DYNAMIC TASK SCHEDULING ONTO HETEROGENEOUS MACHINES

USING SUPPORT VECTOR MACHINE

Except where reference is made to the work of others, the work described in this thesis is my own or was done in collaboration with my advisory committee. This thesis does not include proprietary or classified information.

Yongwon Park

Certificate of Approval:

David Umphress
Associate Professor
Computer Science and
Software Engineering

Sanjeev Baskiyar, Chair
Associate Professor
Computer Science and
Software Engineering

Levent Yilmaz
Assistant Professor
Computer Science and
Software Engineering

George T. Flowers
Interim Dean
Graduate School

DYNAMIC TASK SCHEDULING ONTO HETEROGENEOUS MACHINES

USING SUPPORT VECTOR MACHINE

Yongwon Park

A Thesis

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Master of Science

Auburn, Alabama
May 10, 2008

DYNAMIC TASK SCHEDULING ONTO HETEROGENEOUS MACHINES

USING SUPPORT VECTOR MACHINE

Yongwon Park

Signature of Author

Date of Graduation

THESIS ABSTRACT

DYNAMIC TASK SCHEDULING ONTO HETEROGENEOUS MACHINES

USING SUPPORT VECTOR MACHINE

Yongwon Park

Master of Science, May 10, 2008
(B.S., Kwangwoon University, 1997)

57 Typed pages

Directed by Sanjeev Baskiyar

Distributed computing has been used to overcome the limitations of single computer use. However, the benefit of parallelizing computations may substantially reduce, if there is no well constructed mechanism to coordinate them. In this respect, the task matching problem of mapping a class of independent tasks on heterogeneous computers is critical to increase system performance, especially if the purpose is to reduce the total completion time of tasks. Mapping tasks to non-identical machines is a known NP-complete problem. Many heuristic algorithms have been used to minimize the total completion time in parallel systems. In this thesis, we take a novel approach by using Support Vector Machine (SVM) to dynamically schedule independent tasks to heterogeneous machines to minimize schedule length.

SVM learns from a set of input workload patterns or samples. It maps each sample to a predefined label. Most learning samples of real world problems are non-separable in multi-dimensional input space. In our SVM, Radial Basis Function (RBF) Kernel is used to transform non-separable samples in multi-dimensional input space into high-dimensional feature space, where the samples are separable. The SVM constructs a hyperplane with maximal margin between the positive and negative samples in the high dimensional feature space. This hyperplane is used to classify future mappings.

We constructed a Support Vector Scheduler (SVS), which uses the SVM to map tasks to machines. Using simulations we compared our algorithm against Early Fast (EF), Light Least (LL), and Round Robin (RR). We found that the performance using SVM was similar to EF and better than LL and RR. However, SVM is superior since it can dynamically adapt to changing inputs and machine characteristics.

Style manual or journal used: IEEE style guide

Computer software used: Microsoft word 2003

TABLE OF CONTENTS

LIST OF FIGURES

1 INTRODUCTION

A very large problem can be solved in distributed computing systems by decomposing the problem into small tasks, and distributing the workload fairly, and combining individual results to get a solution. However, as the amount of homogeneous parallelism in applications decreases, homogeneous systems cannot offer the desired speedups. Therefore, a suite of heterogeneous architectures to exploit the heterogeneity in computations has become a critical research issue [Khok93]. Heterogeneous computing (HC) is the well-coordinated use of a suite of diverse high-performance machines to provide super-speed processing for computationally intensive tasks with diverse computing needs. However, the benefit of parallelizing can diminish, if there is no well-found mechanism to coordinate the resources. In this respect, the task-matching problem assigning independent tasks to the most suitable machines should be considered to achieve high performance, especially if the purpose of the system is to minimize the total completion time, or makespan[Brau01][Mahe99][Meht06][Page05][Fuji03]. In the general case, assigning independent tasks onto non-identical machines is known to be an NP-complete problem. So far, many studies have tried to solve the task matching problem [Brau01][Cho94][Hong04]

[Mahe98][Brau01][Cho94][Hong04][Mahe98][Poje02][Min97][Brau98]. In [Brau01], eleven conventional heuristic algorithms for mapping a class of independent tasks onto heterogeneous distributed computing machines are compared. The type of heuristic methods can be categorized into static mode and dynamic mode. In static mode, all the information necessary for scheduling, such as the expected task completion time of each machine, is known a priori. In contrast, the scheduling information in dynamic mode is not known until runtime. Furthermore, dynamic mode is classified into direct mode and batch mode. In direct mode, a task is dispatched to the appropriate machine on arriving, but in batch mode, it waits until a dispatching event occurs [Mahe99]. In heuristic algorithms, the choice of which computers to execute which tasks is commonly determined using the knowledge of computer speeds for each task and the current load on each computer [Freu98]. On the other hand, load sharing algorithms consider the average behavior of total systems, focusing on increasing processor utilization by not allowing any processor to be idle. In [Kara02], classes of independent tasks are mapped onto the heterogeneous computing system using a load sharing algorithm. Similar to load sharing algorithms, load balancing algorithms aim at equalizing the processors' workloads at the time of distribution in order to achieve the enhancement of system performance through system load balance. Load sharing algorithms, in contrast, focus on scattering heavy workloads into idle or light processors. Page and Naughton [Page05] use a genetic algorithm for dynamic task scheduling, in which a search for optimal schedules is made based on the theoretical optimal processing time. In this thesis, we introduce a new approach to solve the task mapping problem in which a machine learning technique is used for a direct task matching with the objective of minimizing the total task completion time. Furthermore, our task mapping scheduler is able to adapt to varying system environments by changing a decision model where computing powers can change suddenly with new computers joining or leaving the network. The decision for task mapping will be conducted based on the evaluation of Support Vector Machine (SVM).

For this research, we devised two simulator programs: a training data simulator (TDS), which creates training data for SVM and a Java simulator based on our Support Vector Scheduler (SVS) as well as three heuristic algorithms for benchmarking. We use the SVM*light*, an implementation of Vapnik's Support Vector Machine, which was written in C by Joakims [Scho99]. The remainder of this paper is organized as follows. In Section 2, we discuss research related to SVM. In Section 3, we review related scheduling heuristics. Section 4 is devoted to explaining our system framework in detail. In Section 5, the analysis of data obtained from simulation will be conducted and the SVS is compared against conventional heuristics such as Early Fast (EF), Round Robin (RR) and Light Least (LL) algorithms. Section 6 concludes by presenting a summary of our findings.

2 BACKGROUND

Recently, SVMs have gained wide acceptance in the application of pattern recognition and data mining. SVMs were proposed by Vapnic et al [Burg98]. It has shown better performance than traditional learning machines such as Neural Network (NN) and Decision Tree (DT). Its running strategy embodies the principal of structural risk minimization (SRM) while the objective of neural networks is only based on the principal of empirical risk minimization (ERM). Therefore, SVMs are able to possess high generalization ability while minimizing the training error. The subject of SVMs is said to have started in the late seventies but only recently it has gained attention with success in pattern recognition, object recognition, speaker identification, face detection, regression estimation and text categorization [Burg98]. SVMs are commonly used as a non-linear classifier through kernel trick, though it naturally was proposed as a linear classifier. So far, SVMs have rarely been used in the area of task scheduling, especially for direct task mapping. In [Gers04], a SVM is used for regression estimation to improve the repair strategy in which a complete schedule is found by iteratively repairing an incomplete schedule for solving a resource constrained project scheduling problem, known as NP-hard problem. Yi-Huung et al. [YiHu05] use a multi-class SVM in scheduling the Flexible Manufacturing System (FMS), in which the most suitable dispatching rule is decided by the SVM and task mapping is conducted according to this rule until it is replaced by a new dispatching rule.

3 TASK SCHEDULING ALGORITHM

In this section, we will review heuristic and machine learning approaches for solving the task mapping problem.

3.1 Heuristic Algorithm

First, heuristic algorithms can be categorized into batch mode and immediate mode algorithms. In immediate mode, the task is mapped onto the machine immediately upon arrival, but in batch mode, it is not scheduled until a mapping event occurs.

3.1.1 Immediate mode mapping heuristics

The Minimum Completion Time (MCT) heuristic is a variant of the fast-greedy heuristic. It has been used as a benchmark for immediate mode [Mahe99]. MCT assigns each task to the machine on which the task will complete the earliest. Braun et al. [Brau01] compared 11 heuristic algorithms and found MCT to perform around the median of heuristics. MCT requires $O(m)$ time to find the machine that can finish a task earliest, where m is the number of machines. In the Minimum Execution Time (MET) heuristic, as a job arrives, each task is assigned to the machine that provides the least execution time for that task. Although the MET heuristic is very simple with complexity $O(m)$, it may result in severe imbalance in load across the machines [Brau01]. All the computing nodes in the cluster are examined to determine the node that gives the best execution time for the job. As mentioned, MET may result in load imbalance at some point because it does not consider the ready time of each machine. To handle this problem, SA (Switching Algorithm) has been proposed [Brau01]. SA switches from MET to MCT when load imbalance is detected. SA has the same complexity with MCT and its performance is close to MCT [Brau01]. K-percent Best (KPB) heuristic implements the idea that too much selection pressure may lead to a sub optimal solution since it suppresses the diversity of search. The parameter K determines the selection pressure. Therefore, in KPB, a subset of machines, in which K is less than 100, is selected based on the earliest completion time. A task is assigned to the machine in the reduced set whose completion time is the least. That is, KPB looks forward to achieving the improvement in the long run by considering task heterogeneity, instead of promptly expecting the current marginal improvement. In a similar way, Feasible Robust K-percent Best (FRKPB) first finds the feasible set of machines for the newly arrived task. From this set, FRKPB identifies the k-percent that has the smallest execution times for the task [Meht06]. In Opportunistic load balancing (OLB), a naïve $O(n)$ algorithm [Arms98], each job is placed in order of arrival on the next available machine regardless of its completion time. The performance of OLB is worse than the other algorithms.

3.1.2 Batch mode heuristics

In batch mode, tasks are collected into a set that is examined for mapping at prescheduled times called mapping events. In immediate mode, they are mapped onto the machines immediately upon arrival. This mode enables the mapping heuristics to possibly make best decisions at every moment. Because the heuristics have the resource requirement information for a whole meta-task, when the task arrival rate is high, there will be a sufficient number of tasks to keep the machines busy between the mapping events.

