YAŞAR UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE

MASTER THESIS

WEB-BASED SOLUTION FOR SCHEDULING PROBLEM IN IDENTICAL PARALLEL MACHINES

Mehmet Emin BUDAK

Supervisor: Mehmet Fatih TAŞGETİREN

Industrial Management and Information Systems

Bornova, IZMIR

2014

YAŞAR UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE

MASTER THESIS

WEB-BASED SOLUTION FOR SCHEDULING PROBLEM IN IDENTICAL PARALLEL MACHINES

Mehmet Emin BUDAK

Supervisor: Prof. Dr. M. Fatih TAŞGETİREN

Industrial Management and Information Systems

Presentation Date: 24.10.2014

Bornova, IZMIR

2014

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

Prof. Dr. M. Fatih TAŞGETİREN (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

Assist. Prof. Dr. Ömer ÖZTÜRKOĞLU

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

Assist. Prof. Dr. Yücel ÖZTÜRKOĞLU

Prof. Dr. Behzat GÜRKAN

Director of the Graduate School

ABSTRACT

WEB-BASED SOLUTION FOR SCHEDULING PROBLEM IN IDENTICAL PARALLEL MACHINES

BUDAK, Mehmet Emin

MSc in Industrial Management and Information Systems
Supervisor: Prof. Dr. M. Fatih TAŞGETİREN

May, 2014

In this thesis, the parallel machine scheduling problem with n number of independent jobs assigned to m number of identical parallel machines to minimize the makespan is studied.

Some algorithms were used that are developed for this type of problem. VND and ILS algorithms were used, and some modifications were made to VND algorithm. The new algorithm was used with this software. These algorithms are used in software which has user interaction with its graphical user interface, animations and user defined variables.

This web-based software is developed with PHP, HTML, JAVASCRIPT and CSS programming languages. In this way it can run with any mobile device or computer with independent operating systems.

This software can be used in scheduling education because it helps generate schedules interactively with easy understanding animations step by step. In addition, we wanted to test the CPU time performance of the web-based scheduling software. Experimental results showed that CPU time requirement of a web-based scheduling system is computationally very expensive.

Keywords: identical parallel machine scheduling, iterated local search, variable neighborhood search, web-based scheduling.

ÖZET

ÖZDEŞ PARALEL MAKİNELERDE ÇİZELGELEME PROBLEMİ İÇİN WEB TABANLI ÇÖZÜM

BUDAK, Mehmet Emin

Yüksek Lisans Tezi, Endüstriyel Yönetim ve Bilişim Sistemleri Bölümü
Tez Danışmanı: Prof. Dr. Mehmet Fatih TAŞGETİREN
Mayıs, 2014

Bu tezde n tane birbirinden bağımsız işin, m tane özdeş paralel makinaya atanarak çizelge uzunluğunu en aza indirmek amacıyla çizelgeleme problemi üzerinde çalışmıştır.

Bu problem için üretilmiş olan bazı algoritmalardan bu tezde VND ve ILS algoritmaları kullanılmıştır ayrıca VND algoritmasının üzerinde değişiklik yapılarak modifiye edilmiş hali de kullanılmıştır. Bu algoritmalar grafiksel kullanıcı arayüzü, işlem animasyonları ve değişkenlerin kullanıcı tarafından belirlendiği bütünleşik bir yazılım parçaları olarak kullanılmıştır.

PHP, HTML, JAVASCRIPT ve CSS programlama dilleri kullanılarak web tabanlı bir yazılım olarak geliştirilmiştir. Bu sayede herhangi bir işletim sisteminde mobil cihazlarda veya bilgisayarlarda çalışabilmektedir.

Bu yazılım, kullanıcı etkileşimli kolay anlaşılır animasyonları ve adım adım çözüm yapmasından dolayı çizelgeleme eğitimlerinde kullanılabilir. Ek olarak web tabanlı çizelgeleme problemlerinde CPU süre performansını test etmek istedik. Deneysel sonuçlar, web tabanlı bir çizelgeleme sisteminde CPU süresinin fazla maliyetli olduğunu gösterdi.

Anahtar Sözcükler: özdeş paralel makina çizelgeleme, iteratif local arama, değişken komşuluk arama, web-tabanlı çizelgeleme.

ACKNOWLEDGEMENTS

I would like to thank to Prof. Dr. M. Fatih Taşgetiren. His advices on issues and experiences of programming allowed me to progress quickly. I learnt a lot of things from him during thesis writing.

I would like to thanks to my colleagues in the office. They helped me a lot on office jobs when I am studying on the thesis.

TEXT OF OATH

I declare and honestly confirm that my study titled "Web-Based Solution for Scheduling Problem in Identical Parallel Machines", and presented as Master's Thesis has been written without applying to any assistance inconsistent with scientific ethics and traditions and all sources I have benefited from are listed in bibliography and I have benefited from these sources by means of making references.

27 / 05 / 2014

Mehmet Emin BUDAK

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

INDEX OF SYMBOLS AND ABBREVIATION

Symbols	**Explanations**
C_{max}	Makespan
r	Uniform random number
$R//C_{max}$	Uniform Parallel Machine Scheduling Problems with a goal of finding the shortest completion time

Abbreviations

VNS	Variable Neighborhood Search
VND	Variable Neighborhood Descent
ILS	Iterated Local Search

CHAPTER 1

INTRODUCTION

Timing and cost, is the basic philosophy of scheduling problems. It is important to get the job done in minimum time. Scheduling is used for many fields and it is important process for production, manufacture, management, and computer science. Good schedules reduce time and increase product quality. Objectives of scheduling are minimization of the completion time and however may be finishing every job before delivery date. Due to the nature of scheduling problems many assumptions are made. Some of these assumptions, there is no interruption or no postponement, each machine cannot do more than one job, there is no machine distortion, machine number usually less then job number. In the parallel machine scheduling problem is assigning n independent jobs to m parallel machines. Parallel machines scheduling environment is given in Figure 1.

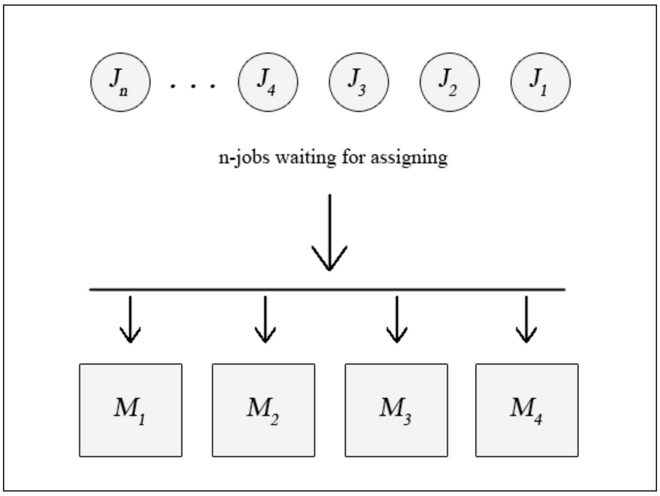


Figure 1. Four Parallel Machines with n-jobs

Web-based software developed for solving parallel machine problem. There is no too much web-based solution for scheduling problems. A Web-based application uses server resources for solving problems. It refers to any program that is accessed over a network connection using HTTP. These applications often run inside a web browser on mobile or computer. Web-based applications are also known as Web apps. User can reach software from any device and everywhere without installing any extra application. The software is easy to understand with clear layout, style, format and with schedule animations. This can be useful for scheduling in industry and education with animations.

Our objective in designing and implementing a web-based solution parallel machine scheduling is to emphasize the power of web technologies for the experimental design and output analysis phases of a schedueling study. Issues of portability, maintaining ability, and conformance to standards are crucial in demonstrating the feasibility of a web-based scheduling system.

Modern factories are becoming increasingly agile. The components of a manufacturing system (e.g. production, purchasing, design, and management) are integrated today to facilitate rapid and frequent changes in products and processes. To succeed in this dynamic environment, new systems must develop. Another objective of this study is to see the CPU time performance of the web-based scheduling systems.

This software can be used in scheduling education because it helps generate schedules interactively with easy understanding animations step by step. In addition, we wanted to test the CPU time performance of the web-based scheduling software.

The remaing part of the thesis is organized as follows: Chapter 2 gives the literature review on parallel machine scheduling problem. Chapter 3 outlines the problem definition. Chapter 4 explains the proposed algorithms. Chapter 5 explains the web-based scheduling software. Computational results are given in Chapter 6. Finally, Chapter 7 makes the conclusions.

CHAPTER 2

LITERATURE REVIEW

In this study and web-based software considered about identical parallel machine scheduling problem and many studies have been made about parallel machine scheduling in literature.

(McNaughton, 1959) was the first to study parallel machine scheduling problem in the literature. Then, (Graham L. , 1969) used some priority rules such as shortest processing time (SPT) and longest processing time (LPT). When scheduling the parallel machines, n jobs are scheduled on m identical machines to minimize some performance measures like total weighted tardiness and total flow time or maximum completion time. For the single machine scheduling with total tardiness criterion, it was shown by (Du & Leung, 1990) that the problem is NP-hard problem. For this reason, parallel machine scheduling problem on identical parallel machines is also strongly NP-hard problem. Different variants of parallel machine scheduling problems can be found in (Root, 1965), (Lawler, 1977), (Elmagraby & Park, 1974), (Dessouky, 1998). Due to the NP-Hardness nature of the problem, many researchers focused on developing heuristic methods. The most popular one is the List Scheduling method in which, the jobs are sorted using a rule and based on this rule, they are assigned to the machines according to their earliest time to finish. Theseheuristic methods are examined by (Wilkerson & Irwin, 1971), (Dogramaci & Surkis, 1979), (Ho & Chang, 1991), (Koulamas C. P., 1994). Furthermore, a decomposition heuristic and hybrid simulated annealing heuristic are presented by (Koulamas C. , 1997). For the parallel machine scheduling problem with the total tardiness criterion, a genetic algorithm is presented by (Bean, 1994). For minimization of the total completion time for the parallel machine flowshop scheduling problem, a tabu search is proposed by (Nowicki & Smutnicki, 1998). Tabu search and simulated annealing algorithms are also presented and compared by (Park & Kim, 1997). A hybrid metaheuristic algorithm is developed by (Anghinolfi & Paolucci, 2007). Another tabu search is also proposed by (Bilge, Kyrac, Kurtulan, & Pekgun, 2004).

Regarding the exact methods, the most known and exact solution algorithm of this problem in the parallel machine systems was proposed by (Pessoa, Uchoa, Aragao, & Rodrigues, 2008). This algorithm can find solution to the problems up to 50 jobs.

CHAPTER 3

PROBLEM DEFINITION

3.1 Problem Formulation

In the parallel machine scheduling problem there are n independent jobs to be processed on m parallel machines. Each job can be processed by only one machine and a machine can process only one job at a time. When a job is started to be processed on a machine, it has to be finished until completion. There is a set N of n jobs $(j = 1,2,\dots,n)$ and a set M of m jobs $(i = 1,2,\dots,m)$. The processing time of each job is known in advance and denoted as p_j. For the identical parallel machine case, in each machine, the processing speed is different for the same job, and it is denoted as s_i. For this reason, the processing time of a job j on machine i can be established as $p_{ij} = p_j/s_i$. In general, processing time of each job depends on the machine and this is referred to as the unrelated parallel machine scheduling problem. In parallel machine scheduling problems, the commonly employed objective is to minimize the maximum completion time C_{max}. The $\alpha/\beta/\gamma$ secheduling problems classification scheme is proposed initially by (Graham, Lawler, Lenstra, & Rinnooy, 1979).

The $R//C_{max}$ problem is simply an assignment problem, since the processing sequence of the jobs assigned to a machine does not make a change the maximum completion time at that machine. It is obvious that there are m^n possible solutions to the problem after all possible assignments. For this reason, the $R//C_{max}$ problem is shown to be NP-hard by (Garey & Johnson, 1979). In addition, the two machine version $P2//C_{max}$ is proven to be NP-hard by (Lenstra, Rinnooy, & Brucker, 1977). Furthermore, no polynomial time algorithm does exist for the general $R//C_{max}$ problem with a better worst case ratio approximation than $3/2$ unless $P = NP$, according to (Lenstra, J. K.; Shmoys, D. B.; Tardos, E., 1990). The Mixed Integer Linear Programming (MILP) formulation for the $R//C_{max}$ is shown below:

$$\min C_{max}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall j \in N,$$

$$\sum_{j=1}^{n} p_{ij} x_{ij} \leq C_{max} \quad \forall i \in M,$$

$$x_{ij} \in \{0,1\} \quad \forall j \in N, \qquad \forall i \in M.$$

In the formulation, the first constraint provides that one job can be processed by only one machine. Second constraint provides that the total processing times of assigned jobs on their machines must be smaller than the maximum completion time, for each machine. The decision variable x_{ij} is a binary variable;

$$x_{ij} = \begin{cases} 1, & if\ job\ j\ is\ assigned\ to\ machine\ i \\ 0, & otherwise \end{cases}$$

To solve the problem described above, we propose a variable neighborhood descent algorithm (VND) and an iterated local search (ILS) algorithm in this thesis. Their details are given in subsequent sections.

CHAPTER 4

PROPOSED ALGORITHMS

4.1 Variable Neighborhood Search

Variable neighborhood search (VNS) is a metaheuristic for solving combinatorial and global optimization problems. Basic idea is systematic change of neighborhood within a local search. Local search methods for combinatorial optimization proceed by performing a sequence of local changes in an initial solution which improve each time the value of the objective function until a local optimum is found. It is proposed by (Mladenovic & Hansen, 1997).

The algorithm involves iterative exploration of larger and larger neighborhoods for a given local optima until there is an improvement, after which time the search is repeated. It is aimed for solving linear program problems, integer program problems, mixed integer program problems, nonlinear program problems, etc.

