4 configurations of the parallel GA.
3. Investigating the effectiveness of a new hybrid parallel GA.
4. This research only concentrate on static job shop scheduling problem.
5. This thesis does not consider the network topology of a PGA into the research.

1.5 Thesis Layout

Chapter 2 introduces genetic algorithms. Each element in the structure of genetic algorithms is briefly described. This is followed by chapter 3 that provides an

introduction to job shop scheduling problems with a simple example of job shop scheduling problem.

Chapter 4 provides an overview of the parallel computer structure and model. A brief survey on available parallel genetic algorithms is discussed. Three classes of approaches are included, (i) global parallelization, (ii) coarse grained parallelization and (iii) fine grained parallelization.

Chapter 5 describes the implementation of sequential genetic algorithms developed in this research to solve JSSP with comparative results and discussions presented on various experiments.

Chapter 6 describes the implementation of proposed parallel genetic algorithms along with results and discussions on various experiments such as migration rate and migration interval.

Chapter 7 summarizes this research, discusses the outcome of experiment and proposed methods. Recommendations on future development are included in this chapter.