The Min-Min heuristic algorithm[Brau01] uses an Expected Completion Time (ECT) table to make a decision for mapping a task onto a suitable machine. The ECT is defined as:

$$C_{ij} = E_{ij} + R_j \tag{3.1}$$

The completion time of task i in machine j is calculated by adding the ready time of machine j to its ECT (3.1). Basically, a task is assigned to the machine that provides minimum completion time. When there is a contention for the same machine on which two or more tasks are eligible, a task is assigned to the machine that will result in the smallest change in ready time. In this algorithm, it is expected that smaller makespans can be obtained if a larger number of tasks are assigned to the machine that not only completes them earliest but also executes them fastest. Max-Min heuristic is similar to Min-Min except a task with maximum completion time is chosen among the candidate tasks whose completion time is minimum in all the machines. The Max-Min heuristic is likely to be better when there are more short tasks than long tasks since it can execute many short tasks concurrently along with the long task. The main idea of the Sufferage heuristic is to assign a task to a machine that would suffer most if it were not assigned to a machine. The Sufferage algorithm uses the same ECT table as it is used in Min-Min heuristic. The algorithm is described in Figure 1. The key point of the algorithm is to use the sufferage value in task mapping. That is, a machine is assigned to the task that would "suffer" most in terms of expected completion time if that particular machine is not assigned to it. When trying to assign a new arbitrary task to a machine that can complete the task earliest, the machine may be in the state of having a task already assigned. If the machine is in the state with a task assigned, a new task and an old task will contend for the same machine. When tasks contend for the same machine, task assignment is determined by their sufferage value. The task replaced by the new task will come back to task queue and the new task will be removed from it.

```
For all tasks
        For all machines
               Update ECT for all tasks
                Find arbitrary a task with Earliest Completion time

                  If corresponding machine is already assigned a task
                         then
                             Calculate sufferage value
                             A task with higher sufferage value is assigned
                         else
                              Assign a machine that gives the earliest
                              completion time to a task tentatively.
                End If
        End For
End For
```

Figure 1 Sufferage Heuristic

3.2 GA (Genetic Algorithm)

Genetic Algorithms (GAs) are adaptive heuristic search algorithms based on the evolutionary ideas of natural selection and genetics. A group of individual solutions known as a population evolves by natural selection using various operators. Its basic procedure is described in Figure 2

Initialize Population

Evaluate Population

Loop until stopping condition not met

Select parent

Crossover

Mutation

Create offspring

Evaluate offspring

Select survivors

End Loop

Figure 2 Genetic Algorithm Procedure

In GA, selection is made according to the fitness of a population. The operators such as crossover and mutation are used to provide diversity in exploration.

In [Page05] GA is used to minimize the makespan, and the algorithm outperforms six other heuristic algorithms (about 10% better than EF). Task scheduling problems in network computing environments are solved using GAs in [Dong02].

3.3 Load sharing algorithm

In heterogeneous computing environments with two processors classes, fast and slow, job migration for distributing loads fairly over all processors is performed from slow to fast processors using six scheduling strategies: Probabilistic (Pr), Probabilistic with Migration of Generic Jobs (PrM), Shortest Queue (SQ), Shortest Queue with

Migration of Generic Jobs (SQM), Least Expected Response Time for Generic Jobs-Maximum Wait for Dedicated Jobs (LERT-MW), Least Expected Response Time for Generic Jobs-Maximum Wait for Dedicated Jobs with Migration (LERT-MWM) [Kara02]. In overall performance, SQ and SQM methods are better than all other methods.

3.4 Machine Learning

Scheduling plays an important role in production control for flexible manufacturing system (FMS), which involves several real-time decisions, such as part type and machine selection [YiHu05]. Consequently, a scheduled FMS is able to improve the machine utilization, enhance throughput, reduce the number of work-in-process (WIP), mean flow time, and the number of tardy parts. Assigning correct dispatching rules dynamically is critical for the scheduling problem. After receiving useful information from an FMS, a good scheduler should be able to make a right decision, i.e., output a right dispatching rule, for the next period to gain good performance. It needs as much expert knowledge stored in the scheduler as possible. Due to such reasons, machine learning technique, which is based on simulated sample data, has been used [YiHu05].

4 DYNAMIC SCHEDULER USING SVM

In this section, we will introduce our framework of task scheduling onto non-identical machines. Our scheduler is focused on minimizing total completion time by using a Support Vector Machine (SVM) in mapping tasks directly onto suitable machines. First, we present an overview of the Support Vector Machine.

4.1 Overview of SVM

SVM is a supervised learning algorithm developed over the past decade by Vapnik and others [Vapn98]. The SVM algorithm addresses the general problem of learning. The binary version of the SVM attempts to discriminate data into two different classes. It does so by constructing the optimal segregating hyperplane using a sample set of training data. Much of the SVM's power comes from its criterion of selecting a separating hyperplane when many other candidate planes may exist. In the optimal hyperplane, samples are separated with maximal margin. Statistical learning theory suggests that, for some classes of well-behaved data, the choice of the maximum margin hyperplane will lead to maximal generalization when predicting the classification of previously unseen examples [Vapn98]. The main element of support vector learning is to construct the optimal separating hyperplane. To construct the optimal hyperplane, we have to solve the quadratic programming (QP) problem:

$$\text{minimize } \frac{1}{2}\sum_{i,j=1}^{N}\alpha_i Q_{ij}\alpha_j - \sum_{i=1}^{N}\alpha_{i,}$$

subject to the constraints

$$0 \le \alpha_i \le C$$
$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

where, Q is an N x N matrix that depends on the training inputs x_i, the labels y_i, and the functional form of the SVM. We call this problem quadratic programming because the function to be minimized (called the objective function) depends on the α_i quadratically, while α_i only appears linearly in the constraints. Definitions and applications of x_i, y_i and Q appear in the tutorial by Bruges [Burg98].

The construction of an optimal hyperplane is depicted in Figure 3. Here, a set of training instances are represented by circles and squares, which denote positive and negative samples respectively. In the left graph, the samples are shown in the non-linear inputspace where they are not separable linearly or by a hyperplane. A mapping is performed to map the samples into the feature space using a non-linear mapping function, *Φ,* which transforms the multi-dimensional input space into a still higher dimensional feature space. In the feature space, samples are separable linearly (using a hyperplane) as shown in Figure 3. An optimal hyperplane in the feature space separates the squares and circles with the maximum margin *w*. The points that lie on the parallel planes that are closest to the optimal hyperplane are called support vectors.

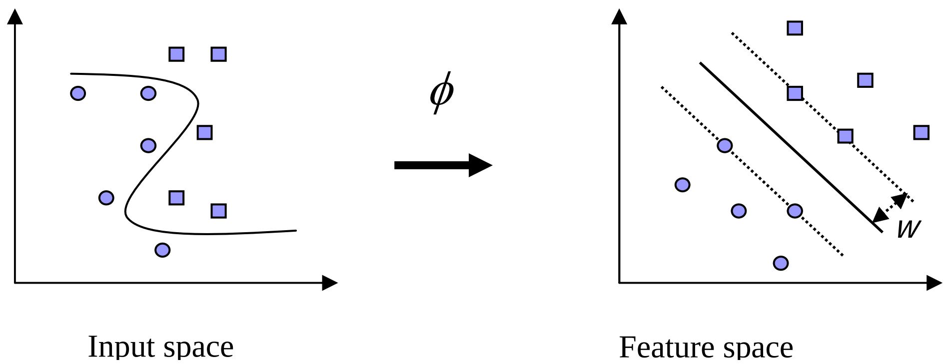

Figure 3 Mapping inputs on the multidimensional input space into high dimensional feature space

4.2 System Design and Methodology

In this section, we present our system framework in which task matching is conducted dynamically. As soon as a task arrives, the decision of which machine will process the arrived task is made by the SVM.

4.2.1 System and Workload Models

We consider a centralized heterogeneous distributed system in which a main scheduler is responsible for mapping tasks onto client machines. In this model, distributed machines are connected to a single server machine via high-speed network, and the server dispatches heterogeneous independent tasks, which arrive at Poisson arrival rate.

Job arrival time is represented by an exponential random variable with a mean of $1/\lambda$. The system design is shown in Figure 4. On task arrival, the Support Vector Scheduler (SVS) sends to the SVM the input vectors which are encoded with information about the ready time of each machine. The ready time changes at every task mapping. The SVM servers as an evaluator for the input vectors from SVS. Furthermore, the SVS dispatches incoming tasks onto the suitable machines based on the result of the evaluation.

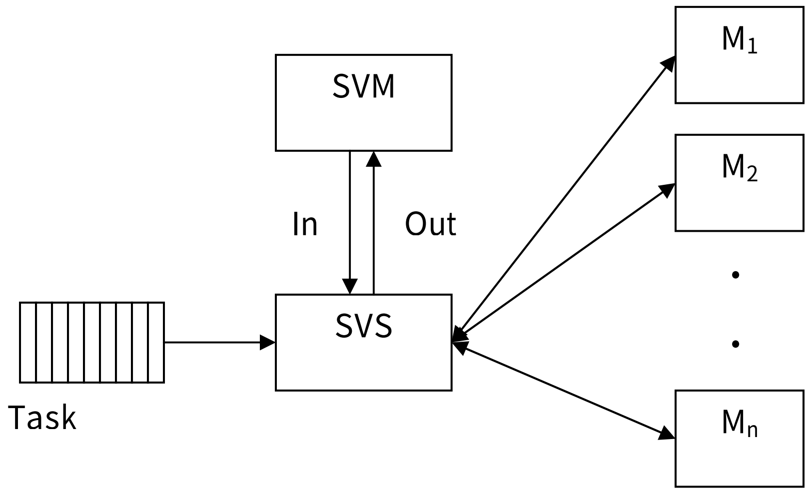

Figure 4 Task Scheduling System Framework

4.2.2 SVS (Support Vector Scheduler)

The process of constructing the scheduler is described in Figure. 5. The SV learner analyzes the training data and creates the SVM model. Then, the SV classifier constructs a decision function from the SVM model. Using the decision function, SV classifier evaluates an input vector from SVS. SVS conducts a task mapping, communicating with the SV classifier.

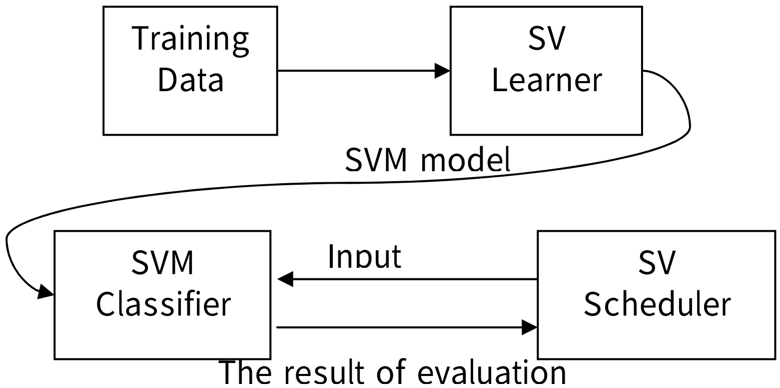

Figure 5 Support Vector Scheduler

4.2.3 Generating Training Data

The training data consist of a processor's computing power, its ready time, and its label. Every label of training instances is either positive or negative. We generate the training data using our Training Data Simulator (TDS). TDS is a set of programmed Excel sheets. It simulates the makespan using Excel sheets in which a set of computing power, ready time, and task is created randomly. The label of training instances is determined by the makespan.

4.2.4 SV Learning

Many real-world problems may not be separable linearly in multi-dimensional input space. In the case of the non-linear problem, we use a non-linear classifier for SV learning. One critical process of a non-linear classifier is to map the training data into high-dimensional feature space via the non-linear mapping function Φ, create a non-linear boundary at the same time, and construct maximal margin hyperplane in the feature space.

If we use a Kernel function, we can conduct non-linear mapping without explicitly coordinating input vectors in feature space. This is the reason why we call it a computational shortcut. A sequence of processes of finding the optimal hyperplane is depicted in Figure 6. After the non-linear mapping function transforms the input vectors in multi-dimensional input space into high dimensional feature space via non-linear mapping function, we find the optimal hyperplane through the maximal margin optimization process. Ultimately, SV learning is the process of finding the support vectors which come to lie on the non-linear boundary.