The strategy of metaheuristic is to guide the search process. Its goal is to explore the solution space to find a new optimal solution than the current one. Metaheuristic algorithms are approximate algorithms and most of the time they are non-deterministic. Metaheuristic methods are not specific for one problem.

Algorithm includes two types of moves. These are insert and swap moves. These moves affects the completion times of the jobs by changing the location of job in the sequence.

Most local search metaheuristics use just few neighbourhoods (one or two, number of neighbourhood) at each iteration, which could be changed from one iteration to another. Changing the neighbourhood structure during the search makes the search process more effective. Therefore, if there is more than one neighbourhood at each observed solution, that will help to improve the solution process to explore the search space and thus find new candidate solutions, fulfilling the basic idea of VNS.

Initialization: Select a set of neighborhood structures Nk(k $= 1, \ldots, k_{max}$), find an initial solution π, set $k = 1$, Until $k = k_{max}$, repeat the following steps Step 1 (shaking): Generate $\pi' \in Nk$ at random Step 2 (local search): apply some local search method with π' as the initial solution; denote with π'' the obtained local optimum Step 3 (move or not): if the solution thus obtained is better than the incumbent, move there ($\pi = \pi''$), and continue the search with ($k = 1$); otherwise, set $k = k + 1$; go back to Step 1. General outline of the VNS algorithm is given in Figure 2.

$\boldsymbol{VNS}()$

$\pi = GenerateInitialSolution$

$k_{max} = 2$

$k = 1$

$do\{$

$\pi_1 = shake(\pi)$

$\pi_2 = N_k(\pi_1)$ % $N_1(\pi_i) = BestInsert(\pi_i)$

$if\ f(\pi_2) < f(\pi)\ then$ % $N_2(\pi_i) = BestSwap(\pi_i)$

$\pi = \pi_1$

$k = 1$

$else$

$k = k + 1$

$\}while\ (k \leq k_{max})$

$Return\ \pi$

Figure 2. VNS Algorithm

An initial solution of the problem is one of the inputs to the main step of the algorithm. This initial solution can be generated randomly or by a construction heuristic if it is expected to positively affect the main step so that better solutions can be obtained in shorter time.

The main step can possibly be iterated until maximum number of iterations, maximum CPU time allowed or maximum number of iterations. The algorithm should run long enough to create near optimal solutions but at the same time it should not continue unnecessarily without making any improvements. Often successive neighborhoods Nk will be nested. Note that point π' is generated at random in Step 2 in order to avoid cycling, which might occur if any deterministic rule was used.

Unlike the VNS local search described above, we employ the deterministic variant called Variable Neihborhood Descent (VND) similar to (Tasgetiren, Liang, Sevkli, & Gencyilmaz, 2007), two different VND local searches are developed in this study. The first one denoted as VND 1 is given in Figure 3.

```
VND1()
    π = GenerateInitialSolution
    k_max = 2
    for i = 1 to n * n{
        k = 1
        do{
            π_1 = N_k(π)                    % N_1(π_i) = BestInsert(π_i)
            if f(π_1) < f(π_i) then          % N_2(π_i) = BestSwap(π_i)
                π_i = π_1
                k = 1
            else
                k = k + 1
        }while (k ≤ k_max)
    }Endfor
Return π_i
```

Figure 3. VND 1 Algorithm

VND 1 algorithm applies the BestInsert function firstly. The BestInsert function removes a job from a solution and inserts it in all posible position in the partial solution and retains the best one. The size of the insert moves is $(n-1)^2$. If this new best solution improves, search continues. Otherwise, the BestSwap function is applied where two jobs are selected randomly and they are exchanged. The size of the exchange move is $n(n-1)/2$.

The choice of the first neighborhood affects the solution quality. In the VND1 algorithm, we take the InsertBest neighborhood as the first neighborhood and the swapBest neighborhood as the second neighborhood. By changing their sequence, we generate another VND algorithm denoted as VND 2. The VND 2 local search is given in Figure 4.

```
VND2()
    π = GenerateInitialSolution
    k_max = 2
    for i = 1 to n * n{
        k = 1
        do{
            π_1 = N_k(π)                  % N_1(π_i) = BestSwap(π_i)
            if f(π_1) < f(π_i) then        % N_2(π_i) = BestInsert(π_i)
                π_i = π_1
                k = 1
            else
                k = k + 1
        }while (k ≤ k_max)
    }Endfor
Return π_i
```

Figure 4. VND 2 Algorithm

The BestInsert and BestSwap procedures are also given in Figure 5 and 6.

```
Procedure N_1(π)
i = 1
while(i ≤ n)do {
    π_1 = remove job π_i from π
    π_2 = the best permutation obtained by inserting job π_i in
          any possible position of π_1
    if(f(π_2) < f(π))then {
            π = π_2
    } else {
            i = i + 1
    }endif
}endwhile
return π
endprocedure
```

Figure 5. $\boldsymbol{N_1(\pi)}$ Neighborhood (BestInsert)

$\boldsymbol{Procedure\ N_2(\pi)}$
$for\ i = 1\ to\ n - 1\ \{$
$\quad for\ j = i + 1\ to\ n\ \{$
$\quad\quad \pi_1 = the\ best\ permutation\ obtained\ by\ swapping\ job\ \pi_i\ and\ ob\ \pi_j\ in\ \pi$
$\quad\quad if\big(f(\pi_1) < f(\pi)\big)then\ \{$
$\quad\quad\quad \pi = \pi_1$
$\quad\quad \}endif$
$\quad \}endfor$
$\}endfor$
$return\ \pi$
$endprocedure$

Figure 6. $\boldsymbol{N_2(\pi)}$ Neighborhood (BestSwap)

4.2 Iterated Local Search

Iterated Iterated local Search (ILS) is a simple stochastic local search method. It iteratively applies local search to perturbations of the current solution, which leads to a random walk in the space of local optima (Lourenc, H. R., Martin, O. & Stützle, T. ,2002). In an ILS algorithm, there are four procedures: *GenerateInitialSolution* constructs the initial solution, *Perturbation* generates new solution for the local search by perturbing the solution, the *AcceptanceCriterion* decides if the new solution will be replaced with the incumbed one, the *LocalSearch* procedure search for the solution space. The history component in Perturbation and AcceptanceCriterion shows that the search history may affect the acceptance decisions made in these procedures. As an alternative, Markovian implementations of ILS can be applied. It means that the output of Perturbation and AcceptanceCriterion is independent of the search history. General outline of the ILS algorithm is given in Figure 7.

procedure *Iterated Local Search*
s_0 ← GenerateInitialSolution
s ← LocalSearch(s_0)
repeat
s' ← Perturbation(s, history)
s'' ← LocalSearch(s′)
s ← AcceptanceCriterion(s, s'' ,history)
until termination condition met
end *Iterated Local Search*

Figure 7. General Outline of Iterated Local Search

Iterated Local Search is based on a simple, but successful idea. Instead of simply repeating local search starting from an initial solution like restart approaches do, ILS optimizes solution *s* with local search, perturbates the local optimal solution *s'* and applies local search again. This procedure is repeated iteratively until a termination condition is met. Figure 7 shows the pseudo-code of the ILS approach. ILS is a simple stochastic local search method. It is because of the fact that only few lines of code have to be added to an currently existing local search procedure to apply an ILS algorithm. In spite of its simplicity, it provided a basis for several state-of-the-art algorithms for problems such as the TSP (Johnson, D. S. & McGeoch, L. A., 2002) or scheduling problems (Balas, E., & Vazacopoulos, A. 1998). In the literature, many authors called many different names for ILS like iterated descent (Baum, E. B., 1986), large-step Markov chains (Martin, O., Otto, S. W. & Felten E.W. 1991), chained local optimization (Martin, O., Otto, S. W. , 1996) etc. Nevertheless, the term iterated local search now becomes widely accepted (Lourenc, H. R., Martin, O. & Stützle, T., 2002).

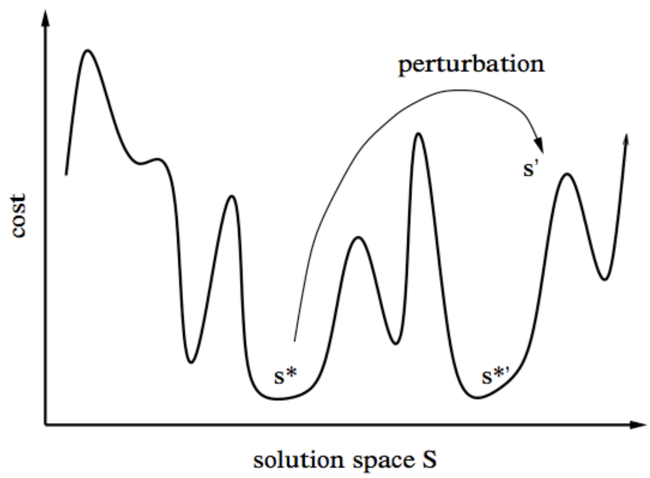


Figure 8. Pictorial Summary of ILS

In Figure 8 shows summary of iterated local search. Starting with a local minimum, applying a perturbation stepping to get to different parts of the search space. After applying LocalSearch, we find a new local minimum.

CHAPTER 5

WEB-BASED SOFTWARE

5.1 Web Application Architecture

This web-based software is developed by combining several programming languages. Some of these languages are used for server side (PHP) and client side (HTML, JAVASCRIPT, CSS). The main scheduling algorithm was programmed with server side language. Graphical user interface and animations are programmed with client side language.

PHP (recursive acronym for PHP: Hypertext Preprocessor) is a widely-used that is especially suited for web development. PHP is not a proper web standard but an open-source technology. It can be embedded into HTML. It is a server-side scripting language designed for web development.

HTML (HyperText Markup Language) is the standard markup language used to create web pages. The purpose of a web browser is to read HTML documents and compose them into visible web pages. HTML tags are the hidden keywords, the browser does not display the HTML tags. These tags tell the browser how to display the text or graphics in the document.

JavaScript (JS) is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously. JavaScript was designed by Netscape to work together with HTML to create more dynamic web pages. It is also being used in server-side programming, the creation of desktop and mobile applications.

CSS (Cascading Style Sheets) is a standart defined by the World Wide Web Consortium, more flexibility and accuracy when defining the appearance of text and formats than standard HTML. CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts. It works with HTML.

The client sends HTTP requests to the Apache web server through the user interface. The Apache server (PHP) solves problem with with a scheduling engine to perform the scheduling process. The processed data is then sent back to the user and displayed on the user interface.

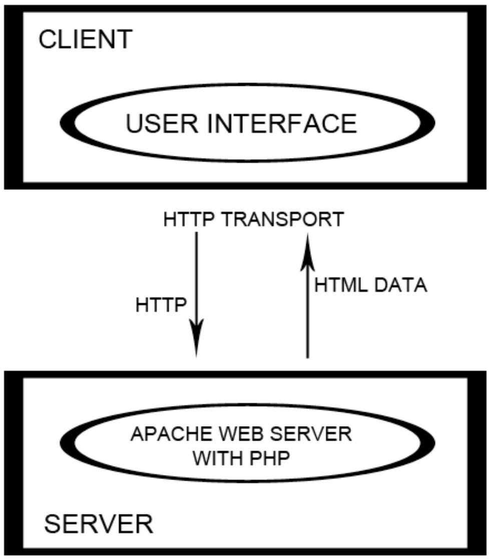


Figure 9. Web Application Model

5.1.1 Web Application Advantages

The web applications have a number of advantages. The ability to update and maintain without distributing and installing is an important reason for developing softwares as web applications. Web application is useful for small and medium enterprises which have hundreds and thousand employees. Software management is done on only a server not on every clients. Software management on every clients machine occurs some management costs. There is no seperately installation on each individual computer. Virtually no IT resources or support needed on each clients. As a user a person does not have to worry about whether the application is up to date. The application is always up to date. Desktop application confined to a physical computer and has mobility constraint, users can access web applications from anywhere with an internet connection. It can easily adopted mobile access. Web-based software tends to have lighter and more simplistic user interfaces, desktop applications tends to have richer and complex user interfaces, so web-based software is usually easy to use. Desktop application, functionality is difficult, so it is usually reserved for new versions of the software. Web applications can be integrated with each other for added functionality. Stand-alone functionality makes it difficult to collaborate in real-time but web application enables the possibility of sharing work and collaborating in real-time.

5.1.2 Web Application Disadvantages

Like there are advantages of web applications, there are certain web-based applications disadvantages as well. If the Internet connectivity is slower, then the application will also take time to run. This may cause finish up the work late. The drawbacks are the inaccessibility of a web application if the internet connection is broken and vulnerability to threats emerging on the internet. Web application can run on local area network to be protected from threats. At the same time, developing a web development often takes more time, as compared to the desktop software development. Since a lot of work has to be done on the compatibility of browsers along with the versions the developers do take considerable amount of time for the same. One has to weigh the advantages of web applications against the disadvantages.

5.2 Graphical User Interface

The user interface will be effective if all user actions are performed with minimal effort. Hence, the user interface is one of the key factors of a successful web-based software. For this reason, this web-based software are designed as easy understandable interface.

The main screen includes four user defined variables. Users should make some settings on this screen before running solving process. These variables are machine number, job number, minimum job processing time, maximum job processing time, replication number, output types and algorithm types. Processing time of jobs for each machine created between minimum and maximum value randomly. User can choose VND 1, VND 2 or ILS algorithms for solving problem. Software includes two types of display output. These are "Only Cmax Results" and "Step by step solution". “Only Cmax Results” type is lighter and faster than other one. In the Figure 10 shows user defined variables form. In Appendix A, html code is given of this user defined variables form.