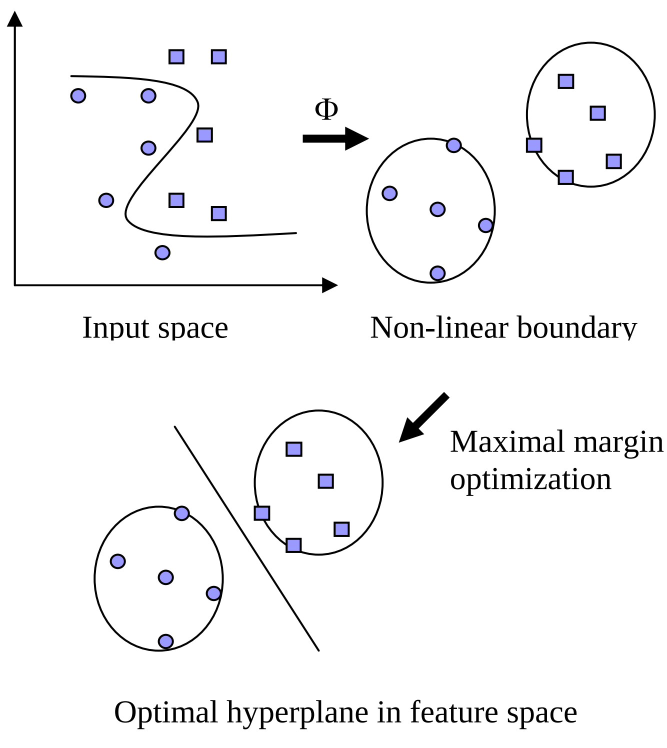

Figure 6 Non-linear mapping by Radial Basis Function (RBF) Kernel

4.2.5 Task Matching Onto Non-identical Machines

SVS dispatches incoming tasks onto non-identical machines by the evaluation result of the SV classifier. For a new task arriving, SVS generates input vectors based on the ready time of each machine. The number of input vector is determined by the number of machines. That is, SVS should generate the same number of input vectors as machines. By pre-assigning a task into each machine, we can create a corresponding input vector for each mapping. The information of the ready time for each mapping is incorporated into input vectors. Figure 7 shows that the SVM evaluates the input vectors in feature space.

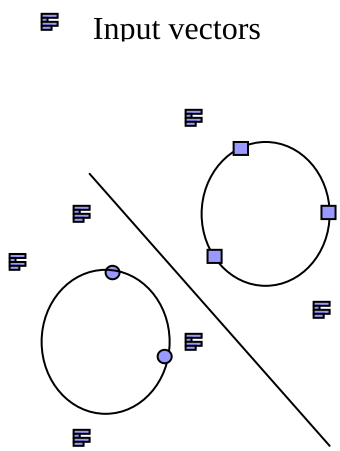

Figure 7 Evaluating Input Vectors

The decision of which machine to run a ready task is made as the result of evaluating input vectors. The machine that has the best evaluation runs the ready task.

5 EXPERIMENT & RESULT ANALYSIS

5.1 Experiment Procedure

Task matching onto non-identical machines is simulated using our Java simulator and the task arrivals are modeled by a Poisson distribution process. The simulator implements Support Vector Scheduler (SVS) and three heuristic algorithms (EF,RR,LL). Heterogeneous independent tasks are simulated by generating random numbers to represent an instruction number of the meta-task. We created seven task sets, each of which has different task size, 100, 200, 500, 1000, 2000, 5000, and 10,000 respectively. We created three processor sets, the processor number of which is 4, 8, and 16, respectively.

The processor is also simulated by generating a random number to represent its computing power. We created 30 different computing power sets. They are classified into 2 groups based on the range of computing power. 15 out of 30 computing power sets ranges from 0 to 100 and other sets range from 0 to 1000. Thus, the experiment is classified into Experiment 1 and Experiment 2 according to the computing power sets. Each experiment is conducted on 7 different task sets and 3 different Processor sets. After conducting each experiment on our SVS and three heuristic algorithms, we average the results of each experiment separately. Next, the result of SVS will be compared with three heuristic algorithms.

In the following, our SVS and three heuristic algorithms are explained briefly. The Earliest First (EF) algorithm assigns a task to the machine that will finish it earliest. Its complexity is $O(m)$ in the worst case, where m is the number of machines. The Lightest Load (LL) algorithm assigns a task preferentially to the machine with the lightest load. Its complexity is also $O(m)$. The Round Robin (RR) algorithm assigns a task in a round robin manner, with complexity $O(1)$. The SVS, evaluating the input vectors corresponding to each machine, has a complexity of $O(m)$.

5.2 Results & Analysis

The experimental evaluation of the heuristics is performed only in immediate mode. SVS is compared with 3 immediate mode heuristics. The immediate mode heuristics consider only one task when they try to reduce the total completion time, and the schedule cannot change, once decided. The average makespan of four algorithms will be plotted. In Figure 8, each point corresponds to the average makespan of each algorithm for different task sizes. From Figure 8 and 9, the average makespan of all algorithms gradually increases, as the task size grows. However, it can be noted that the degree of increase is much different according to the algorithm. Obviously, the shape of graph in SVS and EF changes slightly compared with LL and RR. Notably, LL undergoes a drastic deterioration of the performance in largest task size. In fact, the performance of SVS is very close to EF as shown in Figure 8. In Figure 9, the performance of all algorithms appears to be extremely similar to that in Figure 8, except the performance of LL declined distinctively from Experiment 1 of Figure 8. In the experiment with 4 processors, SVS and EF outperform LL and RR in all task sizes.

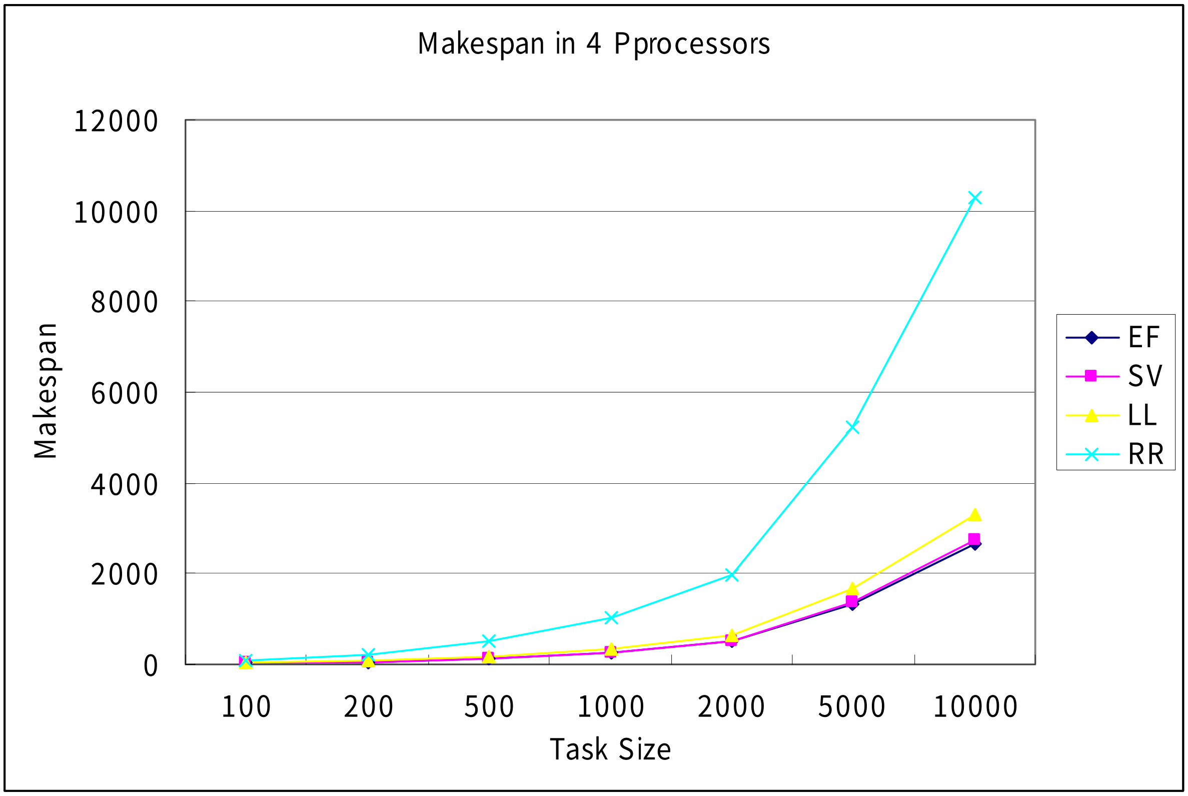

Figure 8 Makespan by task size in 4 processors (Experiment 1)

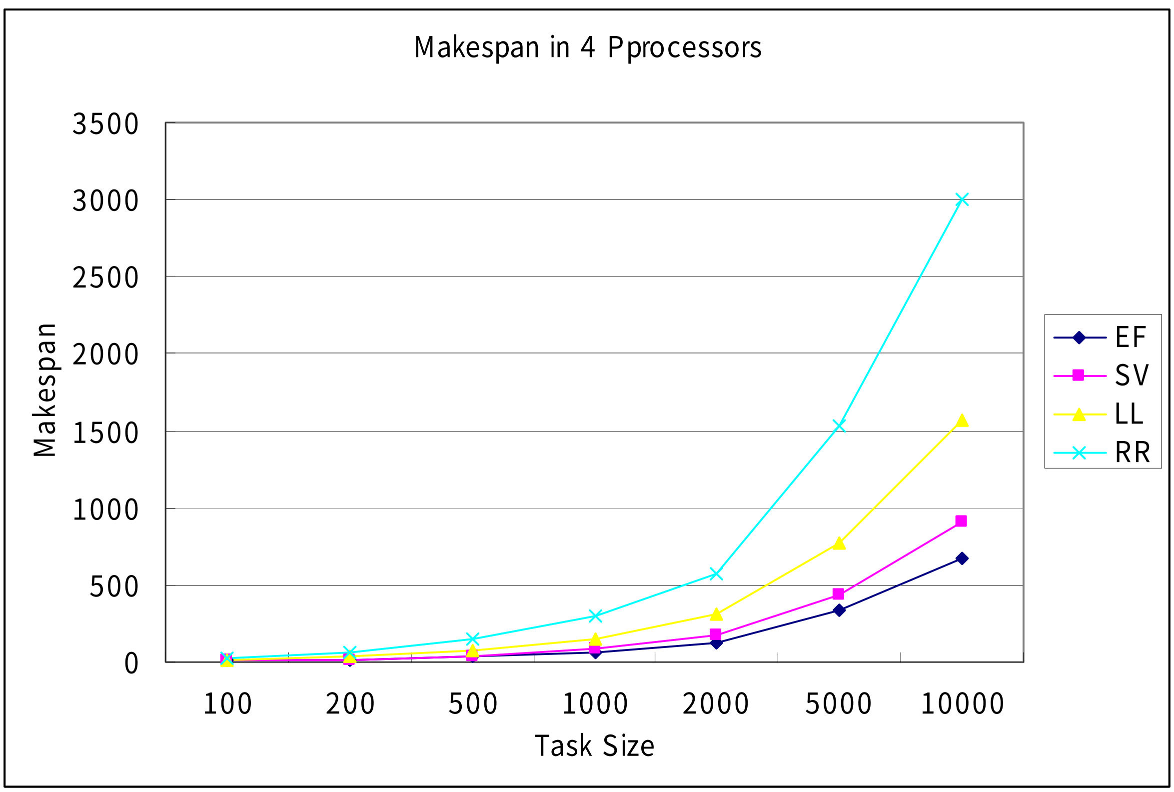

Figure 9 Makespan by task size in 4 processors (Experiment 2)

Figures 10 and 11 show the result from the experiment with 8 processors. Surprisingly, the SVS outperforms EF slightly in the largest task size for the first time as shown in Figure 10. However, the result reverses again in another experiment with different computing power sets (Figure 11). It should be noted significantly in Figure 11 is that the performance of LL drops so dramatically that RR outperforms LL.

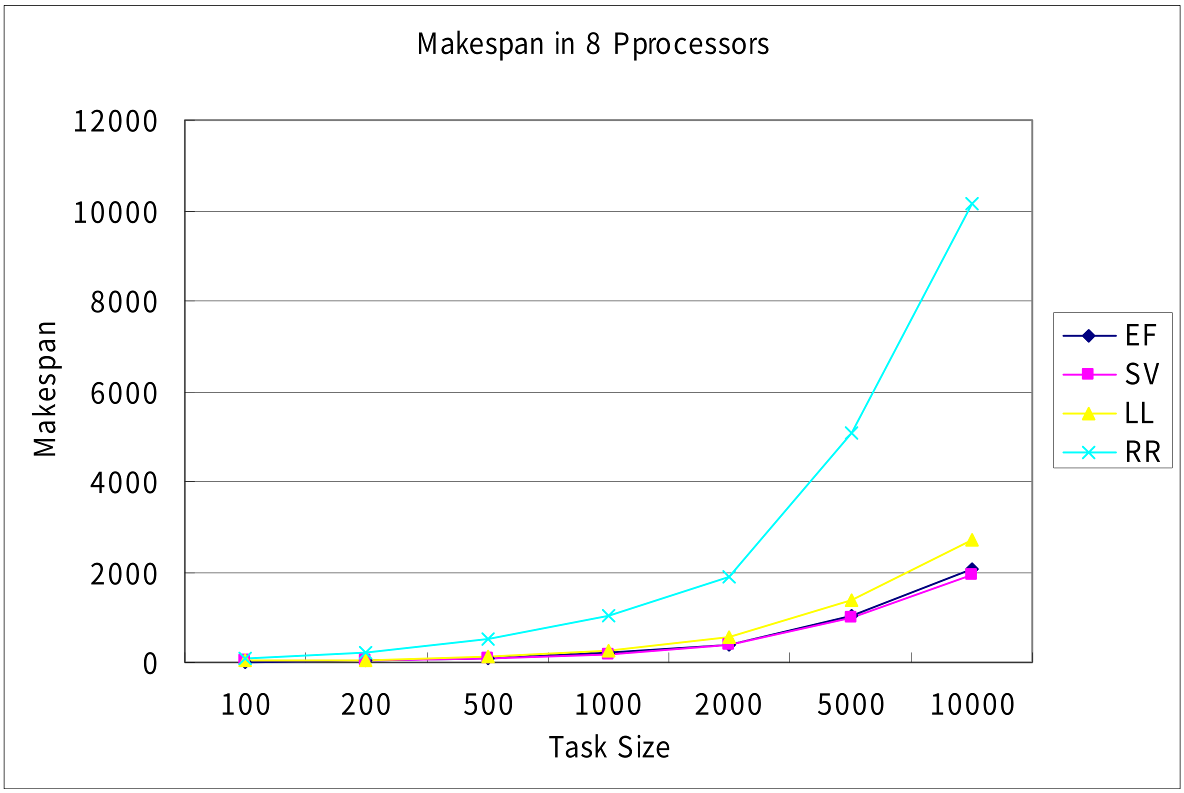

Figure 10 Makespan by task size in 8 processors (Experiment 1)

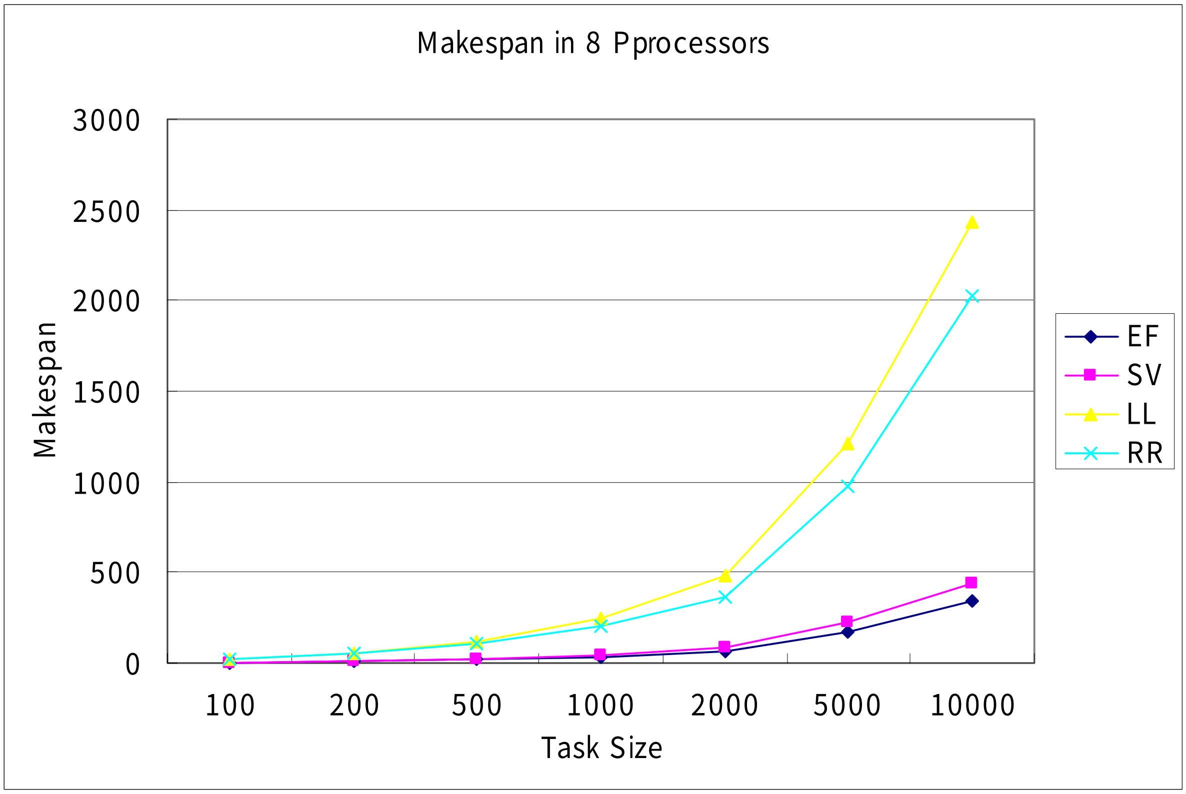

Figure 11 Makespan by task size in 8 processors (Experiment 2)

Fig. 12 and 13 show the result of 16 processors. From Figures 10, 11, 12, and 13, we can tell that the performance in 8 and 16 processors is extremely similar to each other.

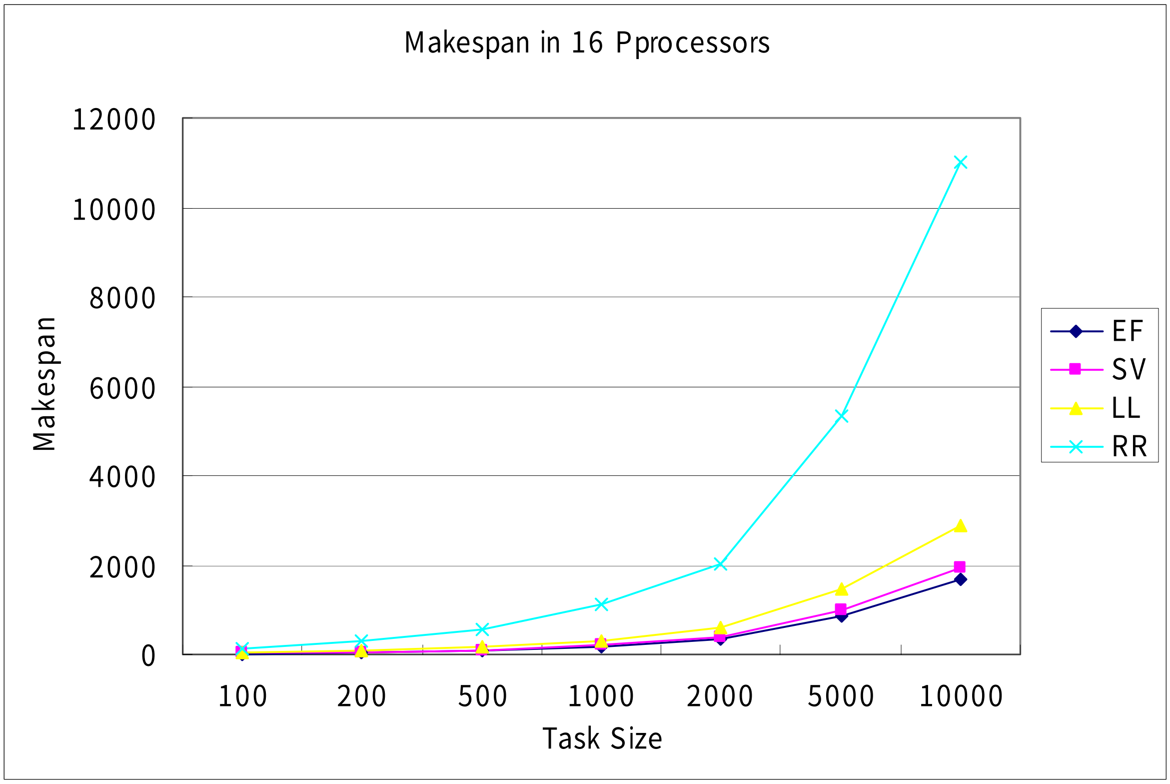

Figure 12 Makespan by task size in 16 processors (Experiment 1)

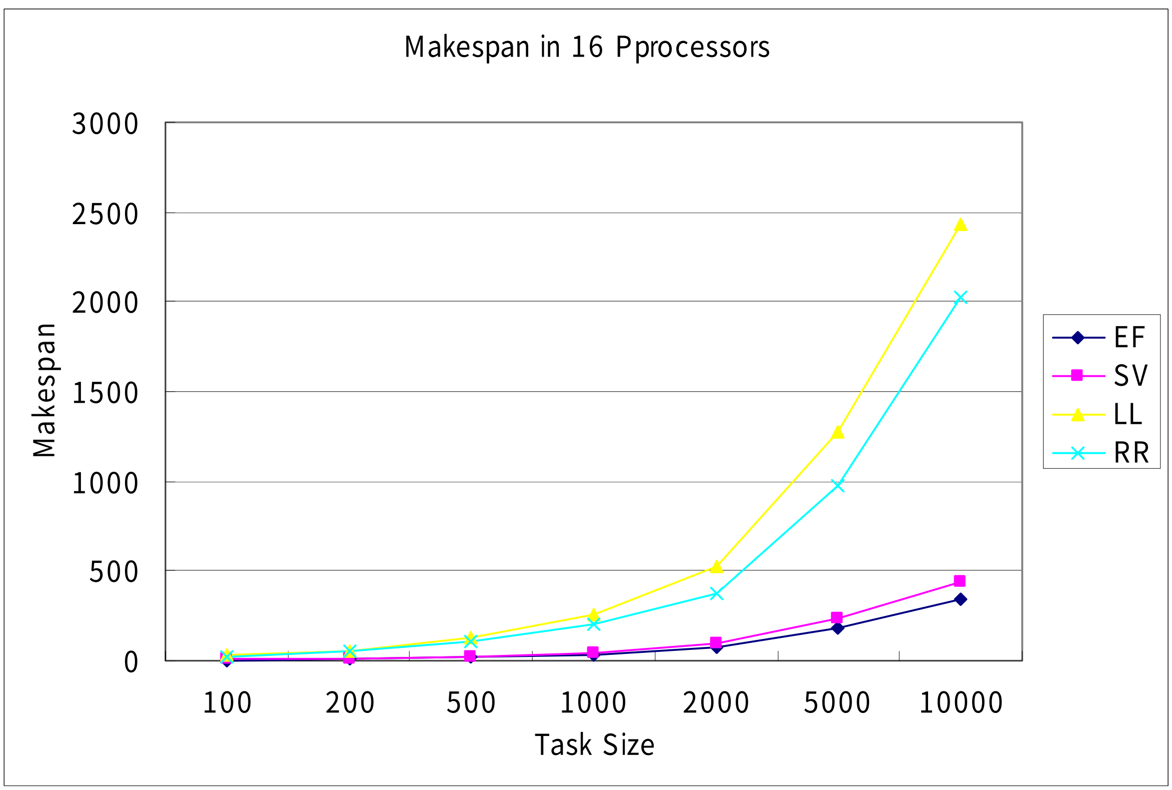

Figure 13 Makespan by task size in 16 processors (Experiment 2)

On the whole, the experiment results show that EF and SVS outperform LL and RR in all task sizes. The performance of SVS is close to EF overall. Further, it can be noted that the number of processors does not have an impact on the performance, but varying computing power sets affect the performance of LL and RR. Lastly, we can infer that the total system performance is affected more by its organization of computing power than by the number of processors.