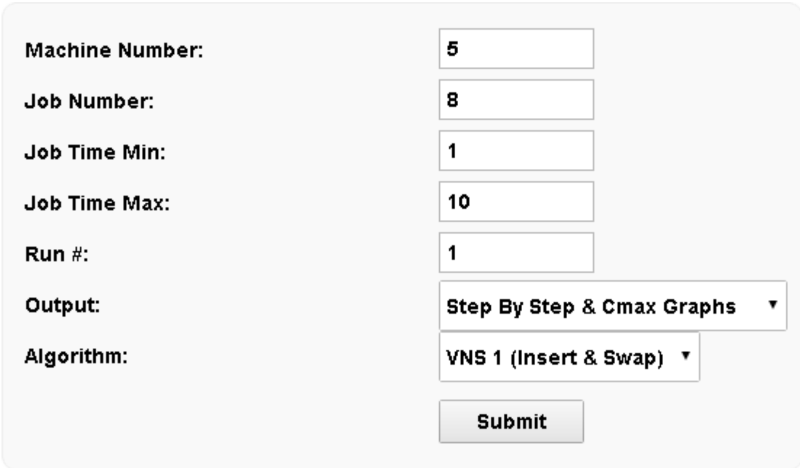
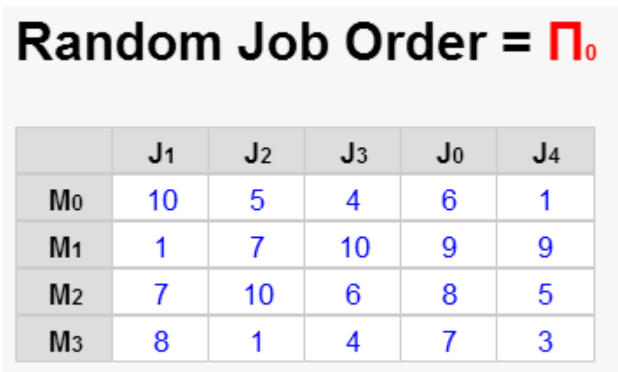


Figure 10. User Defined Variables

Processing time of jobs for each machine will be created after submitting the user defined form. In the Figure 11, machine number machine number defined as 5, job number defined as 8, min processing time as 1, max processing time as 10. First left gray column shows machines, first above gray row shows jobs and blue numbers represents processing time of the jobs. The jobs are sorted randomly and first instance created.



Random Job Order = Π_0

	J_1	J_2	J_3	J_0	J_4
M_0	10	5	4	6	1
M_1	1	7	10	9	9
M_2	7	10	6	8	5
M_3	8	1	4	7	3

Figure 11. Processing Time of Jobs

In the Figure 12 shows scheduling of jobs for each machine. "Play” button is located below of the jobs schedule, it is for simulation/animation of assigning the jobs to machines.

Figure 12. Svheduling of Jobs

In the Figure 13 & Figure 14, randomly selected job were inserted next of another and swapping with each other. All VND steps shown clearly.

Insert (J_3 -> J_4)

	J_1	J_2	J_0	J_4	J_3
M_0	10	5	6	1	4
M_1	1	7	9	9	10
M_2	7	10	8	5	6
M_3	8	1	7	3	4

Figure 13. VND Insert

Swap (J_0 <-> J_2)

	J_2	J_3	J_0	J_1	J_4
M_0	5	4	6	10	1
M_1	7	10	9	1	9
M_2	10	6	8	7	5
M_3	1	4	7	8	3

Figure 14. VNS Swap

In the Figure 15, graph shows result of Cmax values each scheduling.

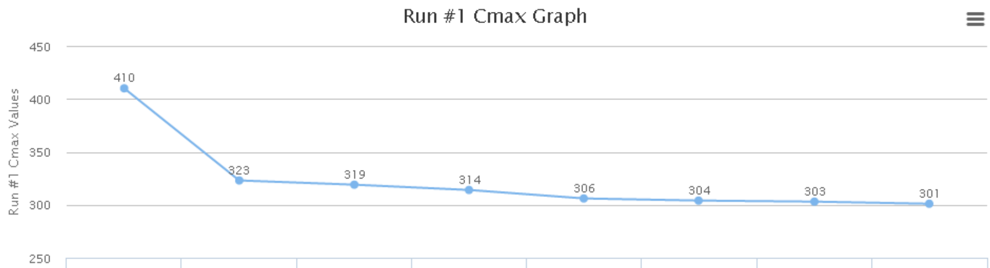


Figure 15. Cmax Graph

CHAPTER 6

COMPUTATIONAL RESULTS

VND1, VND2 and ILS algorithms were coded in PHP and run on an Intel(R) Pentium(R) CPU P6100 @ 2.00 GHz PC with 3 GB memory, 64 bit operating system.

We generated our own benchmarks as follows: We devised instances for 80 jobs with 15, 10 and 5 machines, 70 jobs with 15,10 and 5 machines, 60 jobs with 15, 10 and 5 machines, 50 jobs with 15,10 and 5 machines, 40 jobs with 15,10 and 5 machines, 30 jobs with 15, 10 and 5 machines, 20 jobs with 15, 10 and 5 machines from a uniform distribution of $U(1,100)$. We provided for each instances the average, minimum, maximum, standard deviation, execution time of five runs.

The computational results are given in Table 1, Table 2 and Table 3. Execution times were calculated in seconds. Same instances were calculated with same job processing time in VND 1, VND 2 and ILS.

Table 1. Computational Results of VND 1

J	M	C_{max}^{min}	C_{max}^{max}	C_{max}^{avg}	C_{max}^{std}	Time
80	15	201	240	217,40	17,39	113838s
80	10	259	331	299,40	27,56	75761s
80	5	611	670	627,20	24,41	69103s
70	15	174	220	193,60	16,68	50083s
70	10	219	290	262,00	29,03	45736s
70	5	449	539	506,20	35,04	31775s
60	15	157	178	168,00	9,14	23437s
60	10	167	245	212,40	29,31	22106s
60	5	382	470	447,40	36,78	16045s
50	15	117	147	131,80	11,34	11194s
50	10	157	180	168,60	8,79	10254s
50	5	447	501	468,80	23,75	8521s
40	15	145	210	184,00	24,09	6747s
40	10	168	231	207,80	26,13	5254s
40	5	230	274	254,20	16,56	3357s
30	15	67	88	82,40	8,73	1392s

30	10	75	92	86,20	6,76	984s
30	5	95	125	107,40	12,26	625s
20	15	49	66	58,80	6,53	294s
20	10	63	85	72,80	9,28	212s
20	5	100	161	132,60	24,05	156s
		206,29	254,43	232,81		

Table 2. Computational Results of VND 2

J	M	C_{max}^{min}	C_{max}^{max}	C_{max}^{avg}	C_{max}^{std}	Time
80	15	191	243	211,60	20,01	106724s
80	10	283	332	310,80	18,10	69085s
80	5	573	659	618,40	34,03	53869s
70	15	165	198	183,60	11,84	47845s
70	10	235	265	249,20	11,76	35853s
70	5	429	544	499,20	43,96	31497s
60	15	155	187	164,40	13,13	23647s
60	10	206	237	224,00	12,14	19284s
60	5	380	440	409,00	27,02	14567s
50	15	119	144	128,60	10,11	12457s
50	10	165	196	177,20	12,56	10024s
50	5	365	447	410,20	29,66	8021s
40	15	168	190	180,80	9,09	6457s
40	10	165	196	177,20	12,56	5024s
40	5	228	249	238,00	7,75	4221s
30	15	59	91	80,20	13,26	1315s
30	10	62	90	80,00	11,68	925s
30	5	97	120	105,40	9,63	521s
20	15	50	77	63,00	10,07	253s
20	10	92	92	66,40	17,42	197s
20	5	104	158	136,80	22,03	138s
		204,33	245,48	224,48		

Table 3. Computational Results of ILS

J	M	C_{max}^{min}	C_{max}^{max}	C_{max}^{avg}	C_{max}^{std}	Time
80	15	191	226	209,20	13,48	40711s
80	10	297	333	317,60	13,92	30906s
80	5	580	631	609,80	20,13	20083s
70	15	178	215	193,40	14,84	24007s
70	10	225	260	243,20	13,66	20250s
70	5	437	582	527,80	55,46	17642s
60	15	154	174	161,80	8,04	15372s
60	10	211	247	231,40	14,31	8999s
60	5	475	540	505,20	28,21	6578s
50	15	118	143	133,40	10,01	5439s
50	10	198	247	226,20	22,91	2657s
50	5	365	440	397,20	27,11	1854s
40	15	165	195	177,80	13,46	856s
40	10	157	196	171,60	15,21	752s
40	5	243	315	277,80	28,31	654s
30	15	76	90	82,60	6,47	588s
30	10	79	98	87,60	8,38	352s
30	5	243	315	277,80	28,31	254s
20	15	48	74	64,20	10,73	106s
20	10	66	92	75,40	10,38	84s
20	5	114	173	144,60	22,91	61s
		220,00	266,00	243,60		

As can be seen from Table 1, Table 2 and Table 3, VND2 algorithm was superior to both VND1 and ILS algorithm. Overall average results are also summarized in Table 4. As seen from Table 4 that VND2 outperformed both VND1 and ILS algorithm because it generated the lowest overall averages of minimum Cmax, maximum Cmax and average Cmax (i,e., 204.33, 245.47, 224.47).

Table 4. Overall Avarage of Cmax

	Min. Cmax	Max. Cmax	Avr. Cmax
VND 1	206,28	254,42	232,8
VND 2	204,33	245,47	224,47
ILS	220	266	243,6

However, these results should be analyzed statistically. To do so, From Figure 16 to 21, we provide the interval plots of VND1, VND2 and ILS algorithm in order to see if there is a statistical difference between algorithms . From these Figures, it can be seen that three algorithms are statistically equivalent in terms of all performance measures because their confidence intervals does not coincide.

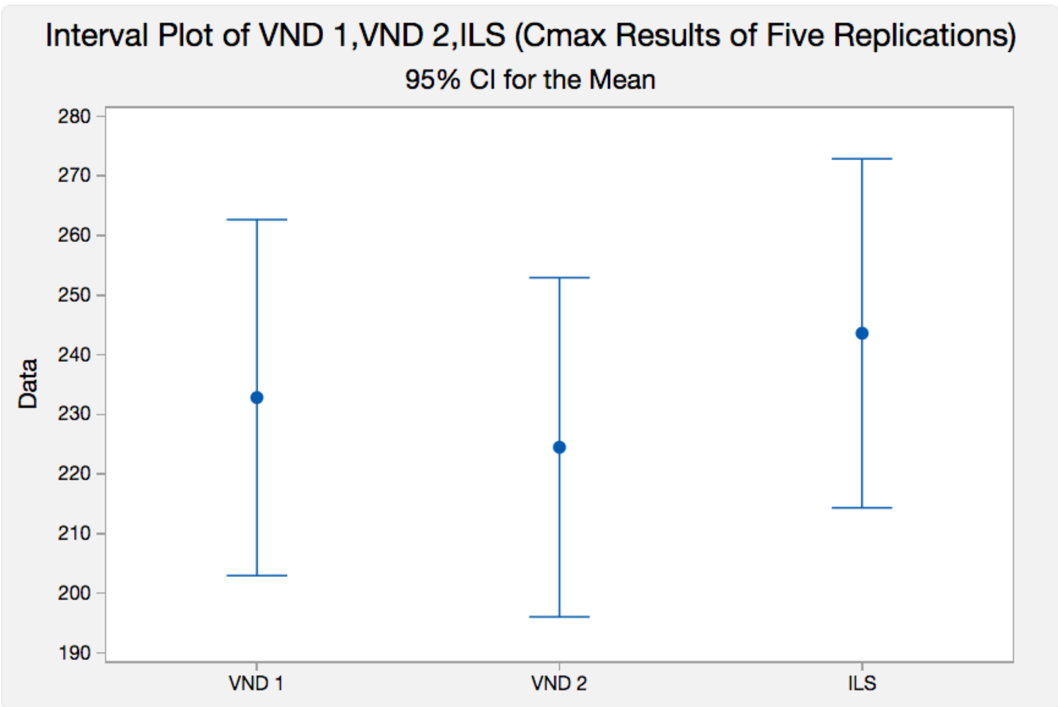


Figure 16. Interval Plot of VND1, VND2 and ILS (All Cmax Results)

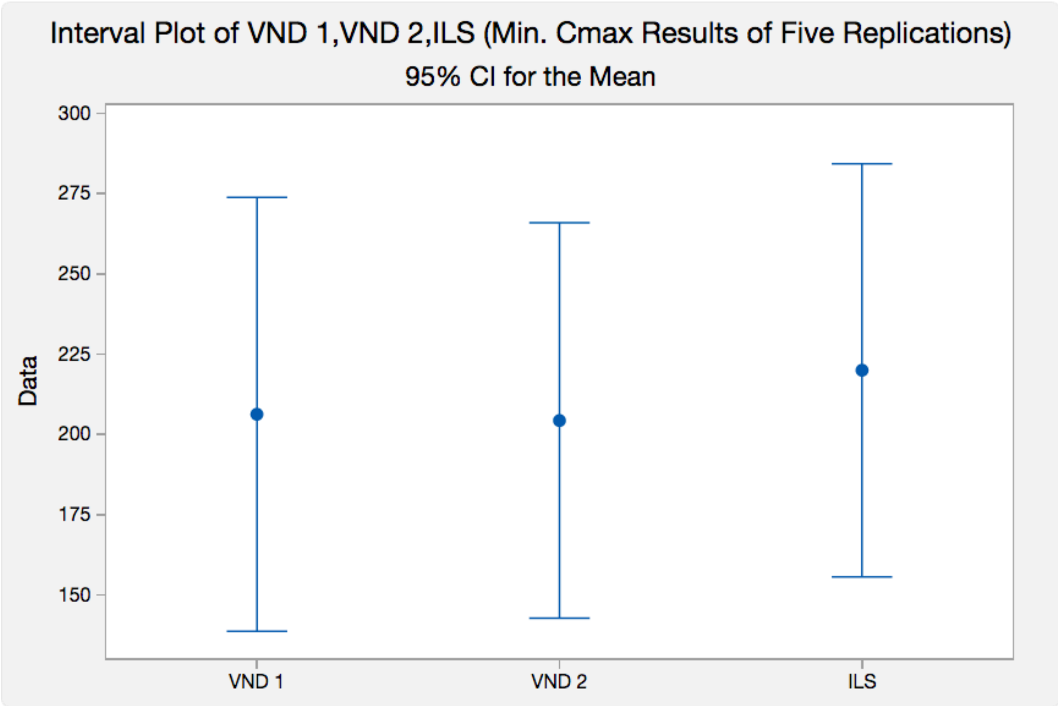


Figure 17. Interval Plot of VND1, VND2 and ILS (Minimum Cmax Results)