6 CONCLUSION

In this thesis, we used a novel machine learning technology to solve the task matching problem of mapping a class of independent tasks onto the suitable machines. Using the Support Vector Machine (SVM), we analyzed the workload patterns of the total system in which a workload of each machine changes constantly when the machine consumes tasks. By learning the mapping between the pattern and the corresponding makespan, the SVM is able to map incoming tasks to appropriate machines. We trained the SVM using the data that our Training Data Simulator (TDS) created, and constructed a decision model to process unknown input vectors. Our Support Vector Scheduler (SVS) and three conventional heuristics for mapping a class of independent tasks onto non-identical machines were compared under a variety of simulated environments. Using simulations we compared our algorithm against Early Fast (EF), Light Least (LL), and Round Robin (RR). Results show that SVS gives a very close performance to EF in all processor sets and computing power sets. However, SVM is superior since it can dynamically adept to changing inputs and machine characteristics.

In this research, we used 10,000 samples, which were randomly generated, to construct a support vector model. The learning capability of the SVM entirely depends on the samples. In future works, we will study the relation between chosen samples and their corresponding performances.

REFERENCES

[Khok93] A. A. Khokhar, V. K. Prasanna, M. E. Shaaban, and C. L. Wang, "Heterogeneous computing: challenges and opportunities," *Computer*, vol. 26, pp. 18-27, 1993.

[Brau01] T. D. Braun, H. J. Siegel, N. Beck, L. L. Bölöni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, B. Yao, and D. Hensgen, "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," *Journal of Parallel and Distributed Computing*, vol. 61, pp. 810-837, 2001.

[Mahe99] M. Maheswaran, S. Ali, H. J. Siegel, D. A. Hensgen, and R. F. Freund, "Dynamic mapping of a class of independent tasks onto heterogeneous computing systems," *Journal of Parallel and Distributed Computing*, vol. 59, pp. 107-131, 1999.

[Meht06] A. M. Mehta, J. Smith, H. J. Siegel, A. A. Maciejewski, A. Jayaseelan, and B. Ye, "Dynamic resource management heuristics for minimizing makespan while maintaining an acceptable level of robustness in an uncertain environment," *Proceedings of the 12th International Conference on Parallel and Distributed Systems, Volume 1*, pp. 107-114, 2006.

[Page05] A. J. Page and T. J. Naughton, "Dynamic task scheduling using genetic algorithms for heterogeneous distributed computing," *Proceedings of 19th IEEE*

International Parallel and Distributed Processing Symposium, pp. 189a-189a, 2005.

[Fuji03] N. Fujimoto and K. Hagihara, "Near-optimal dynamic task scheduling of independent coarse-grained tasks onto a computational grid," *Proceedings of 2003 International Conference on Parallel Processing*, pp. 391-398, 2003.

[Cho94] S. Y. Cho and K. H. Park, "Dynamic task assignment in heterogeneous linear array networks for metacomputing," *Proceedings of the Heterogeneous Computing Workshop*, vol. 94, pp. 66-71, 1994.

[Hong04] B. Hong and V. K. Prasanna, "Distributed adaptive task allocation in heterogeneous computing environments to maximize throughput," *Proceedings of 18th International Parallel and Distributed Processing Symposium*, 2004.

[Mahe98] M. Maheswaran and H. J. Siegel, "A dynamic matching and scheduling algorithm for heterogeneous computing systems," *Proceedings of the Seventh Heterogeneous Computing Workshop*, pp. 57, 1998.

[Poje02] C. Po-Jen and W. Chia-Hsin, "An efficient optimization technique for task matching and scheduling in heterogeneous computing systems," *Proceedings of Parallel and Distributed Systems,* 2002.

[Min97] T. Min, H. J. Siegel, J. K. Antonio, and Y. A. Li, "Minimizing the application execution time through scheduling of subtasks and communication traffic in a heterogeneous computing system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 8, pp. 857, 1997.

[Brau98] T. D. Braun, H. J. Siegel, N. Beck, L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, and M. D. Theys, "A taxonomy for describing matching and scheduling heuristics formixed-machine heterogeneous computing systems," *Proceedings of Seventeenth IEEE Symposium on Reliable Distributed Systems*, pp. 330-335, 1998.

[Freu98] R. F. Freund, M. Gherrity, S. Ambrosius, M. Campbell, M. Halderman, D. Hensgen, E. Keith, T. Kidd, M. Kussow, and J. D. Lima, "Scheduling resources in multi-user, heterogeneous, computingenvironments with SmartNet," *Proceedings of 1998 Seventh Heterogeneous Computing Workshop*, pp. 184-199, 1998.

[Kara02] H. D. Karatza and R. C. Hilzer, "Load sharing in heterogeneous distributed systems," *Proceedings of the 2002 Winter Simulation Conference, 2*002.

[Scho99] B. Scholkopf, C. J. C. Burges, and A. J. Smola, "Advances in kernel methods: Support Vector Learning," MIT Press, Cambridge, MA, 1999.

[Burg98] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.

[Gers04] K. Gersmann and B. Hammer, "A reinforcement learning algorithm to improve scheduling search heuristics with the svm," *Proceedings of 2004 IEEE International Joint Conference on Neural Networks*, vol. 3, 2004.

[YiHu05] L. Yi-Hung, H. Han-Pang, and L. Yu-Sheng, "Dynamic scheduling of flexible manufacturing system using support vector machines," *Proceedings of the 2005 IEEE,* 2005.

[Arms98] R. Armstrong, D. Hensgen, and T. Kidd, "The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions," *7th IEEE Heterogeneous Computing Workshop (HCW98)*, vol. 5, 1998.

[Dong02] L. Dongmei, L. Yuanxiang, and Y. Mingzhao, "A genetic algorithm for task scheduling in network computing environment," *Proceedings of Algorithms and Architecture for Parallel Processing,* 2002.

[Vapn98] V. N. Vapnik, *Statistical Learning Theory*: Wiley New York, pp. 401-567, 1998.

APPENDIX

4 Processors of Experiment 1					
#set	Task size	Algorithm			
		EF	SV	LL	RR
set1	100.00	27.04	24.60	28.75	36.84
	200.00	51.96	47.01	52.09	79.56
	500.00	127.86	107.89	128.26	195.41
	1000.00	257.13	216.13	262.25	387.25
	2000.00	511.69	437.79	520.00	741.09
	5000.00	1298.90	1101.17	1326.34	1979.75
	10000.00	2612.99	2194.91	2666.72	3897.50
set2	100.00	12.48	14.79	21.82	34.68
	200.00	24.81	30.31	45.09	74.88
	500.00	58.24	65.20	106.15	183.91
	1000.00	118.18	136.76	221.38	364.47
	2000.00	241.45	268.63	446.29	697.50
	5000.00	613.26	720.83	1122.29	1863.29
	10000.00	1229.91	1431.49	2261.24	3668.24
set3	100.00	12.90	14.98	22.62	34.68
	200.00	24.56	30.01	47.06	74.88
	500.00	58.98	67.24	109.88	183.91
	1000.00	119.96	137.58	223.18	364.47
	2000.00	240.97	272.80	451.71	697.50
	5000.00	613.98	722.44	1118.68	1863.29
	10000.00	1224.65	1419.18	2254.88	3668.24
set4	100.00	27.04	24.60	28.75	36.84
	200.00	51.96	47.01	52.09	79.56
	500.00	127.86	107.89	128.26	195.41
	1000.00	257.13	216.13	262.25	387.25
	2000.00	511.69	437.79	520.00	741.09
	5000.00	1298.90	1101.17	1326.34	1979.75
	10000.00	2612.99	2194.91	2666.72	3897.50
set5	100.00	51.57	53.44	67.20	235.80
	200.00	99.76	110.30	116.60	509.20
	500.00	238.09	261.67	287.60	1250.60

	1000.00	482.57	529.96	555.40	2478.40
	2000.00	967.26	1058.48	1128.00	4743.00
	5000.00	2474.59	2726.22	2862.00	12670.40
	10000.00	4968.60	5478.07	5723.80	24944.00
set6	100.00	25.53	22.63	37.73	107.18
	200.00	48.77	44.33	68.00	231.45
	500.00	118.51	108.06	157.73	568.45
	1000.00	240.89	209.24	315.09	1126.55
	2000.00	485.61	453.82	634.45	2155.91
	5000.00	1229.94	1110.00	1605.00	5759.27
	10000.00	2460.76	2280.09	3181.45	11338.18
set7	100.00	39.12	39.19	50.50	196.50
	200.00	74.89	80.81	94.00	424.33
	500.00	178.92	188.43	220.17	1042.17
	1000.00	362.24	388.83	451.50	2065.33
	2000.00	724.79	775.57	909.50	3952.50
	5000.00	1851.65	1986.74	2308.50	10558.67
	10000.00	3718.91	3989.30	4610.67	20786.67
set8	100.00	39.76	42.17	50.40	147.38
	200.00	75.10	82.48	93.13	318.25
	500.00	179.81	188.25	228.88	781.63
	1000.00	363.71	395.15	451.75	1549.00
	2000.00	729.87	779.15	911.50	2964.38
	5000.00	1865.53	1996.96	2313.50	7919.00
	10000.00	3747.22	4008.75	4592.13	15590.00
set9	100.00	41.62	46.93	55.88	147.38
	200.00	80.58	92.15	96.30	318.25
	500.00	191.44	215.11	234.50	781.63
	1000.00	387.31	442.07	487.50	1549.00
	2000.00	776.74	871.59	964.13	2964.38
	5000.00	1983.93	2259.52	2441.75	7919.00
	10000.00	3983.63	4499.59	4876.63	15590.00
set10	100.00	26.04	26.67	25.00	26.20
	200.00	50.08	50.52	50.13	58.69
	500.00	123.99	115.33	116.33	138.96
	1000.00	255.16	235.75	242.40	275.38
	2000.00	499.37	473.32	481.27	527.00
	5000.00	1293.40	1229.62	1218.09	1407.82
	10000.00	2572.88	2439.92	2450.29	2799.18
set11	100.00	18.81	19.77	18.75	16.19

	200.00	38.14	38.47	32.48	34.30
	500.00	90.41	94.03	80.42	81.29
	1000.00	179.73	184.24	156.73	160.94
	2000.00	367.34	376.09	324.77	319.64
	5000.00	951.72	932.19	796.49	822.75
	10000.00	1860.47	1862.80	1597.17	1635.88
set12	100.00	27.47	25.36	34.50	84.21
	200.00	50.91	49.58	66.07	181.86
	500.00	129.51	118.14	156.21	446.64
	1000.00	253.49	236.42	311.50	885.14
	2000.00	510.03	470.12	635.93	1693.93
	5000.00	1309.23	1207.27	1613.07	4525.14
	10000.00	2622.35	2411.16	3217.14	8908.57
set13	100.00	31.63	32.43	34.81	56.14
	200.00	62.11	61.33	71.62	121.24
	500.00	149.23	147.87	167.86	297.76
	1000.00	297.58	296.76	340.57	590.10
	2000.00	598.15	594.67	682.57	1129.29
	5000.00	1495.35	1523.41	1736.38	3016.76
	10000.00	2996.72	3065.78	3460.33	5939.05
set14	100.00	22.48	23.36	34.00	65.50
	200.00	44.21	42.04	55.28	141.44
	500.00	105.39	103.17	142.11	347.39
	1000.00	209.98	199.76	287.89	688.44
	2000.00	416.72	407.11	569.56	1317.50
	5000.00	1091.06	1029.69	1439.11	3519.56
	10000.00	2188.89	2089.83	2865.39	6928.89
set15	100.00	26.76	24.59	48.20	235.80
	200.00	53.68	49.57	75.20	509.20
	500.00	129.86	116.78	173.60	1250.60
	1000.00	264.29	235.60	367.40	2478.40
	2000.00	528.13	504.80	714.20	4743.00
	5000.00	1332.22	1269.40	1818.00	12670.40
	10000.00	2699.59	2612.80	3618.40	24944.00

Task size	The Average of 4 Processors in Experiment 1			
	EF	SV	LL	RR
100	28.682559	29.033	37.260415	97.421
200	55.435881	57.061	67.676111	210.47
500	133.87278	133.67	162.53061	516.38
1000	269.9573	270.69	329.11915	1023.3
2000	540.65441	545.45	659.5912	1959.2
5000	1380.2431	1394.4	1669.7036	5231.7
10000	2766.7034	2798.6	3336.1965	10302

4 Processors of Experiment 2					
Set#	Task size	Algorithm			
		EF	SV	LL	RR
set1	100	12.899563	14.98333	22.61765	34.67647
	200	24.558952	30.00833	47.05882	74.88235
	500	58.978166	67.24167	109.8824	183.9118
	1000	119.9607	137.575	223.1765	364.4706
	2000	240.96507	272.8	451.7059	697.5
	5000	613.97817	722.4417	1118.676	1863.294
	10000	1224.6463	1419.183	2254.882	3668.235
set2	100	4.9915074	5.521127	46.4	235.8
	200	9.5711253	9.785915	67.4	509.2
	500	22.135881	24.84789	150	1250.6
	1000	45.600849	49.10704	297.6	2478.4
	2000	89.386412	96.28169	580.8	4743
	5000	233.2569	340.6	1458.8	12670.4
	10000	471.0276	1066.6	2929	24944
set3	100	8.1327801	8.716157	18.10204	12.03061
	200	15.93361	16.18672	33.55102	25.97959
	500	37.524017	40.67686	72.21429	63.80612
	1000	76.286307	83.41485	149.8265	126.449
	2000	152.46473	159.3493	300.4796	241.9898
	5000	390.0262	414.6201	741.5612	646.449
	10000	776.88797	827.0699	1525.745	1272.653
set4	100	6.5392562	8.246377	13.07377	9.663934
	200	13.301653	16.04831	27.22131	20.86885
	500	32.169421	40.31884	59.07377	51.2541
	1000	66.43595	79.92271	118.8689	101.5738
	2000	133.91116	160.57	244.0902	194.3852

	5000	343.32645	423.7729	613.7295	519.2787
	10000	685.46901	837.2754	1240.254	1022.295
set5	100	6.0583554	8.842324	17.7931	13.55172
	200	12.421751	14.14938	33.24138	29.26437
	500	31.599469	34.08299	75.62069	71.87356
	1000	60.824934	72.52282	143.6552	142.4368
	2000	124.10345	144.2365	304.4598	272.5862
	5000	311.11141	369.1079	768.4598	728.1839
	10000	628.11141	724.1992	1534.437	1433.563
set6	100	9.0304569	9.418182	31	98.25
	200	17.507614	34.5	56.5	212.1667
	500	42.13198	107.0625	143.6667	521.0833
	1000	85.162437	220.9375	294.5	1032.667
	2000	171.20812	453.5	577.4167	1976.25
	5000	434.78934	1150.813	1442.917	5279.333
	10000	875.72843	2340.875	2893.5	10393.33
set7	100	4.0789474	5.171053	5.982332	4.166078
	200	8.3717172	9.644737	11.73498	8.996466
	500	19.945455	23.69474	25.34982	22.09541
	1000	41.262626	48.24474	52.79859	43.78799
	2000	80.624242	95.32368	103.6042	83.79859
	5000	210.18947	247.6947	264.0283	223.8587
	10000	413.69091	488.9737	539.636	440.7067
set8	100	5.0635359	5.795518	5.931973	4.010204
	200	9.6823204	10.60504	10.70748	8.659864
	500	21.008403	24.8232	23.55442	21.26871
	1000	43.237569	50.84314	49.80612	42.14966
	2000	86.914365	100.5798	101.6259	80.66327
	5000	220.86188	259.5994	256.4558	215.483
	10000	436.12431	521.2157	509.5	424.2177
set9	100	6.2092555	8.728395	16.65672	17.59701
	200	12.820926	17.2716	36.34328	38
	500	29	50.80247	88.16418	93.32836
	1000	59.072435	91.11111	159.0746	184.9552
	2000	116.50704	208.321	334.5672	353.9552
	5000	297.86117	521.9136	838.1343	945.5522
	10000	604.52515	1102.321	1689.045	1861.493
set10	100	6.0241228	6.085271	18.53731	17.59701
	200	11.778509	15.33333	36.80597	38
	500	27.776316	36.6124	84.29851	93.32836

	1000	56.723684	71.48837	164.4925	184.9552
	2000	111.26535	155.3953	339.3284	353.9552
	5000	282.20175	400.3953	836.806	945.5522
	10000	573.80702	836.3023	1714.791	1861.493
set11	100	4.524173	5.318066	9.379888	6.586592
	200	9.311828	10.00509	17.44134	14.22346
	500	22.690323	22.80662	40.09497	34.93296
	1000	45.498925	46.64122	82.74302	69.22905
	2000	90.529032	94.99237	164.0503	132.486
	5000	230.14839	245.4758	423.3128	353.9218
	10000	461.18925	487.6031	849.7709	696.7598
set12	100	10.205607	12.50327	16.75532	12.54255
	200	19.523364	24.32026	29.84043	27.08511
	500	49.28972	55.95425	76.28723	66.52128
	1000	99.538941	114.9346	143.7766	131.8298
	2000	198.26791	226.6928	298	252.2872
	5000	508.83178	603.6536	742.8511	673.9574
	10000	1012.0156	1178.412	1498.574	1326.809
set13	100	6.3839662	6.268293	14.89916	9.907563
	200	11.49789	12.47154	24.95798	21.47154
	500	28.812236	32.45528	59.86179	52.54622
	1000	55.472574	62.73984	128.7563	104.1345
	2000	111.93671	133.4797	250.1513	199.2857
	5000	282.8038	330.2195	639.8908	532.3697
	10000	575.04008	696.7805	1267.815	1048.067
set14	100	7.4501279	7.477612	21.97727	26.79545
	200	12.936061	16.49254	41.88636	57.86364
	500	33.12532	39.88806	103.4773	142.1136
	1000	64.222506	81.50746	207.2955	281.6364
	2000	126.6266	181.2687	408.75	538.9773
	5000	326.93862	451.1791	1005.205	1439.818
	10000	656.68286	945.7313	2068.455	2834.545
set15	100	7.924581	8.009091	23.42424	35.72727
	200	14.122905	20.51818	46.93939	77.15152
	500	34.100559	50.28182	113.6061	189.4848
	1000	70.329609	107.4727	226.2121	375.5152
	2000	138.84358	223.4091	440.697	718.6364
	5000	359.03352	560.2273	1133.97	1919.758
	10000	714.71229	1196.964	2291.03	3779.394

Task size	The Average of 4 Processors in Experiment 2			
	EF	SV	LL	RR
100.00	7.03	8.07	18.84	35.93
200.00	13.56	17.16	34.78	77.59
500.00	32.69	43.44	81.68	190.54
1000.00	65.98	87.90	162.84	377.61
2000.00	131.57	180.41	326.65	722.65
5000.00	336.36	469.45	818.99	1930.48
10000.00	673.98	977.97	1653.76	3800.50

The 8 Processors of Experiment 1					
Set#	Task size	Algorithm			
		EF	SV	LL	RR
set 1	100.00	5.04	4.73	21.09	14.14
	200.00	9.22	9.70	46.26	29.98
	500.00	22.28	23.23	105.35	71.84
	1000.00	44.83	46.57	204.56	145.02
	2000.00	87.58	93.10	405.09	266.42
	5000.00	228.22	241.44	1021.05	718.74
	10000.00	453.45	478.41	2056.44	1434.67
set 2	100.00	4.77	5.32	21.02	14.14
	200.00	9.15	9.59	42.67	29.98
	500.00	21.53	23.16	100.77	71.84
	1000.00	44.74	46.96	204.77	145.02
	2000.00	88.05	93.80	403.19	266.42
	5000.00	229.34	240.72	1028.79	718.74
	10000.00	451.99	476.01	2060.58	1434.67
set 3	100.00	40.49	49.13	57.00	304.00
	200.00	73.98	91.96	114.50	644.50
	500.00	174.24	217.87	225.00	1544.50
	1000.00	347.91	438.06	433.50	3118.00
	2000.00	696.23	870.57	868.00	5728.00
	5000.00	1775.72	2203.19	2135.50	15453.00
	10000.00	3564.72	4379.57	4279.50	30845.50
	100.00	22.37	23.94	27.75	38.00
	200.00	45.88	40.86	53.25	80.56
	500.00	109.17	98.99	128.80	193.06
	1000.00	219.19	201.36	259.63	389.75

	2000.00	435.84	397.66	521.13	716.00
	5000.00	1109.10	1006.09	1301.63	1931.63
	10000.00	2214.90	2070.94	2601.75	3855.69
set 5	100.00	26.91	24.66	55.00	608.00
	200.00	52.86	44.43	100.00	1289.00
	500.00	122.93	104.82	172.00	3089.00
	1000.00	251.66	215.84	324.00	6236.00
	2000.00	502.43	447.08	667.00	11456.00
	5000.00	1286.71	1138.88	1562.00	30906.00
	10000.00	2566.87	2311.40	3140.00	61691.00
set 6	100.00	20.16	18.58	29.40	60.80
	200.00	34.96	37.45	57.40	128.90
	500.00	80.74	88.68	140.50	308.90
	1000.00	163.04	176.06	275.20	623.60
	2000.00	334.65	346.22	547.10	1145.60
	5000.00	854.32	909.71	1368.50	3090.60
	10000.00	1679.65	1784.46	2752.70	6169.10
set 7	100.00	24.84	27.59	28.85	38.00
	200.00	50.80	48.98	52.25	80.56
	500.00	119.59	113.06	125.31	193.06
	1000.00	253.05	228.98	257.38	389.75
	2000.00	500.34	451.10	519.63	716.00
	5000.00	1265.66	1138.16	1307.00	1931.63
	10000.00	2568.81	2267.18	2597.63	3855.69
set 8	100.00	19.99	19.77	29.05	30.40
	200.00	38.01	38.01	49.95	64.45
	500.00	93.74	86.16	128.20	154.45
	1000.00	185.33	178.47	252.20	311.80
	2000.00	371.35	346.76	503.35	572.80
	5000.00	961.65	882.69	1263.90	1545.30
	10000.00	1921.20	1778.29	2537.60	3084.55
set 9	100.00	20.00	23.35	30.50	60.80
	200.00	43.29	41.85	58.40	128.90
	500.00	98.74	96.91	136.60	308.90
	1000.00	193.08	194.09	276.60	623.60
	2000.00	391.39	388.03	547.80	1145.60
	5000.00	999.53	966.00	1381.80	3090.60
	10000.00	1991.48	1952.58	2749.20	6169.10
set 10	100.00	26.54	24.52	27.76	28.95
	200.00	50.33	43.78	48.38	61.38