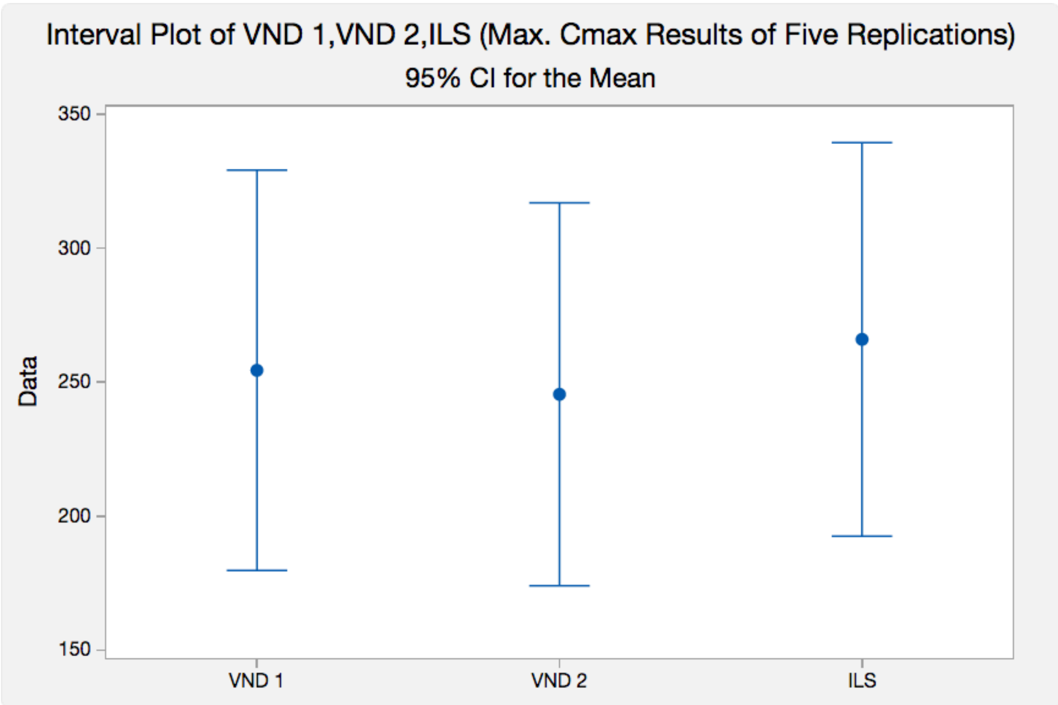


Figure 18. Interval Plot of VND1, VND2 and ILS (Maximum Cmax Results)

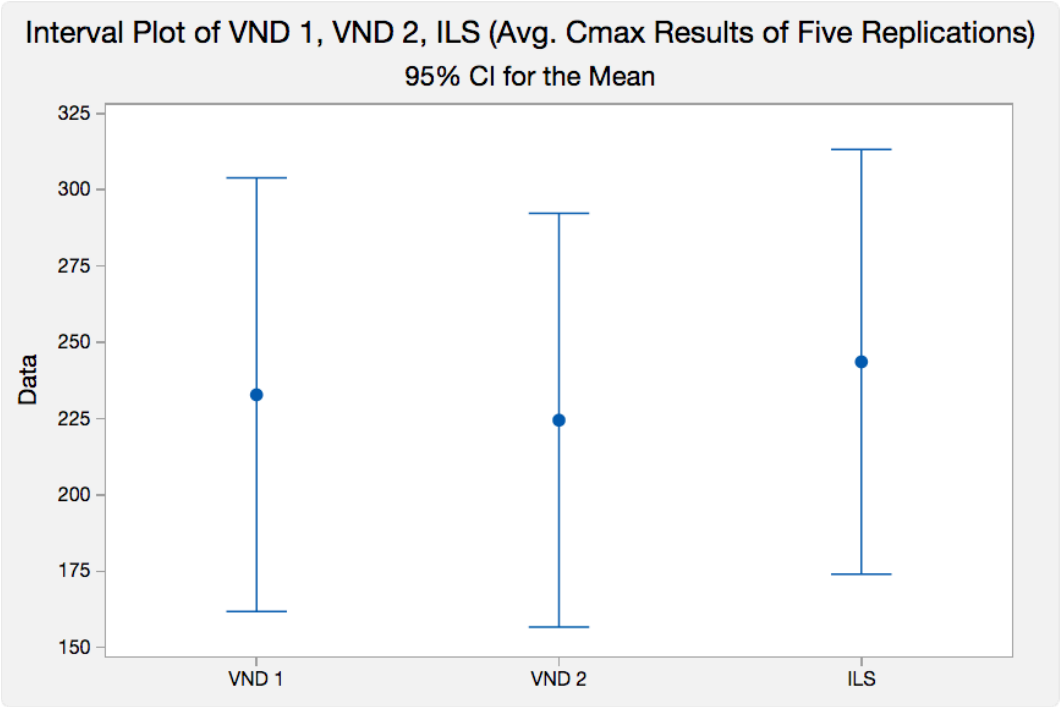


Figure 19. Interval Plot of VND1, VND2 and ILS (Average Cmax Results)

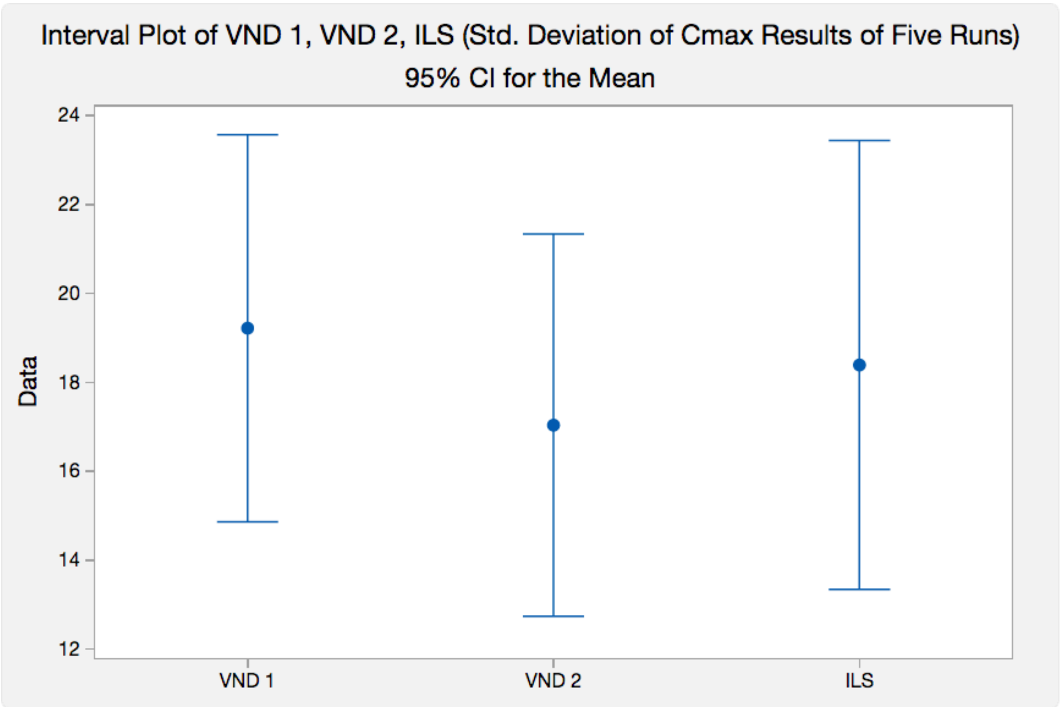


Figure 20. Interval Plot of VND1, VND2 and ILS (Standart Deviations of Cmax Results)

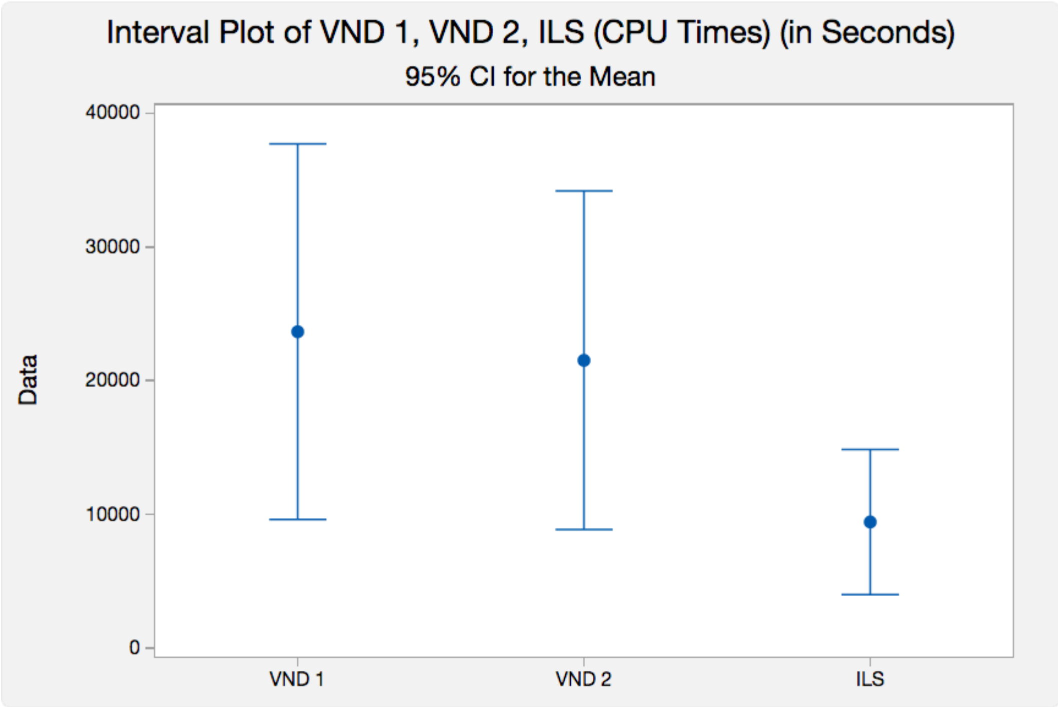


Figure 21. Interval Plot of VND1, VND2 and ILS (CPU Times)

CHAPTER 7

CONCLUSION

In this paper, we combined identical parallel machine problem with web page software. N units independent jobs assigned to m units identical parallel machines to minimize the makesp an is studied. Variable neighborhood search and iterated local search algothims used for solving problem. Web-based software makes it accessible on every device and at everywhere. It can be used for scheduling education with user interaction and easy understandable layout. Some companies can use this for solving parallel machine problem. Web solution was developed with open source technologies, such as PHP, HTML and Javascript. The web solution is built in a modular way, with an intuitive and user-friendly graphical user interface to support man-machine interaction. Web-based technology is a better option for companies required to prepare and implement scheduling problems.

In addition, we wanted to test the CPU time performance of the web-based scheduling software. Experimental results showed that CPU time requirement of a web-based scheduling system is computationally very expensive.

In future, scheduling algorithms can programming with C, C++ or C# console application because of reducing high CPU times. Web-based application can be a bridge with console appplication with end-user. (i.e, ASP.NET or PHP) User can define scheduling variables with web-based application (i.e, machine number, job number etc...). These variables should send to console application as arguments. When the process finished console application, it sends scheduling results to web-based application and user can be informed about process status. Developing an integrated system like this, increases the usability and reduces CPU time.

REFERENCES

Anghinolfi, D., & Paolucci, M. (2007). Parallel machine total tardiness scheduling with a new hybrid metaheuristic approach. *Computers and Operations Research*, 34: 3471-3490.

Balas, E., & Vazacopoulos, A. (1998). Guided local search with shifting bottleneck for job shop schedueling, Management Science 44 (2), 262-275.

Baum, E. B. (1986). Iterated descent: A better algorithm for local search in combinatorial optimization problems, Manuscript

Crainic T. G, Monclar F.C., M. Gendreau, P. Hansen, & N. Mladenovic (2004). Cooperative parallel variable neighborhood search for the p-median. Journal of Heuristics, 10:293–314.

Costa M.C, Monclar F.C., & Zrikem M. (2002). Variable neighborhood decomposition search for the optimization of power plant cable layout. Journal of Intelligent Manufacturing, 13(5): 353–365.

Dessouky, M. (1998). Scheduling identical jobs with unequal ready times on uniform parallel machines to minimize the maximum lateness. *Computers and Industrial Engineering*, 34(4): 793-806.

Dogramaci, A., & Surkis, J. (1979). Evaluation of a heuristic for scheduling independent jobs on parallel identical processors. *Management science*, 25: 1025-1041.

Gao, J., & Sun, L. & Gen, M. (2008). A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. Computers and Operations Research, 35(9): 2892–2907.

Garcl´a-L´opez, F., & Meli´an-Batista, B. & Moreno-P´erez, J. A. & Moreno-Vega J. M. (2002). The parallel variable neighborhood search for the p-median problem. Journal of Heuristics, 8:375–388.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy, A. K. (1979). *Optimization and Approximation in Deterministic Sequencing and Scheduling.* Annals of Discrete Mathematics.

Gupta, S. R. & Smith J. S. (2006). Algorithms for single machine total tardiness scheduling with sequence dependent setups," European Journal of Operational Research, vol. 175, no. 2, pp. 722–739.

Hansen, P. & Mladenovic, N. (1999). An introdıction to variable neighbourhood search, in: S. Voss, S. Martello, I.H. Osman, C. Roucairol (Eds.) Meta-Heuristics: Adances and Trends in Local Search Paradigms for Optimization, Kluwer Academic Publishers, Boston, MA, 433-458.

Hansen, P. & Mladenovic, N. (2003). Variable Neighborhood Search. In F. W. Glover and G. A. Kochenberger Handbook of Metaheuristics, Kluwer Academic Publisher, 145-184.

Hansen, P. & Mladenovic, N. (2001). Indudtrial applications of the variable neighborhood search. In: C. Riberio, P. Hansen (eds.), Essays and surveys in metaheuristics, pages 415–440. Kluwer academic publishers, Boston, United state of America, 1 edition.

Hansen, P. , Mladenovic, N. & Perez-Britos D. (2001). Variable neighborhood decomposition search Journal of Heuristics, 7(4):335–350.

Hansen, P. , Mladenovic, N. & Urosevic D. (2006). Variable neighborhood search and local branch ing. Computers and Operations Research, 33(10):3034–3045.

Hansen, P. , Brimberg,, J. , Urosevic D. & Mladenovic, N. (2007). Primal-dual variable neighborhood search for the simple plant-location problem. INFORMS Journal on Computing, 19(4):552–564.

Hansen, P. , Lazic, J. & Mladenovic, N. (2007). Variable neighbourhood search for colour image quantization. IMA Journal of Management Mathematics, 18(2):207–221.

Hansen, P., Mladenovic, N. & Moreno Perez, J. A. (2008). Variable neighbourhood search: methods and applications. 4OR, 6:319–360.

Hu, Bu. & Raidl., G. R. (2006). Variable neighborhood descent with self-adaptive neighborhood ordering. In In Proceedings of th EU/Meeting on Adaptive, Self-Adaptive, and Multi-Level Metaheuristics, Malaga, Spain, 2006, Malaga, Spain.

Hu, Bu. Leitner M., & Raidl., G. R. (2009). The generalized minimum edge biconnected network problem: Efficient neighborhood structures for variable neighborhood search. Technical Report TR-186-1-07-02.

Johnson, D. S. & McGeoch, L. A. (2002). *Experimental analysis of heuristics for the STSP, in: G. Gutin, A. Punnen (Eds.), The Traveling Salesman Problem and Its Variations, Kluwer Academic Publishers, Dordrecht, The Netherlands, 369-443.*

Lenstra, J. K., Rinnooy, A. K., & Brucker, P. (1977). *Complexity of Machine Scheduling Problems.* Annals of Discrete Mathematics.

Liberti, L. & Drazic.,M (2005). *Variable neighbourhood search for the global optimization of constrained nlps. In In Proceedings of GO Workshop, Almeria, Spain*.

Lourenc, H. R., Martin, O. & Stützle, T. (2002). *Iterated local search, in: F. Glover, G. Kochenberger (Eds.), Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol. 57, Kluwer Academic Publishers, Norwell, MA, 321-353*

Martin, O., Otto, S. W. & Felten E.W. (1991). Large-step Markov chains for the traveling salesman problem, Complex Systems 5 (3). 299-326.

Martin, O., Otto, S. W. (1996). Combining simulated annealing with local search heuristics, Annals of Operations Research 63. 57-75.

McNaughton, R. (1959). Scheduling with deadlines and loss functions. *Management Science*, 1-12.

Min L., and Cheng W. (1999), A genetic algorithm for minimizing the makespan in the case of scheduling identical parallel machines, Artificial Intelligence in Engineering 13, 399–403.

Mladenovic, N. (1995). A Variable Neighborhood Algorithm *A New Metaheuristic for Combinatorial Optimization, Abstracts of papers presented at Optimization Days, Montréal, 112.*

Mladenovic, N., & Hansen, P. (1997). Variable Neighborhood Search. *Computers and Operations Research*, 24: 1097-1100.

Pan, Q. K., Tasgetiren, M. F., & Liang, Y. C. (2008). A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem with makespan and total flowtime criteria. *Computers and Operations Research*, 35(9), 2807-2839.

Park, M. W., & Kim, Y. D. (1997). Search heuristics for a parallel machine scheduling problem with ready times and due dates. *Computers and Industrial Engineering*, 33(3-4): 793-796.

Paula, M. R. D., Ravetti, M. G., Mateus, G. R., & Pardalos P. M. (2007). Solving parallel machines scheduling problems with sequence-dependent setup times using variable neighbourhood search," IMA Journal of Management Mathematics, vol. 18, no. 2, pp. 101–115.

Pessoa, A., Uchoa, E., Aragao, M. P., & Rodrigues, R. (2008). Algorithms over arc-time indexed formulations for single and parallel machine scheduling problems. *Technical Support RPEP*, vol.8 no.8.

Ribeiro, C. C. & Souza, M. C. (2002). Variable Neighbourhood Search for the Degree-Constrained Minimum Spanning Tree Problem, Discrete Applied Mathematics, 118, 43-54

Saurabh Kumar Garg, Rajkumar Buyya & Howard Jay Siegel (2004). Time and cost trade-off management for scheduling parallel applications on Utility Grids. *Future Generation Computer Systems*, vol.26 issue.8, 1344–1355

Sang-Oh Shim & Y-D. Kim (2007). Scheduling on parallel identical machines to minimize total tardiness. European Journal of Operational Research Vol. 177, No. 1, pp. 135-146.

Sevkli, Mehmet & Aydin, M. Emin (2006). A Variable Neighbourhood Search Algorithm for Job Shop Scheduling Problems.

Tasgetiren, M. F., Liang, Y. C., Sevkli, M., & Gencyilmaz, G. (2007). A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. *European Journal of Operational Research*, 177, 1930-1947.

Wilkerson, L. J., & Irwin, J. D. (1971). An improved algorithm for scheduling independent tasks. *AIIE Transactions*, 3: 239-245.

APPENDIX A

DETAILED RESULT OF ALGORITHMS

A1. Detailed VND 1 Run Results

Web Based VND 1 Solution *U(1,100)*											
Jobs	Machines	Cmax Run #1	Cmax Run #2	Cmax Run #3	Cmax Run #4	Cmax Run #5	Min. Cmax	Max. Cmax	Avr. Cmax	Std. Deviation	Server Time
80	15	201	207	240	207	232	201	240	217,40	17,39	113838s
80	10	259	318	292	297	331	259	331	299,40	27,56	75761s
80	5	611	617	624	614	670	611	670	627,20	24,41	69103s
70	15	193	188	193	220	174	174	220	193,60	16,68	50083s
70	10	246	290	219	277	278	219	290	262,00	29,03	45736s
70	5	528	449	514	539	501	449	539	506,20	35,04	31775s
60	15	178	161	168	157	176	157	178	168,00	9,14	23437s
60	10	167	204	226	220	245	167	245	212,40	29,31	22106s
60	5	464	470	382	459	462	382	470	447,40	36,78	16045s
50	15	138	117	130	147	127	117	147	131,80	11,34	11194s
50	10	180	165	157	174	167	157	180	168,60	8,79	10254s
50	5	447	453	501	487	456	447	501	468,80	23,75	8521s
40	15	186	210	195	184	145	145	210	184,00	24,09	6747s
40	10	168	220	231	195	225	168	231	207,80	26,13	5254s
40	5	256	248	230	274	263	230	274	254,20	16,56	3357s
30	15	67	84	86	87	88	67	88	82,40	8,73	1392s
30	10	75	85	89	90	92	75	92	86,20	6,76	984s
30	5	105	95	98	114	125	95	125	107,40	12,26	625s
20	15	56	66	61	49	62	49	66	58,80	6,53	294s
20	10	67	85	80	63	69	63	85	72,80	9,28	212s
20	5	141	100	144	161	117	100	161	132,60	24,05	156s
							206,29	254,43	232,81		

A2. Detailed VND 2 Run Results

Web Based VND 2 Solution *U(1,100)*											
Jobs	Machines	Cmax Run #1	Cmax Run #2	Cmax Run #3	Cmax Run #4	Cmax Run #5	Min. Cmax	Max. Cmax	Avr. Cmax	Std. Deviation	Server Time
80	15	198	212	243	191	214	191	243	211,60	20,01	106724s
80	10	283	310	332	320	309	283	332	310,80	18,10	69085s
80	5	596	637	573	627	659	573	659	618,40	34,03	53869s
70	15	198	185	184	186	165	165	198	183,60	11,84	47845s
70	10	250	241	235	255	265	235	265	249,20	11,76	35853s
70	5	517	429	544	518	488	429	544	499,20	43,96	31497s
60	15	164	160	155	156	187	155	187	164,40	13,13	23647s
60	10	206	224	220	233	237	206	237	224,00	12,14	19284s
60	5	440	435	400	380	390	380	440	409,00	27,02	14567s
50	15	119	120	131	144	129	119	144	128,60	10,11	12457s
50	10	165	174	168	183	196	165	196	177,20	12,56	10024s
50	5	365	408	447	410	421	365	447	410,20	29,66	8021s
40	15	184	175	187	190	168	168	190	180,80	9,09	6457s
40	10	165	174	168	183	196	165	196	177,20	12,56	5024s
40	5	241	228	235	237	249	228	249	238,00	7,75	4221s
30	15	59	85	76	90	91	59	91	80,20	13,26	1315s
30	10	62	75	84	89	90	62	90	80,00	11,68	925s
30	5	98	102	110	97	120	97	120	105,40	9,63	521s
20	15	63	67	58	77	50	50	77	63,00	10,07	253s
20	10	64	68	43	65	92	92	92	66,40	17,42	197s
20	5	156	104	136	158	130	104	158	136,80	22,03	138s
							204,33	245,48	224,48		

A3. Detailed ILS Run Results

Web Based ILS Solution *U(1,100)*											
Jobs	**Machines**	**Cmax Run #1**	**Cmax Run #2**	**Cmax Run #3**	**Cmax Run #4**	**Cmax Run #5**	**Min. Cmax**	**Max. Cmax**	**Avr. Cmax**	**Std. Deviation**	**Server Time**
80	15	210	217	226	191	202	191	226	209,20	13,48	40711s
80	10	297	333	311	324	323	297	333	317,60	13,92	30906s
80	5	622	580	600	616	631	580	631	609,80	20,13	20083s
70	15	178	190	201	215	183	178	215	193,40	14,84	24007s
70	10	240	253	260	238	225	225	260	243,20	13,66	20250s
70	5	532	437	582	561	527	437	582	527,80	55,46	17642s
60	15	154	163	163	155	174	154	174	161,80	8,04	15372s
60	10	211	223	239	237	247	211	247	231,40	14,31	8999s
60	5	488	475	493	530	540	475	540	505,20	28,21	6578s
50	15	141	118	135	143	130	118	143	133,40	10,01	5439s
50	10	205	198	247	238	243	198	247	226,20	22,91	2657s
50	5	395	440	397	365	389	365	440	397,20	27,11	1854s
40	15	189	165	173	195	167	165	195	177,80	13,46	856s
40	10	168	175	162	157	196	157	196	171,60	15,21	752s
40	5	257	289	315	243	285	243	315	277,80	28,31	654s
30	15	76	89	78	90	80	76	90	82,60	6,47	588s
30	10	80	94	79	98	87	79	98	87,60	8,38	352s
30	5	257	289	315	243	285	243	315	277,80	28,31	254s
20	15	62	48	63	74	74	48	74	64,20	10,73	106s
20	10	73	78	66	68	92	66	92	75,40	10,38	84s
20	5	154	130	152	173	114	114	173	144,60	22,91	61s
							220,00	266,00	243,60		

APPENDIX B

USER DEFINED VARIABLE INPUT FORM

```
<!DOCTYPE html>
<html lang=en dir=ltr class=client-nojs>
<title>WEB-BASED SOLUTION FOR SCHEDULING PROBLEM IN IDENTICAL
PARALLEL MACHINES</title>
<meta charset="UTF-8">
 <link href="style.css" rel="stylesheet" />
 <link href="animate.css" rel="stylesheet" />
  <script src="jquery-1.9.0.min.js" type="text/javascript"></script>
</head>
<body>
 <div class="header">WEB-BASED SOLUTION FOR SCHEDULING PROBLEM IN
IDENTICAL PARALLEL MACHINES</div>
 <br />
 <br />
 <form name="process" id="process" method="GET" action="process.php">
  <div style="width:500px; margin:0 auto;border: 1px solid rgb(218, 218, 218);padding:
15px;border-radius: 10px;background: rgb(248, 248, 248);">
   <div style="width:200px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-
height: 34px;">Machine Number: </div>
   <div style="width:200px; float:right; height: 35px;">
    <input type="text" name="machinenumber" id="machinenumber" style="width:95px; font-
size: 14px; font-weight: bold; padding: 4px;" value="2" />
   </div>
   <div class="clear"></div>
   <div style="width:200px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-
height: 34px;">Job Number: </div>
   <div style="width:200px; float:right; height: 35px;">
    <input type="text" name="jobnumber" id="jobnumber" value="10" style="width:95px; font-
size: 14px; font-weight: bold; padding: 4px;" />
   </div>
   <div class="clear"></div>
   <div style="width:200px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-
height: 34px;">Job Time Min: </div>
   <div style="width:200px; float:right; height: 35px;">
    <input type="text" name="jobtimemin" id="jobtimemin" value="1" style="width:95px; font-
size: 14px; font-weight: bold; padding: 4px;" />
   </div>
   <div class="clear"></div>
   <div style="width:200px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-
height: 34px;">Job Time Max: </div>
   <div style="width:200px; float:right; height: 35px;">
    <input type="text" name="jobtimemax" id="jobtimemax" value="10" style="width:95px;
font-size: 14px; font-weight: bold; padding: 4px;" />
   </div>
   <div class="clear"></div>
   <div style="width:200px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-
height: 34px;">Run #: </div>
   <div style="width:200px; float:right; height: 35px;">
    <input type="text" name="runquantity" id="runquantity" value="1" style="width:95px; font-
size: 14px; font-weight: bold; padding: 4px;" />
   </div>
   <div class="clear"></div>
   <div style="width:200px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-
height: 34px;">Output: </div>
   <div style="width:200px; float:right; height: 35px;">
```

```
    <select name="output" id="output" style="width: 175px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-height: 34px;">
     <option value="1">Only Cmax Results</option>
     <option value="2">Step By Step & Cmax Graphs</option>
    </select>
   </div>
   <div class="clear"></div>
   <div style="width:200px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-height: 34px;">Algorithm: </div>
   <div style="width:200px; float:right; height: 35px;">
    <select name="algorithm" id="algorithm" style="width: 175px; float:left; font-weight: bold; font-size: 14px; height: 35px; line-height: 34px;">
     <option value="1">VND 1 (Insert & Swap)</option>
     <option value="2">VND 2 (Swap & Insert)</option>
     <option value="3">ILS</option>
    </select>
   </div>
   <div class="clear" style="height:10px;"></div>
   <div style="width: 200px; float:right; height: 35px;">
    <input type="submit" name="submit" id="submit" value="Submit" style="font-size: 14px;font-weight: bold;padding: 5px;width: 100px;" />
   </div>
   <div class="clear"></div>
  </div>
 </form>
</body>
</html>
```