	500.00	123.11	99.07	123.95	147.10
	1000.00	246.67	201.53	245.10	296.95
	2000.00	495.56	411.08	502.86	545.52
	5000.00	1257.31	1051.57	1251.67	1471.71
	10000.00	2512.82	2088.37	2495.95	2937.67
set 11	100.00	23.33	22.80	28.47	35.76
	200.00	45.95	40.23	53.59	75.82
	500.00	111.58	97.43	131.06	181.71
	1000.00	232.55	192.08	262.24	366.82
	2000.00	468.15	386.45	513.29	673.88
	5000.00	1178.31	983.60	1295.94	1818.00
	10000.00	2372.79	1955.21	2581.94	3628.88
set 12	100.00	19.59	19.25	37.29	86.86
	200.00	40.57	39.97	69.86	184.14
	500.00	94.54	88.58	140.86	441.29
	1000.00	189.00	182.05	290.57	890.86
	2000.00	381.59	365.08	568.57	1636.57
	5000.00	990.15	922.34	1424.86	4415.14
	10000.00	1961.03	1840.63	2829.57	8813.00
set 13	100.00	19.91	20.65	32.25	76.00
	200.00	39.60	39.68	57.58	161.13
	500.00	96.51	94.53	140.38	386.13
	1000.00	193.28	184.53	280.88	779.50
	2000.00	397.25	370.85	565.88	1432.00
	5000.00	1016.81	936.16	1400.00	3863.25
	10000.00	2036.88	1862.82	2815.13	7711.38
set 14	100.00	18.39	19.87	23.68	19.35
	200.00	36.06	36.84	49.50	37.91
	500.00	83.52	89.10	112.44	90.85
	1000.00	170.61	182.56	228.53	183.41
	2000.00	347.52	363.55	444.62	336.94
	5000.00	871.99	913.51	1125.03	909.00
	10000.00	1710.65	1853.08	2251.56	1814.44
set 15	100.00	24.50	22.29	31.33	67.56
	200.00	50.23	41.56	57.86	143.22
	500.00	116.31	97.91	138.67	343.22
	1000.00	236.17	201.59	280.22	692.89
	2000.00	480.81	394.82	565.00	1272.89
	5000.00	1214.52	1002.80	1385.56	3434.00
	10000.00	2421.73	2019.44	2786.22	6854.56

Task Size	The Average of 8 Processors in Experiment 1			
	EF	SV	LL	RR
100.00	21.12	21.76	32.03	98.85
200.00	41.39	40.32	60.76	209.36
500.00	97.90	94.63	136.66	501.72
1000.00	198.07	191.38	271.69	1012.87
2000.00	398.58	381.74	542.83	1860.71
5000.00	1015.96	969.12	1350.21	5019.82
10000.00	2028.60	1941.23	2702.38	10019.99

The 8 Processors of Experiment 2					
Set#	Task size	Algorithm			
		EF	SV	LL	RR
set 1	100.00	4.77	5.32	21.02	14.14
	200.00	9.15	9.59	42.67	29.98
	500.00	21.53	23.16	100.77	71.84
	1000.00	44.74	46.96	204.77	145.02
	2000.00	88.05	93.80	403.19	266.42
	5000.00	229.34	240.72	1028.79	718.74
	10000.00	451.99	476.01	2060.58	1434.67
set 2	100.00	3.92	4.04	50.67	202.67
	200.00	7.95	8.52	84.33	429.67
	500.00	18.39	19.24	149.33	1029.67
	1000.00	37.34	39.26	302.33	2078.67
	2000.00	74.54	81.05	594.67	3818.67
	5000.00	188.02	204.58	1507.33	10302.00
	10000.00	375.68	413.57	2963.00	20563.67
set 3	100.00	6.99	7.80	25.84	24.32
	200.00	13.68	16.57	53.92	51.56
	500.00	33.24	40.06	120.56	123.56
	1000.00	66.65	76.11	250.60	249.44
	2000.00	134.73	153.01	486.52	458.24
	5000.00	345.75	395.99	1216.36	1236.24
	10000.00	686.94	792.08	2402.72	2467.64
set 4	100.00	4.49	4.77	31.00	33.78
	200.00	8.62	9.09	55.44	71.61
	500.00	20.81	22.64	124.56	171.61

	1000.00	42.00	44.76	262.39	346.44
	2000.00	83.16	88.66	513.33	636.44
	5000.00	215.48	234.21	1284.94	1717.00
	10000.00	428.89	460.99	2568.78	3427.28
set 5	100.00	4.67	4.20	23.24	17.88
	200.00	8.65	8.50	47.47	37.91
	500.00	21.56	21.20	110.38	90.85
	1000.00	41.39	42.90	217.15	183.41
	2000.00	84.36	83.49	440.91	336.94
	5000.00	213.21	223.84	1108.18	909.00
	10000.00	427.52	434.91	2254.15	1814.44
set 6	100.00	4.32	4.88	37.67	101.33
	200.00	8.87	9.42	66.50	214.83
	500.00	21.20	22.77	148.83	514.83
	1000.00	43.21	46.92	290.17	1039.33
	2000.00	83.89	92.90	575.00	1909.33
	5000.00	218.02	240.83	1438.83	5151.00
	10000.00	434.67	473.42	2873.17	10281.83
set 7	100.00	4.88	5.23	34.44	67.56
	200.00	10.11	9.98	58.89	143.22
	500.00	23.04	24.13	137.67	343.22
	1000.00	46.81	48.07	281.44	692.89
	2000.00	94.35	95.88	563.67	1272.89
	5000.00	243.27	250.65	1391.33	3434.00
	10000.00	484.18	494.27	2799.56	6854.56
set 8	100.00	4.00	4.50	22.47	15.59
	200.00	7.50	8.13	43.79	33.05
	500.00	19.14	19.29	101.98	79.21
	1000.00	37.64	39.81	208.79	159.90
	2000.00	74.67	76.63	415.95	293.74
	5000.00	187.35	202.50	1071.15	792.46
	10000.00	376.74	399.18	2103.23	1581.82
set 9	100.00	5.19	4.78	37.29	86.86
	200.00	9.75	8.94	60.86	184.14
	500.00	23.76	21.76	148.57	441.29
	1000.00	46.34	45.65	297.71	890.86
	2000.00	95.98	91.03	571.14	1636.57
	5000.00	241.47	233.87	1433.43	4415.14
	10000.00	484.31	460.69	2832.57	8813.00
set 10	100.00	4.54	4.56	11.47	4.54

	200.00	8.76	10.02	24.42	9.62
	500.00	21.03	22.23	53.45	23.05
	1000.00	42.92	46.65	112.51	46.54
	2000.00	86.80	90.66	219.57	85.49
	5000.00	221.89	237.09	551.88	230.64
	10000.00	441.66	479.61	1128.03	460.38
set 11	100.00	4.33	4.36	22.10	15.57
	200.00	8.28	8.06	45.37	31.44
	500.00	18.93	18.90	108.24	75.34
	1000.00	37.08	39.03	206.41	152.10
	2000.00	74.53	78.10	423.95	279.41
	5000.00	190.98	198.70	1047.07	753.80
	10000.00	376.72	400.22	2123.61	1504.66
set 12	100.00	4.47	4.29	19.73	11.69
	200.00	7.85	8.59	39.73	24.79
	500.00	19.53	20.71	95.02	59.40
	1000.00	38.91	42.15	192.08	119.92
	2000.00	77.39	83.12	389.08	220.31
	5000.00	200.06	210.16	985.40	594.35
	10000.00	395.92	424.68	1943.35	1186.37
set 13	100.00	6.61	8.31	17.22	9.30
	200.00	13.05	15.83	37.09	19.53
	500.00	31.88	36.65	83.09	46.80
	1000.00	65.12	74.95	161.14	94.48
	2000.00	132.42	145.97	333.56	173.58
	5000.00	338.34	388.86	847.86	468.27
	10000.00	673.89	780.80	1719.56	934.71
set 14	100.00	5.53	6.11	30.05	40.53
	200.00	11.14	11.40	57.27	85.93
	500.00	27.50	28.46	127.93	205.93
	1000.00	54.80	59.36	261.00	415.73
	2000.00	109.19	117.87	529.73	763.73
	5000.00	280.57	300.51	1326.13	2060.40
	10000.00	558.19	599.05	2614.93	4112.73
set 15	100.00	3.81	4.06	21.04	12.67
	200.00	7.67	8.00	41.48	26.85
	500.00	16.95	19.12	96.96	64.35
	1000.00	35.49	37.65	194.38	129.92
	2000.00	70.47	74.61	398.63	238.67
	5000.00	182.41	195.43	1002.46	643.88

	10000.00	364.92	391.85	2028.60	1285.23

Task Size	The Average of 8 Processors in Experiment 2			
	EF	SV	LL	RR
100.00	4.84	5.15	27.02	43.89
200.00	9.40	10.04	50.62	92.94
500.00	22.57	24.02	113.82	222.73
1000.00	45.36	48.68	229.52	449.64
2000.00	90.97	96.45	457.26	826.03
5000.00	233.08	250.53	1149.41	2228.46
10000.00	464.15	498.76	2294.39	4448.20

The 16 Processors of Experiment 1					
Set#	Task size	Algorithm			
		EF	SV	LL	RR
set 1	100.00	3.57	4.81	34.27	33.00
	200.00	6.31	9.02	62.45	78.73
	500.00	15.33	19.38	144.36	146.09
	1000.00	30.60	38.97	278.91	287.73
	2000.00	60.94	75.91	552.18	512.00
	5000.00	158.99	192.49	1366.09	1400.73
	10000.00	313.71	383.73	2736.45	2878.00
set 2	100.00	18.67	26.06	37.50	61.60
	200.00	35.07	48.99	68.17	173.20
	500.00	82.26	106.96	152.17	321.40
	1000.00	166.58	218.59	294.33	633.00
	2000.00	333.80	411.87	587.80	1124.20
	5000.00	854.54	1031.78	1444.80	2986.20
	10000.00	1712.52	2011.21	2901.20	6183.80
set 3	100.00	19.15	26.38	38.50	61.60
	200.00	35.97	49.68	71.67	173.20
	500.00	83.28	107.82	151.20	321.40
	1000.00	162.95	212.90	292.33	633.00
	2000.00	337.48	416.66	582.00	1124.20