APPENDIX C

SCHEDULING AND ALGORITHMS CODE

```php
<?php
set_time_limit(0);
ini_set("memory_limit","-1");
ini_set("max_execution_time","0");
ini_set('output_buffering', 'off');
ini_set('zlib.output_compression', 0);
ini_set('implicit_flush', 1);
ob_implicit_flush(true);
session_start();
ob_end_clean();
ob_start();

$debug = 1;
if($debug == 1)
{
 ini_set('display_errors','1');
 ini_set('display_startup_errors','1');
 error_reporting (E_ALL);
}
else
{
 ini_set('display_errors','0');
 ini_set('display_startup_errors','0');
}

$starttime=microtime(TRUE);

$machinenumber='';
$jobnumber='';
$jobtimemin='';
$jobtimemax='';
$runquantity='';
$output='';
$algorithm='';
$samedata='';

$Runs = array();

if(isset($_GET['machinenumber'])){$machinenumber=$_GET['machinenumber'];}
if(isset($_GET['jobnumber'])){$jobnumber=$_GET['jobnumber'];}
if(isset($_GET['jobtimemin'])){$jobtimemin=$_GET['jobtimemin'];}
if(isset($_GET['jobtimemax'])){$jobtimemax=$_GET['jobtimemax'];}
if(isset($_GET['runquantity'])){$runquantity=$_GET['runquantity'];}
if(isset($_GET['algorithm'])){$algorithm=$_GET['algorithm'];}
if(isset($_GET['output'])){$output=$_GET['output'];}
if(isset($_GET['samedata'])){$samedata=$_GET['samedata'];}

// makina ve iş sayısı kadar random processing time
function CreateJobsRandom()
{
 global $machinenumber;
 global $jobnumber;
 global $jobtimemin;
 global $jobtimemax;
```

```
 $MachinesTMP = array();
 for($x=0;$x<$machinenumber;$x++)
 {
  $MachinesTMP[$x] = array();
  for($y=0;$y<$jobnumber;$y++)
  {
   array_push($MachinesTMP[$x],array('Job'=>$y,'Time'=>rand_except($jobtimemin,
$jobtimemax)));
  }
 }
 return $MachinesTMP;
}

// işleri random sıralıyorum
function JobsOrderRandom($MachineDataTMP)
{
 $arrayRandomTMP=array();
 for($j=0;$j<count($MachineDataTMP[0]);$j++)
 {
  $arrayRandomTMP[$j] = $j;
 }

 shuffle($arrayRandomTMP);

 $MachineDataRandomizedTMP = array();
 for($x=0;$x<count($MachineDataTMP);$x++) // makina sayısı kadar
 {
  $MachineDataRandomizedTMP[$x] = array();
  for($y=0;$y<count($MachineDataTMP[0]);$y++) // job sayısı kadar
  {
   array_push($MachineDataRandomizedTMP[$x],array('Job'=>$arrayRandomTMP[$y],'Time'=
>$MachineDataTMP[$x][$arrayRandomTMP[$y]]['Time']));
  }
 }

 return $MachineDataRandomizedTMP;
}

function MachinesTimes($data)
{
 $ScheduleTimesArray = array();
 for($x=0;$x<count($data);$x++)
 {
  $total = 0;
  for($y=0;$y<count($data[$x]);$y++)
  {
   $total = $total + $data[$x][$y]['Time'];
  }
  $ScheduleTimesArray[$x] = $total;
 }

 return $ScheduleTimesArray;
}

function CurrentOptimalMachine($data)
{
 $min = 0;
 $minKey = 0;
 for($x=0;$x<count($data);$x++)
 {
```

```
    if($x==0)
    {
     $min = $data[$x];
     $minKey = 0;
    }
    else
    {
     if($data[$x] < $min)
     {
      $min = $data[$x];
      $minKey = $x;
     }
    }
   }

  return $minKey;
 }

function RunSchedule($MachinesDataTMP)
{
  $ScheduleTMP = array();
  for($x=0;$x<count($MachinesDataTMP);$x++)
  {
   $ScheduleTMP[$x] = array();
  }

  $CurrentOptimalMachine = -1;

  for($j=0;$j<count($MachinesDataTMP[0]);$j++) // job sayısı kadar
  {
   if($j==0 && $CurrentOptimalMachine == -1)
   {
     $CurrentOptimalMachine = rand_except(0,(count($MachinesDataTMP)-1)); // işleme ilk
başladığında rastgele bi makinaya iş atayarak başlıyorum
   }
   else if($j>0)
   {
     $ScheduleTimesArray = MachinesTimes($ScheduleTMP); //anlık makinalara atanan işlerin
toplam zamanları
     $CurrentOptimalMachine =CurrentOptimalMachine($ScheduleTimesArray); //şuan müsait
olan makina yani atanan iş sayısı az olan makina
   }

   array_push($ScheduleTMP[$CurrentOptimalMachine],array('Job'=>$MachinesDataTMP[$Curr
entOptimalMachine][$j]['Job'],'Time'=>$MachinesDataTMP[$CurrentOptimalMachine][$j]['Time'
]));
  }


  $ScheduleTimesArrayTotalTMP = MachinesTimes($ScheduleTMP); //makinalara işler atandı iş
zamanlarının toplamını hesaplıyoruz her bir makina için

  $CMaxTMP = max($ScheduleTimesArrayTotalTMP); //cmax bulunuyor (ençok iş zamanı)
  $CMaxIndexTMP = array_search($CMaxTMP,$ScheduleTimesArrayTotalTMP); // ençok iş
zamanı olan makinayı buluyoruz
  $CMaxJobsTMP = array();
  for($z=0;$z<count($ScheduleTMP[$CMaxIndexTMP]);$z++)  //cmax olan makinaya atanan
işlerin sıralaması
  {
   $CMaxJobsTMP[$z] = $ScheduleTMP[$CMaxIndexTMP][$z]['Job'];
```

```
 }
 $CMaxArrayTMP
=array('Index'=>$CMaxIndexTMP,'Jobs'=>$CMaxJobsTMP,'Value'=>$CMaxTMP);


 $sonuc
=array('MachinesData'=>$MachinesDataTMP,'ScheduleData'=>$ScheduleTMP,'ScheduleTimesA
rrayTotal'=>$ScheduleTimesArrayTotalTMP,'CMaxArray'=>$CMaxArrayTMP);
 return $sonuc;
}
function BestSwapFunction($BestScheduleResult)
{
 $BestScheduleResultTMP=array();
 $BestCmax = $BestScheduleResult['CMaxArray']['Value'];

 for($i=0;$i<(count($BestScheduleResult['MachinesData'][0])-1);$i++)
 {
  $TargetIndexArray = array();

  for($x=($i+1);$x<count($BestScheduleResult['MachinesData'][0]);$x++)
  {
   $TargetIndexArray[] = $x;
  }

  shuffle($TargetIndexArray);

  for($z=0;$z<count($TargetIndexArray);$z++)
  {
   $FirstIndex = $i;
   $SecondIndex = $TargetIndexArray[$z];

   $SwappedMachineData = array();
   $SwappedMachineData
=SwapArrayElement($BestScheduleResult['MachinesData'],$FirstIndex,$SecondIndex);
   $ScheduleResult = RunSchedule($SwappedMachineData);

   if($ScheduleResult['CMaxArray']['Value'] < $BestCmax)
   {
    $BestCmax = $ScheduleResult['CMaxArray']['Value'];

    $BestScheduleResultTMP=array();
    $BestScheduleResultTMP
=array('Data'=>$ScheduleResult,'FirstIndex'=>$FirstIndex,'SecondIndex'=>$SecondIndex);

    //return $BestScheduleResultTMP; // ilk iyi sonucu döndürür
   }
  }
 }

 return $BestScheduleResultTMP;  //en iyi sonucu bulup döndürür
}

function SwapArrayElement($MachineDataTMP,$FirstIndex,$SecondIndex)
{
 for($i=0;$i<count($MachineDataTMP);$i++)
 {
  $tmpelement = $MachineDataTMP[$i][$FirstIndex];
  $MachineDataTMP[$i][$FirstIndex] = $MachineDataTMP[$i][$SecondIndex];
  $MachineDataTMP[$i][$SecondIndex] = $tmpelement;
 }
```

```
 return $MachineDataTMP;
}

function BestInsertFunction($BestScheduleResult)
{
 $BestScheduleResultTMP=array();
 $BestCmax = $BestScheduleResult['CMaxArray']['Value'];

 for($i=0;$i<(count($BestScheduleResult['MachinesData'][0])-1);$i++)
 {
  $TargetIndexArray = array();

  for($x=($i+1);$x<count($BestScheduleResult['MachinesData'][0]);$x++)
  {
   $TargetIndexArray[] = $x;
  }

  shuffle($TargetIndexArray);

  for($z=0;$z<count($TargetIndexArray);$z++)
  {
   $MoveIndex = $i;
   $TargetIndex = $TargetIndexArray[$z];

   $InsertedMachineData = array();
   $InsertedMachineData
=InsertArrayElement($BestScheduleResult['MachinesData'],$MoveIndex,$TargetIndex);
   $ScheduleResult = RunSchedule($InsertedMachineData);

   if($ScheduleResult['CMaxArray']['Value'] < $BestCmax)
   {
    $BestCmax = $ScheduleResult['CMaxArray']['Value'];

    $BestScheduleResultTMP = array();
    $BestScheduleResultTMP
=array('Data'=>$ScheduleResult,'MoveIndex'=>$MoveIndex,'TargetIndex'=>$TargetIndex);

    //return $BestScheduleResultTMP; // ilk iyi sonucu döndürür
   }
  }
 }

 return $BestScheduleResultTMP;  //en iyi sonucu bulup döndürür
}


function InsertArrayElement($MachineDataTMP, $MoveIndex,$TargetIndex)
{
 for($i=0;$i<count($MachineDataTMP);$i++)
 {
  if($MoveIndex > $TargetIndex) $TargetIndex++; // targetindexin hep sağ tarafına insert yapsın
diye yapıyoruz
  $tmp = array_splice($MachineDataTMP[$i], $MoveIndex, 1);
  array_splice($MachineDataTMP[$i], $TargetIndex, 0, $tmp);
 }
 return $MachineDataTMP;
}

function rand_except($min, $max, $excepting = array()) //belli sayılar olmadan random sayı üretir
```

```
{
 $num = mt_rand($min, $max);
 return in_array($num, $excepting) ? rand_except($min, $max,$excepting) : $num;
}


function RunProcess($iterationNumber,$RandomizedMachineDataTMPSame)
{
 global $Runs;
 global $algorithm;
 global $debug;

 $Step = array();

 $RandomizedMachineDataTMP = null;

 if(isset($RandomizedMachineDataTMPSame))
 {
  $RandomizedMachineDataTMP = $RandomizedMachineDataTMPSame;
 }
 else
 {
  $MachineData = CreateJobsRandom();
  $RandomizedMachineDataTMP = JobsOrderRandom($MachineData);
 }

 $BestScheduleResult = array();
 $ScheduleResult = array();
 $ScheduleResult = RunSchedule($RandomizedMachineDataTMP);
 array_push($Step,array('StepResult'=>$ScheduleResult,'Description'=>'Random Job Order =
<span style="color:red;">&#928;<span style="font-size:12px;">0</span></span>'));
 $BestScheduleResult = $ScheduleResult;


 for($iteration=0;$iteration<$iterationNumber;$iteration++)
 {
  if($debug==1)
  {
   echo '<div class="step">';
   echo '<strong>Iteration '.$iteration.':</strong> ';
   ob_flush();
   flush();
  }


  if($algorithm=='1') // VND1
  {
   $kmax=2;
   $k=1;

   do
   {
    $ScheduleResult = array();

    if($k==1)
    {
     $BestInsertFunctionResult =BestInsertFunction($BestScheduleResult);

     if(count($BestInsertFunctionResult)==0)
     {
```

```
  if($debug==1)
  {
   echo '<span style="color:red;">insert not improve</span>, ';
   ob_flush();
   flush();
  }

  $Description = 'Best insert not improve cmax';
  $ScheduleResult = array();
  //array_push($Step, array('StepResult'=>$ScheduleResult,'Description'=>$Description));
 }
 else
 {
  if($debug==1)
  {
   echo '<span style="color:green;">insert improve</span>, ';
   ob_flush();
   flush();
  }

  $ScheduleResult = $BestInsertFunctionResult['Data'];
  $MoveIndex = $BestInsertFunctionResult['MoveIndex'];
  $TargetIndex = $BestInsertFunctionResult['TargetIndex'];
  $Description = 'Best Insert (J<span style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$MoveIndex]['Job'].'</span> -> J<span
style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$TargetIndex]['Job'].'</span>)';
  array_push($Step,array('StepResult'=>$ScheduleResult,'Description'=>$Description));
 }
}

if($k==2)
{
 $BestSwapFunctionResult =BestSwapFunction($BestScheduleResult);

 if(count($BestSwapFunctionResult)==0)
 {
  if($debug==1)
  {
   echo '<span style="color:red;">swap not improve</span>, ';
   ob_flush();
   flush();
  }

  $Description = 'Best swap not improve cmax';
  $ScheduleResult = array();
  //array_push($Step, array('StepResult'=>$ScheduleResult,'Description'=>$Description));
 }
 else
 {
  if($debug==1)
  {
   echo '<span style="color:green;">swap improve</span>,';
   ob_flush();
   flush();
  }

  $ScheduleResult = $BestSwapFunctionResult['Data'];
  $FirstIndex = $BestSwapFunctionResult['FirstIndex'];
  $SecondIndex = $BestSwapFunctionResult['SecondIndex'];
```

```
      $Description = 'Best Swap (J<span style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$FirstIndex]['Job'].'</span> <-> J<span
style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$SecondIndex]['Job'].'</span>)';
      array_push($Step,array('StepResult'=>$ScheduleResult,'Description'=>$Description));
     }
    }

    if(count($ScheduleResult) > 0)
    {
     if($ScheduleResult['CMaxArray']['Value'] <$BestScheduleResult['CMaxArray']['Value'])
     {
      $BestScheduleResult = array();
      $BestScheduleResult = $ScheduleResult;
      $k=1;
     }
     else
     {
      $k++;
     }
    }
    else
    {
     $k++;
    }
   }
   while ($k <= $kmax);
  }
  else if($algorithm=='2') // VND2
  {
   $kmax=2;
   $k=1;

   do
   {
    $ScheduleResult = array();

    if($k==1)
    {
     $BestSwapFunctionResult =BestSwapFunction($BestScheduleResult);
     if(count($BestSwapFunctionResult)==0)
     {
      if($debug==1)
      {
       echo '<span style="color:red;">swap not improve</span>, ';
       ob_flush();
       flush();
      }

      $Description = 'Best swap not improve cmax';
      $ScheduleResult = array();
      //array_push($Step, array('StepResult'=>$ScheduleResult,'Description'=>$Description));
     }
     else
     {
      if($debug==1)
      {
       echo '<span style="color:green;">swap improve</span>,';
       ob_flush();
       flush();
```

```
        }

        $ScheduleResult = $BestSwapFunctionResult['Data'];
        $FirstIndex = $BestSwapFunctionResult['FirstIndex'];
        $SecondIndex = $BestSwapFunctionResult['SecondIndex'];
        $Description = 'Best Swap (J<span style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$FirstIndex]['Job'].'</span> <-> J<span
style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$SecondIndex]['Job'].'</span>)';
        array_push($Step,array('StepResult'=>$ScheduleResult,'Description'=>$Description));
      }
    }

    if($k==2)
    {
      $BestInsertFunctionResult =BestInsertFunction($BestScheduleResult);
      if(count($BestInsertFunctionResult)==0)
      {
        if($debug==1)
        {
          echo '<span style="color:red;">insert not improve</span>, ';
          ob_flush();
          flush();
        }

        $Description = 'Best insert not improve cmax';
        $ScheduleResult = array();
        //array_push($Step, array('StepResult'=>$ScheduleResult,'Description'=>$Description));
      }
      else
      {
        if($debug==1)
        {
          echo '<span style="color:green;">insert improve</span>, ';
          ob_flush();
          flush();
        }

        $ScheduleResult = $BestInsertFunctionResult['Data'];
        $MoveIndex = $BestInsertFunctionResult['MoveIndex'];
        $TargetIndex = $BestInsertFunctionResult['TargetIndex'];
        $Description = 'Best Insert (J<span style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$MoveIndex]['Job'].'</span> -> J<span
style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$TargetIndex]['Job'].'</span>)';
        array_push($Step,array('StepResult'=>$ScheduleResult,'Description'=>$Description));
      }
    }

    if(count($ScheduleResult) > 0)
    {
      if($ScheduleResult['CMaxArray']['Value']<$BestScheduleResult['CMaxArray']['Value'])
      {
        $BestScheduleResult=array();
        $BestScheduleResult = $ScheduleResult;
        $k=1;
      }
      else
      {
        $k++;
```

```
      }
     }
     else
     {
      $k++;
     }
    }
    while ($k <= $kmax);
   }
   else if($algorithm=='3') // ILS
   {
    $ScheduleResult = array();

    $MoveIndex = rand_except(0,(count($BestScheduleResult['MachinesData'][0])-1));
    $TargetIndex = rand_except(0,(count($BestScheduleResult['MachinesData'][0])-
1),array($MoveIndex)); // kendisi hariç bir hedef seçiyoruz random
   $InsertedMachineData=InsertArrayElement($BestScheduleResult['MachinesData'],$MoveInde
x,$TargetIndex); // random bir şekilde 5,3 i değiştirmeliyiz
    $PerturbationScheduleResult =RunSchedule($InsertedMachineData);
    $Description = 'Random Insert (Perturbation) (J<span style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$MoveIndex]['Job'].'</span> -> J<span
style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$TargetIndex]['Job'].'</span>)';
    array_push($Step,array('StepResult'=>$PerturbationScheduleResult,'Description'=>$Descripti
on));

    if($PerturbationScheduleResult['CMaxArray']['Value']
<$BestScheduleResult['CMaxArray']['Value'])
    {
     $BestScheduleResult=array();
     $BestScheduleResult=$PerturbationScheduleResult;
    }


    array_push($Step,array('StepResult'=>$PerturbationScheduleResult,'Description'=>$Descripti
on));


    $random = rand(1,2);
    if($random==1)
    {
     $BestInsertFunctionResult =BestInsertFunction($BestScheduleResult);
     if(count($BestInsertFunctionResult)==0)
     {
      if($debug==1)
      {
       echo '<span style="color:red;">insert not improve</span>, ';
       ob_flush();
       flush();
      }

      $Description = 'Best insert not improve cmax';
      $ScheduleResult = array();
      //array_push($Step, array('StepResult'=>$ScheduleResult,'Description'=>$Description));
     }
     else
     {
      if($debug==1)
      {
       echo '<span style="color:green;">insert improve</span>,';
```

```
    ob_flush();
    flush();
   }

   $ScheduleResult = array();
   $ScheduleResult = $BestInsertFunctionResult['Data'];
   $MoveIndex = $BestInsertFunctionResult['MoveIndex'];
   $TargetIndex = $BestInsertFunctionResult['TargetIndex'];
   $Description = 'Best Insert (J<span style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$MoveIndex]['Job'].'</span> -> J<span
style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$TargetIndex]['Job'].'</span>)';
   array_push($Step,array('StepResult'=>$ScheduleResult,'Description'=>$Description));
  }
 }
 else
 {
  $BestSwapFunctionResult =BestSwapFunction($BestScheduleResult);
  if(count($BestSwapFunctionResult)==0)
  {
   if($debug==1)
   {
    echo '<span style="color:red;">swap not improve</span>,';
    ob_flush();
    flush();
   }

   $Description = 'Best swap not improve cmax';
   $ScheduleResult = array();
   //array_push($Step, array('StepResult'=>$ScheduleResult,'Description'=>$Description));
  }
  else
  {
   if($debug==1)
   {
    echo '<span style="color:green;">swap improve</span>, ';
    ob_flush();
    flush();
   }

   $ScheduleResult = array();
   $ScheduleResult = $BestSwapFunctionResult['Data'];
   $FirstIndex = $BestSwapFunctionResult['FirstIndex'];
   $SecondIndex = $BestSwapFunctionResult['SecondIndex'];
   $Description = 'Best Swap (J<span style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$FirstIndex]['Job'].'</span> <-> J<span
style="font-
size:18px;">'.$BestScheduleResult['MachinesData'][0][$SecondIndex]['Job'].'</span>)';
   array_push($Step,array('StepResult'=>$ScheduleResult,'Description'=>$Description));
  }
 }

 if(count($ScheduleResult) > 0)
 {
  if($ScheduleResult['CMaxArray']['Value'] <$BestScheduleResult['CMaxArray']['Value'])
  {
   $BestScheduleResult=array();
   $BestScheduleResult = $ScheduleResult;
  }
 }
```

```
    }

    if($debug==1)
    {
     echo '</div>';
     ob_flush();
     flush();
    }
  }
  array_push($Runs, $Step);
  return $RandomizedMachineDataTMP;
}

function MachineTimeTotalUntil($data,$index)
{
  $total = 0;
  for($i=0;$i<$index;$i++)
  {
    $total = $total + $data[$i]['Time'];
  }
  return $total;
}

function getCurrentUrl()
{
  $protocol =strpos(strtolower($_SERVER['SERVER_PROTOCOL']),'https') === FALSE ?'http' :
'https';
  $host     = $_SERVER['HTTP_HOST'];
  $script   = $_SERVER['SCRIPT_NAME'];
  $params   = $_SERVER['QUERY_STRING'];
  $currentUrl = $protocol . '://' . $host . $script . '?' . $params;
  return $currentUrl;
}

?>
<!DOCTYPE html>
<html>
  <title>WEB-BASED SOLUTION FOR SCHEDULING PROBLEM IN IDENTICAL
PARALLEL MACHINES</title>
  <meta charset="UTF-8">
  <link href="style.css" rel="stylesheet" />
  <link href="animate.css" rel="stylesheet" />
    <script src="jquery-1.9.0.min.js" type="text/javascript"></script>
  <script src="http://code.highcharts.com/highcharts.js"></script>
  <script src="http://code.highcharts.com/modules/exporting.js"></script>
  <script>
    var starttime = new Date().getTime();
    $(document).ready(function() {
      var endtime = new Date().getTime();
      var displaytime = (endtime-starttime)/1000;
      $('#sure').html(displaytime+'s');
    });
  </script>
</head>
<body>
<div class="header">
  WEB-BASED SOLUTION FOR SCHEDULING PROBLEM IN IDENTICAL PARALLEL
MACHINES
  <br />
  <?php
```

```php
 if($algorithm=='1')
 {
   echo '<br />';
   echo 'VND 1 (Insert & Swap) Results';
   echo '<br />';
   echo '<a href="'.str_replace('&algorithm=1','&algorithm=2',getCurrentUrl()).'&samedata=1"
target="_blank" style="font-size: 15px; color:blue; text-decoration:none;">Run VND 2 with Same
Data (Swap & Insert)</a>';
   echo '<br />';
   echo '<a href="'.str_replace('&algorithm=1','&algorithm=3',getCurrentUrl()).'&samedata=1"
target="_blank" style="font-size: 15px; color:blue; text-decoration:none;">Run ILS with Same
Data</a>';
 }
 else if($algorithm=='2')
 {
   echo '<br />';
   echo 'VND 2 (Swap & Insert)';
   echo '<br />';
   echo '<a href="'.str_replace('&algorithm=2','&algorithm=1',getCurrentUrl()).'&samedata=1"
target="_blank" style="font-size: 15px; color:blue; text-decoration:none;">Run VND 1 with Same
Data (Insert & Swap)</a>';
   echo '<br />';
   echo '<a href="'.str_replace('&algorithm=2','&algorithm=3',getCurrentUrl()).'&samedata=1"
target="_blank" style="font-size: 15px; color:blue; text-decoration:none;">Run ILS with Same
Data</a>';
 }
 else if($algorithm=='3')
 {
   echo '<br />';
   echo 'ILS';
   echo '<br />';
   echo '<a href="'.str_replace('&algorithm=3','&algorithm=1',getCurrentUrl()).'&samedata=1"
target="_blank" style="font-size: 15px; color:blue; text-decoration:none;">Run VND 1 with Same
Data (Insert & Swap)</a>';
   echo '<br />';
   echo '<a href="'.str_replace('&algorithm=3','&algorithm=2',getCurrentUrl()).'&samedata=1"
target="_blank" style="font-size: 15px; color:blue; text-decoration:none;">Run VND 2 with Same
Data (Swap & Insert)</a>';
 }
 ?>
</div>
<?php
if($output=='1')
{
 for($runindex=0;$runindex<$runquantity;$runindex++)
 {
   $smallestcmaxvalue = 0;
   $smallestcmaxinput = 0;

   echo '<div class="header">Run #'.($runindex+1).'</div>';

   if($samedata=='1' &&isset($_SESSION['RandomizedMachineDataTMP_'.$runindex]))
   {
    $_SESSION['RandomizedMachineDataTMP_'.$runindex]
=RunProcess($jobnumber,$_SESSION['RandomizedMachineDataTMP_'.$runindex]);
   }
   else
   {
    $_SESSION['RandomizedMachineDataTMP_'.$runindex] =RunProcess($jobnumber,null);
   }
```

```
    for($loop=0;$loop<count($Runs[$runindex]);$loop++)
    {
     if(count($Runs[$runindex][$loop]['StepResult']) > 0)
     {
      if($smallestcmaxvalue==0)
      {
       $smallestcmaxvalue = $Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'];

       $jobsinput = '';
       for($z=0;$z<count($Runs[$runindex][$loop]['StepResult']['MachinesData'][0]);$z++)
       {
        $jobsinput .= '<span style="font-size:15px;font-weight:bold;">J</span><span style="font-
size:12px;font-
weight:bold;">'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][0][$z]['Job'].'</span>,';
       }

       $smallestcmaxinput = substr_replace($jobsinput ,"",-1);
      }
      else if($smallestcmaxvalue > $Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'])
      {
       $smallestcmaxvalue = $Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'];

       $jobsinput = '';
       for($z=0;$z<count($Runs[$runindex][$loop]['StepResult']['MachinesData'][0]);$z++)
       {
        $jobsinput .= '<span style="font-size:15px;font-weight:bold;">J</span><span style="font-
size:12px;font-
weight:bold;">'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][0][$z]['Job'].'</span>,';
       }

       $smallestcmaxinput = substr_replace($jobsinput ,"",-1);
      }
     }
    }

    echo '<div class="step">';
    echo '<div class="description">Result Run #'.($runindex+1).'</div>';
     echo '<span style="font-size:15px;font-weight:bold;">C<span style="font-size:12px;font-
weight:bold;">Max</span> Value: </span>'.$smallestcmaxvalue;
     echo '<br />';
     echo '<br />';
     echo '<span style="font-size:15px;font-weight:bold;">Input:</span> '.$smallestcmaxinput;
     echo '<br />';
     echo '<br />';
    echo '</div>';
  }
}

if($output=='2')
{
  $javascriptgeneralgraphcategories = '';
  $javascriptgeneralgraphvalues = '';

  for($runindex=0;$runindex<$runquantity;$runindex++)
  {
    $smallestcmaxvalue = 0;
    $smallestcmaxinput = 0;
    $javascriptgraphcategories = '';
    $javascriptgraphvalues = '';
```

```
    echo '<div class="header">Run #'.($runindex+1).'</div>';

    if($samedata=='1' &&isset($_SESSION['RandomizedMachineDataTMP_'.$runindex]))
    {
     $_SESSION['RandomizedMachineDataTMP_'.$runindex]
=RunProcess($jobnumber,$_SESSION['RandomizedMachineDataTMP_'.$runindex]);
    }
    else
    {
     $_SESSION['RandomizedMachineDataTMP_'.$runindex] =RunProcess($jobnumber,null);
    }


    for($loop=0;$loop<count($Runs[$runindex]);$loop++)
    {
     echo '<div class="step">';
     echo '<div class="description">'.$Runs[$runindex][$loop]['Description'].'</div>';

       if(count($Runs[$runindex][$loop]['StepResult']) > 0)
       {
        echo '<div>';
         echo '<div class="column">';
          echo '<div class="cell"></div>';
          for($x=0;$x<count($Runs[$runindex][$loop]['StepResult']['MachinesData']);$x++)
          {