	5000.00	865.59	1028.56	1441.20	2986.20
	10000.00	1701.99	2000.66	2908.80	6183.80
set 4	100.00	18.11	26.37	58.00	154.00
	200.00	32.48	48.01	96.00	433.00
	500.00	77.44	108.07	177.00	803.50
	1000.00	164.09	204.36	331.50	1582.50
	2000.00	317.51	395.38	618.50	2810.50
	5000.00	834.57	1013.31	1492.00	7465.50
	10000.00	1629.55	2019.03	3021.50	15459.50
set 5	100.00	21.22	29.83	34.91	38.50
	200.00	37.92	56.22	63.50	108.25
	500.00	91.67	121.70	146.38	200.88
	1000.00	183.68	233.79	281.13	395.63
	2000.00	378.17	445.96	558.25	702.63
	5000.00	951.59	1100.08	1399.00	1866.38
	10000.00	1875.61	2173.19	2811.00	3864.88
set 6	100.00	17.11	26.17	50.67	102.67
	200.00	35.57	48.23	76.33	288.67
	500.00	80.55	108.57	168.33	535.67
	1000.00	159.61	205.04	300.33	1055.00
	2000.00	336.67	393.71	592.00	1873.67
	5000.00	846.66	979.34	1493.33	4977.00
	10000.00	1685.74	1958.38	2954.00	10306.33
set 7	100.00	18.34	26.20	55.00	308.00
	200.00	35.77	48.48	100.00	866.00
	500.00	80.13	106.00	171.00	1607.00
	1000.00	171.40	208.06	340.00	3165.00
	2000.00	338.30	399.03	686.00	5621.00
	5000.00	851.01	990.90	1561.00	14931.00
	10000.00	1691.71	1965.99	3026.00	30919.00
set 8	100.00	20.95	27.82	55.00	308.00
	200.00	35.50	48.36	100.00	866.00
	500.00	81.50	108.19	184.00	1607.00
	1000.00	170.99	206.79	341.00	3165.00
	2000.00	339.69	401.33	607.00	5621.00
	5000.00	877.53	1001.53	1553.00	14931.00
	10000.00	1724.43	1969.60	3016.00	30919.00
set 9	100.00	19.78	27.61	51.00	154.00
	200.00	42.02	52.79	70.00	433.00
	500.00	97.58	119.29	162.00	803.50

	1000.00	190.39	217.99	318.50	1582.50
	2000.00	368.13	421.58	602.50	2810.50
	5000.00	979.13	1047.56	1527.50	7465.50
	10000.00	1937.08	2077.10	2996.50	15459.50
set 10	100.00	19.75	25.65	55.00	308.00
	200.00	35.45	49.47	100.00	866.00
	500.00	83.47	106.86	153.86	1607.00
	1000.00	164.55	211.99	359.00	3165.00
	2000.00	336.49	401.57	626.00	5621.00
	5000.00	843.28	964.05	1504.00	14931.00
	10000.00	1691.12	1911.82	3041.00	30919.00
set 11	100.00	17.46	24.16	31.10	30.80
	200.00	34.52	45.59	56.19	86.60
	500.00	77.85	103.72	135.40	160.70
	1000.00	157.22	199.82	278.70	316.50
	2000.00	310.08	383.79	551.60	562.10
	5000.00	797.34	967.18	1384.70	1493.10
	10000.00	1612.86	1926.21	2754.60	3091.90
set 12	100.00	19.66	28.85	34.80	36.30
	200.00	36.58	55.40	57.50	96.22
	500.00	85.94	119.04	140.22	178.56
	1000.00	179.57	230.14	281.00	351.67
	2000.00	362.22	423.80	558.33	624.56
	5000.00	919.28	1042.85	1398.67	1659.00
	10000.00	1851.19	2071.13	2791.00	3435.44
set 13	100.00	18.32	24.71	31.09	28.00
	200.00	33.28	47.92	56.91	78.73
	500.00	81.21	103.24	143.82	146.09
	1000.00	161.41	203.21	276.91	287.73
	2000.00	320.35	393.62	540.91	511.00
	5000.00	834.84	973.36	1377.18	1357.36
	10000.00	1629.37	1975.48	2742.00	2810.82
set 14	100.00	20.33	29.16	31.43	24.20
	200.00	39.07	55.94	52.71	61.86
	500.00	90.44	116.37	134.71	114.79
	1000.00	179.83	227.08	268.20	226.07
	2000.00	356.30	414.65	531.86	401.50
	5000.00	918.47	1009.31	1333.00	1066.50
	10000.00	1845.16	1956.59	2669.93	2208.50
set 15	100.00	20.84	32.54	38.75	77.00

	200.00	43.06	60.98	63.50	216.50
	500.00	96.91	135.67	146.25	401.75
	1000.00	202.05	248.77	299.50	791.25
	2000.00	414.64	482.44	585.25	1405.25
	5000.00	1016.51	1217.21	1472.50	3732.75
	10000.00	2039.60	2410.35	2925.75	7729.75

Task Size	The Average of 16 Processors in Experiment 1			
	EF	SV	LL	RR
100.00	18.22	25.76	42.47	115.04
200.00	34.57	48.34	73.00	321.73
500.00	80.37	106.06	154.05	597.02
1000.00	162.99	204.50	302.76	1175.84
2000.00	327.38	390.75	585.35	2088.34
5000.00	836.62	970.63	1449.86	5549.95
10000.00	1662.78	1920.70	2886.38	11491.28

The 16 Processors of Experiment 2					
Set#	Task size	Algorithm			
		EF	SV	LL	RR
set 1	100.00	3.36	4.93	33.82	33.00
	200.00	6.79	8.72	57.09	78.73
	500.00	15.27	20.52	141.82	146.09
	1000.00	30.77	37.62	273.00	287.73
	2000.00	61.63	76.39	553.64	512.00
	5000.00	156.80	192.88	1364.36	1400.73
	10000.00	314.33	383.15	2747.45	2878.00
set 2	100.00	3.96	5.21	19.98	6.06
	200.00	6.71	9.93	39.23	15.75
	500.00	16.15	21.52	90.20	29.22
	1000.00	32.56	41.06	189.85	57.55
	2000.00	64.61	86.69	371.18	102.20
	5000.00	165.50	235.59	945.89	271.47
	10000.00	328.39	482.47	1901.78	562.16
set 3	100.00	3.68	4.83	26.00	13.96
	200.00	6.83	9.15	51.44	34.64
	500.00	16.92	20.74	120.46	64.28

	1000.00	33.32	40.15	238.54	126.60
	2000.00	66.87	78.73	479.42	224.84
	5000.00	172.04	202.31	1219.16	597.24
	10000.00	338.00	409.25	2434.12	1236.76
set 4	100.00	3.85	5.17	23.81	11.00
	200.00	7.50	9.76	48.58	27.94
	500.00	18.70	24.65	116.06	51.84
	1000.00	37.69	51.94	226.87	102.10
	2000.00	73.74	100.89	467.58	181.32
	5000.00	188.55	254.92	1139.23	481.65
	10000.00	374.69	512.03	2292.68	997.39
set 5	100.00	3.84	4.86	26.25	19.25
	200.00	7.33	9.05	52.69	54.13
	500.00	16.61	21.12	125.19	100.44
	1000.00	33.03	41.61	260.19	197.81
	2000.00	67.68	83.73	515.81	351.31
	5000.00	171.47	213.76	1298.88	933.19
	10000.00	335.02	429.13	2621.19	1932.44
set 6	100.00	3.85	5.05	26.46	23.69
	200.00	6.74	9.42	59.77	66.62
	500.00	15.98	20.17	133.15	123.62
	1000.00	31.81	42.01	269.77	243.46
	2000.00	63.62	79.90	540.00	432.38
	5000.00	165.28	204.47	1344.46	1148.54
	10000.00	325.67	408.28	2686.77	2378.38
set 7	100.00	3.99	5.63	29.40	15.40
	200.00	7.72	10.45	51.45	43.30
	500.00	19.13	23.77	128.20	80.35
	1000.00	37.85	49.80	250.25	158.25
	2000.00	75.92	96.73	508.10	281.05
	5000.00	194.65	249.74	1263.60	746.55
	10000.00	383.18	495.11	2531.00	1545.95
set 8	100.00	3.35	4.65	30.08	25.67
	200.00	6.69	8.67	54.58	72.17
	500.00	16.51	19.97	137.08	133.92
	1000.00	31.65	38.13	275.58	263.75
	2000.00	63.80	77.37	542.25	468.42
	5000.00	162.58	198.76	1357.17	1244.25
	10000.00	324.46	395.93	2682.50	2576.58
set 9	100.00	4.40	5.65	37.00	61.60

	200.00	7.58	10.52	63.80	173.20
	500.00	18.55	24.36	143.80	321.40
	1000.00	37.20	46.51	289.20	633.00
	2000.00	76.68	91.80	589.20	1124.20
	5000.00	193.54	237.30	1450.00	2986.20
	10000.00	378.86	476.03	2895.00	6183.80
set 10	100.00	4.06	5.75	21.30	8.32
	200.00	8.50	10.85	45.89	23.41
	500.00	19.87	25.39	105.16	43.43
	1000.00	39.40	55.54	215.86	85.54
	2000.00	77.99	104.69	427.73	151.92
	5000.00	203.60	269.79	1064.54	403.54
	10000.00	406.32	538.59	2156.27	835.65
set 11	100.00	3.48	4.67	26.06	18.12
	200.00	6.42	8.65	54.41	50.94
	500.00	15.13	19.85	124.29	94.53
	1000.00	30.35	37.96	252.65	186.18
	2000.00	60.99	75.11	512.47	330.65
	5000.00	154.92	188.04	1284.82	878.29
	10000.00	309.30	376.32	2568.24	1818.76
set 12	100.00	3.38	4.87	18.80	6.16
	200.00	6.45	9.08	40.42	17.32
	500.00	16.20	20.48	104.02	32.14
	1000.00	30.46	38.52	201.32	63.30
	2000.00	62.40	74.85	404.00	112.42
	5000.00	159.43	192.48	1000.28	298.62
	10000.00	309.86	377.33	2016.94	618.38
set 13	100.00	3.61	4.98	22.15	9.06
	200.00	6.66	9.22	47.41	25.47
	500.00	15.76	19.71	112.65	47.26
	1000.00	31.99	38.86	214.44	93.09
	2000.00	66.81	76.96	436.06	165.32
	5000.00	166.75	195.71	1129.76	439.15
	10000.00	328.65	398.66	2254.03	909.38
set 14	100.00	3.68	5.46	30.54	23.69
	200.00	7.39	9.84	55.38	66.62
	500.00	17.05	21.82	133.08	123.62
	1000.00	34.76	44.28	263.54	243.46
	2000.00	68.45	83.70	534.77	432.38
	5000.00	177.31	214.00	1348.00	1148.54

	10000.00	357.05	425.98	2681.69	2378.38
set 15	100.00	3.88	4.90	14.71	3.85
	200.00	7.84	9.86	33.91	10.83
	500.00	17.62	22.26	77.31	20.09
	1000.00	35.19	44.06	157.49	39.56
	2000.00	70.82	84.88	311.86	70.26
	5000.00	182.89	222.51	796.38	186.64
	10000.00	364.88	441.21	1598.90	386.49

Task Size	The Average of 16 Processors in Experiment 2			
	EF	SV	LL	RR
100.00	3.76	5.11	25.76	18.59
200.00	7.14	9.54	50.40	50.74
500.00	17.03	21.76	119.50	94.15
1000.00	33.87	43.20	238.57	185.42
2000.00	68.13	84.83	479.60	329.38
5000.00	174.35	218.15	1200.44	877.64
10000.00	345.24	436.63	2404.57	1815.90