           echo '<div class="cell" style="color: black;background: rgb(221, 221, 221);font-weight:
bold;">M<span style="font-size:12px;">'.$x.'</span></div>';
          }
         echo '</div>';

         for($j=0;$j<count($Runs[$runindex][$loop]['StepResult']['MachinesData'][0]);$j++)
         {
          echo '<div class="column">';
           for($x=0;$x<count($Runs[$runindex][$loop]['StepResult']['MachinesData']);$x++)
           {
            if($x==0)
            {
             echo '<div class="cell">J<span style="font-
size:12px;">'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][$x][$j]['Job'].'</span></div
>';
            }
            echo '<div class="cell" id="Table-M'.$x.'-
'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][$x][$j]['Job'].'-iteration-
'.$loop.'">'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][$x][$j]['Time'].'</div>';
           }
          echo '</div>';
         }
        echo '</div>';

        $onclick='';
        for($m=0;$m<count($Runs[$runindex][$loop]['StepResult']['MachinesData']);$m++)
        {
         echo '<div style="height: 30px; clear: both; padding-top: 20px;" class="Solution-
'.$loop.'">';
          echo '<div style="width: 45px; height: 21px; float:left;padding-top: 4px;text-align:
center;font-weight: bold;">M<span style="font-size:12px;">'.$m.'</span>:</div>';
          $timeout = 0;
          for($i=0;$i<count($Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m]);$i++)
          {
```

```
                echo '<div class="schedulebar" id="M'.$m.'-
'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'-iteration-'.$loop.'"
style="float:left; height:20px; padding-top:5px;
width:'.($Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Time']*33).'px; text-
align:center; opacity:1; background:#eee; border:1px solid #ccc; font-weight:bold; font-size:
12px;">';
                 echo 'J<span style="font-
size:10px;">'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'</span>='.$R
uns[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Time'];
                echo '</div>';

                echo '<script>';
                 echo '$("#Table-M'.$m.'-
'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'-iteration-'.$loop.',
#M'.$m.'-'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'-iteration-
'.$loop.'").hover(';
                  echo 'function () {';
                   echo '$(\'#M'.$m.'-
'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'-iteration-
'.$loop.'\').css(\'background\',\'#ffff99\');';
                   echo '$(\'#Table-M'.$m.'-
'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'-iteration-
'.$loop.'\').css(\'background\',\'#ffff99\');';
                  echo '},';
                  echo 'function () {';
                   echo '$(\'#M'.$m.'-
'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'-iteration-
'.$loop.'\').css(\'background\',\'#eee\');';
                   echo '$(\'#Table-M'.$m.'-
'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Job'].'-iteration-
'.$loop.'\').css(\'background\',\'#fff\');';
                  echo '}';
                 echo ');';
                echo '</script>';

              }
              echo '<div style="clear:both;width:0;height:0;"></div>';
            echo '</div>';
            echo '<div style="clear:both;width:0;height:0;"></div>';
          }

          echo '<br />';
          echo '<span style="font-size:16px; font-weight:bold;">C<span style="font-
size:13px;">Max</span> : </span> ';
          echo '<span style="font-size:16px; font-weight:bold;">M<span style="font-
size:13px;">'.$Runs[$runindex][$loop]['StepResult']['CMaxArray']['Index'].'</span> | </span>';
          echo '<span style="font-size:16px; font-
weight:bold;">'.$Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'].'</span> | </span>';
          echo '<span style="font-size:16px; font-weight:bold;">J<span style="font-
size:13px;">'.implode('</span>,J<span style="font-
size:13px;">',$Runs[$runindex][$loop]['StepResult']['CMaxArray']['Jobs']).'</span>'.'
'.'</span></span>';
          echo '<button id="play'.$loop.'">Play</button>';
          echo '<br />';
          echo '<br />';
          echo '<script>';
            echo '$("#play'.$loop.'").click(function(){ ';
              echo 'var javascriptArray=null;';
              echo 'javascriptArray = new
Array('.count($Runs[$runindex][$loop]['StepResult']['MachinesData']).');';
```

```
            for($m=0;$m<count($Runs[$runindex][$loop]['StepResult']['MachinesData']);$m++)
            {
             echo 'javascriptArray['.$m.'] = new
Array('.count($Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m]).');';
             for($i=0;$i<count($Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m]);$i++)
             {
               echo
'javascriptArray['.$m.']['.$i.']="'.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['J
ob'].','.$Runs[$runindex][$loop]['StepResult']['ScheduleData'][$m][$i]['Time'].'";';
             }
            }
            echo 'Animation('.$loop.',javascriptArray);';
          echo '});';
        echo '</script>';
      }
    echo '</div>';


    if(count($Runs[$runindex][$loop]['StepResult'])>0)
    {
      $jobsinputtmp = '';
      for($z=0;$z<count($Runs[$runindex][$loop]['StepResult']['MachinesData'][0]);$z++)
      {
        $jobsinputtmp .= '<span style="font-size:15px;font-weight:bold;">J</span><span
style="font-size:12px;font-
weight:bold;">'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][0][$z]['Job'].'</span>,';
      }
      $jobsinputtmp = substr_replace($jobsinputtmp ,"",-1);

      $javascriptgraphcategories .= "'<span style=\"font-size:15px;font-weight:bold;
color:blue;\">Input:</span>".$jobsinputtmp."',";
      $javascriptgraphvalues .= $Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'].',';

      if($smallestcmaxvalue==0)
      {
        $smallestcmaxvalue = $Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'];

        $jobsinput = '';
        for($z=0;$z<count($Runs[$runindex][$loop]['StepResult']['MachinesData'][0]);$z++)
        {
          $jobsinput .= '<span style="font-size:15px;font-weight:bold;">J</span><span style="font-
size:12px;font-
weight:bold;">'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][0][$z]['Job'].'</span>,';
        }

        $smallestcmaxinput = substr_replace($jobsinput ,"",-1);
      }
      else if($smallestcmaxvalue > $Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'])
      {
        $smallestcmaxvalue = $Runs[$runindex][$loop]['StepResult']['CMaxArray']['Value'];

        $jobsinput = '';
        for($z=0;$z<count($Runs[$runindex][$loop]['StepResult']['MachinesData'][0]);$z++)
        {
          $jobsinput .= '<span style="font-size:15px;font-weight:bold;">J</span><span style="font-
size:12px;font-
weight:bold;">'.$Runs[$runindex][$loop]['StepResult']['MachinesData'][0][$z]['Job'].'</span>,';
        }
        $smallestcmaxinput = substr_replace($jobsinput ,"",-1);
      }
```

```
    }
  }

  $javascriptgeneralgraphcategories .= "'<span style=\"font-size:15px;font-weight:bold;
color:blue;\">Input:</span>".$smallestcmaxinput."',";
  $javascriptgeneralgraphvalues .= $smallestcmaxvalue.',';

  echo '<div class="step">';
  echo '<div class="description">Result Run #'.($runindex+1).'</div>';
    echo '<span style="font-size:15px;font-weight:bold;">C<span style="font-size:12px;font-
weight:bold;">Max</span> Value: </span>'.$smallestcmaxvalue;
    echo '<br />';
    echo '<br />';
    echo '<span style="font-size:15px;font-weight:bold;">Input:</span> '.$smallestcmaxinput;
    echo '<br />';
    echo '<br />';
  echo '</div>';
  echo '<div id="container-'.$runindex.'" style="width: 100%; height: 400px;margin-bottom:
200px;margin-top: 60px;"></div>';
  ?>
  <script>
   $(function () {
    $('#container-<?php echo $runindex; ?>').highcharts({
        chart: {
            type: 'line',
            zoomType: 'x',
            spacingRight: 20
        },
        title: {
            text: 'Run #<?php echo ($runindex+1); ?> Cmax Graph'
        },
        subtitle: {
            text: ''
        },
        xAxis: {
            categories: [<?php echo $javascriptgraphcategories;?>]
        },
        yAxis: {
            title: {
                text: 'Run #<?php echo ($runindex+1); ?> Cmax Values'
            }
        },
        tooltip: {
            enabled: true,
            formatter: function() {
                return this.x +'<br /><br /><span style="font-size:15px; font-weight:bold;
color:blue;">Cmax:</span> <span style="font-size:15px; font-weight:bold;">'+ this.y+'</span>';
            }
        },
        plotOptions: {
            line: {
                dataLabels: {
                    enabled: true
                },
                enableMouseTracking: true
            }
        },
        series: [{
            name: 'Line',
            data: [<?php echo $javascriptgraphvalues; ?>]
```

```
            }]
        });

        $('#container-<?php echo $runindex; ?> .highcharts-axis-
labels:eq(0)').css('visibility','hidden');
      });
      </script>
    <?php
  }
  ?>
  <div id="container" style="width: 100%; height: 400px;margin-bottom: 200px;margin-top:
60px;"></div>

  <script>
  function Animation(loop,scheduletimearray)
  {
    $('.Solution-'+loop+' .schedulebar').css({ opacity:0, width:0});
    for(var m=0;m<scheduletimearray.length;m++)
    {
      var timeout = 0;
      for(var j=0;j<scheduletimearray[m].length;j++)
      {
        var job = scheduletimearray[m][j].split(',')[0];
        var time = scheduletimearray[m][j].split(',')[1];
        var animasyonzamani = time*500;
        var genislik = time*33;

        (function(m,job,loop,animasyonzamani,time,genislik,timeout){
          setTimeout(function(){
            $('#M'+m+'-'+job+'-iteration-'+loop).css('background','#ffff99');
            $('#Table-M'+m+'-'+job+'-iteration-'+loop).css('background','#ffff99');

            $('#M'+m+'-'+job+'-iteration-'+loop).animate({opacity:1, width:genislik+'px' },
animasyonzamani, function() {
              $('#M'+m+'-'+job+'-iteration-'+loop).css('background','#eee');
              $('#Table-M'+m+'-'+job+'-iteration-'+loop).css('background','#fff');
            });
          },timeout);
        })(m,job,loop,animasyonzamani,time,genislik,timeout);

        timeout = timeout + animasyonzamani;
      }
    }
  }

  $(function () {
    $('#container').highcharts({
        chart: {
            type: 'line',
            zoomType: 'x',
            spacingRight: 20
        },
        title: {
            text: 'All Runs Cmax Graph'
        },
        subtitle: {
            text: ''
        },
        xAxis: {
            categories: [<?php echo$javascriptgeneralgraphcategories; ?>]
```

```
        },
        yAxis: {
            title: {
                text: 'All Runs Cmax Values'
            }
        },
        tooltip: {
            enabled: true,
            formatter: function() {
                return this.x +'<br /><br /><span style="font-size:15px; font-weight:bold;
color:blue;">Cmax:</span> <span style="font-size:15px; font-weight:bold;">'+ this.y+'</span>';
            }
        },
        plotOptions: {
            line: {
                dataLabels: {
                    enabled: true
                },
                enableMouseTracking: true
            }
        },
        series: [{
            name: 'Line',
            data: [<?php echo $javascriptgeneralgraphvalues; ?>]
        }]
    });
    $('#container .highcharts-axis-labels:eq(0)').css('visibility','hidden');
  });
  </script>
<?php
}
  $endtime=microtime(TRUE);
  $calculationtime=$endtime-$starttime;
  $calculationtime=number_format($calculationtime,2);
  echo '<br />';
  echo '<div style="font-size:13px; text-align:center; font-weight:bold;">';
  echo 'Calculation Time (Server Side): '.$calculationtime.'s';
  echo '<br /><br />';
  echo 'Displaying Time (Client Side): <span id="sure">0s</span></div>';
  ob_end_flush();
?>
</body>
</html>
```