OPERATIONS SCHEDULING FOR MANUFACTURING SYSTEMS WITH PARALLEL COMPUTING

M. Blaho, M. Foltin*, P. Nagy*, M. Hudáček*, K. Kopčok***

*Institute of Control and Industrial Informatics, FEI STU

Ilkovičová 3, 812 19 Bratislava, Slovak Republic

**eProjekt Slovensko s.r.o., Nobelová 16, 836 14 Bratislava, Slovak Republic

Abstract

Some manufacturing processes can make products parallel. Many researchers work on the problem of the maximizing manufacturing speed, operating efficiency or the minimization of the overload of the parallel system. Parallel computing can accelerate the optimization process with condition that the computing is much longer then sending, receiving and handling of the computed data. This paper compares the parallel computing in MATLAB Parallel Computing Toolbox and Java RMI.

1 Introduction

The changes in the manufacturing process come very often today. It is necessary that we adapt to the needs of the markets. The products are usually made on the production line with the set of sequential operations from the one machine to another. Production lines are designed for the mass production of various types of products like house electronic or cars. On the other side some systems can make products parallel, they share the same machines. The examples of such systems are the telecommunications, computers or multiprocessors systems.

Another example of parallel system is the flexible manufacturing system. The basic idea is to create the production line capable of making the products parallel with better response time to the needs of the market and effectively make use of the expensive machines for the parallel manufacturing. Flexible manufacturing system consists of a large number of configurable stations and different production paths. Effective operations scheduling and controlling of such a system is complex and difficult problem.

2 Operations scheduling

With the operations scheduling problem in real time we try to accomplish changing goals in the production. It is necessary to think of the limited recourses and random failure of the machines. Research of operations scheduling in flexible manufacturing systems focuses on the modeling and optimization of the material flow. Typical examples are maximization of the manufacturing speed, system capacity, machines usage or minimization of the system overload.

The thesis of Kopčok[1] deals with problem of operations scheduling for the parallel flexible manufacturing system with the focus on the time minimization for the needed operations. Some assumptions must be made:

- systems consist of multiple inputs, outputs and manufacturing machines
- processes are independent from each other and they share same machines
- for any operation there is alternative (it can be made on other machine)
- operation is processing only at one machine
- there is sufficient numbers of object for manufacturing
- loop is terminated when all products are at the output or when all operations are finished

Such system can be expressed on figure 1.

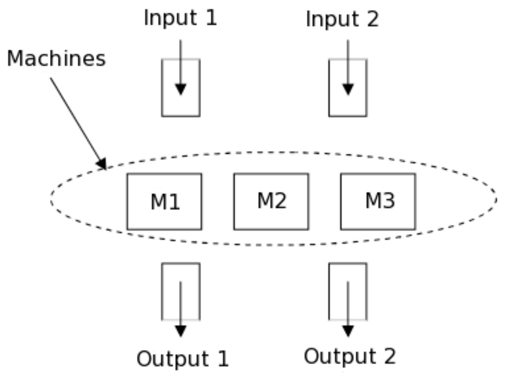

Figure 1: Illustration of the flexible manufacturing system

There are many approaches for modeling of the manufacturing systems like Petri nets or Statecharts. Model of simple system is in the thesis of Kopčok[1]. From Petri nets systems can be expressed in form of the matrices for example:

$$P_1 = \begin{bmatrix} 3 & 4 & 0 \\ 0 & 3 & 2 \end{bmatrix} \quad P_2 = \begin{bmatrix} 4 & 0 & 2 \\ 3 & 4 & 4 \end{bmatrix} \tag{1}$$

For each process there is one matrix. Rows represent operation of the process. Columns represent the machines on which operations can be made. If we can't made operation on the specific machine there is a zero in that column. The cells represent time for the operations on each machine. For example if we take matrix P_1 then the first row and second column means that the first operation on the second machine will take 4 unit times. The first row and second column of matrix P_2 means that the first operation on the second machine can't be made. Kopčok[1] designed two methods for operations scheduling base on this model.

2.1 Heuristic approach

Heuristic approach is based on the heuristic search trough state space. Algorithm doesn't search in the whole state space, but it search for optimal or near optimal path. Path is determined by the rules which are based on assumptions. Operations are divided into two groups. First group create operations which can be made on the multiple machines. Alternative which take minimal time is called fastest operation time. Second group are operations that can be made on single machine and are called single machine operations. Heuristic method consists of five rules and two exceptions.

First rule is that every operation should by make on the machine with fastest operation time if it is possible. Second rule tells that with all operations in particular time with fastest operation time the highest priority has the operation with longest time. It is based on assumption that if the algorithm didn't take this alternative there could be situation where other operation with non-minimal time is chosen. Third rule tells that every machine should be working maximum time. If there is possibility for the machine to work it should be working. Fourth rule tells about situation when on the machine is not possible to do operation with fastest time. In that case is operation shifted to the end of the decision. Fifth rule solve situation that there is no fastest operation time on the machine. In that case we take the operation with shortest time. First exception is that the single machine operations have maximum priority. The second is that no of the mentioned rules holds. Because the algorithm is dependent on the first machines choice it is necessary to repeat all rules with all possible starts. Example of the heuristic method is in the thesis of Kopčok[1].

2.2 Genetic approach

Genetic algorithm is optimization method which tries to simulate the evolution in short time. They are capable of getting away from local extreme and get near to global extreme. Genetic algorithms search in every direction but they are time-consuming. Algorithm works with individuals in population. Individuals represent potential solution. Each solution is evaluated with fitness function.

According to Sekaj[2] after evaluating of the fitness function next population is stochastically selected based on the some genetic operations:

- selection (based on fitness, random, …)
- crossover (recombination)
- mutation

Crossover and mutation are also called reproduction. Genetic algorithm consists of few steps. Choosing or generating initial population and evaluating the fitness function of each individual in that population. Then repeatedly until termination selection of new individuals (best or random), crossover and mutation operations, evaluating fitness function of the new individuals and finally replacing old population with new one. Short figure of genetic algorithm is on figure 2.

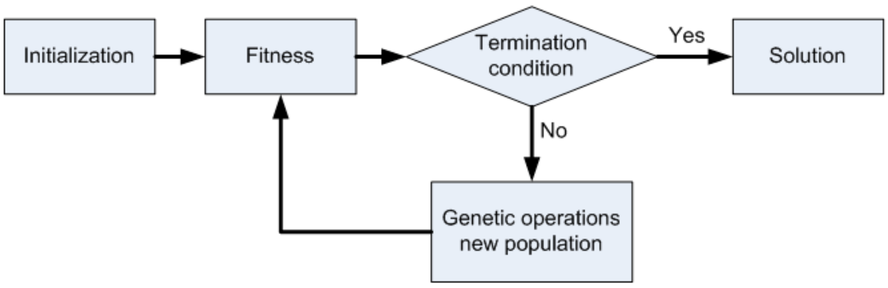

Figure 2: Genetic algorithm

Genetic algorithm was used for searching optimal solution in operations scheduling. Individuals were selected as a sequence of machines and priorities (if two operations want to be realized on the machine). Processes matrices remain the same as in equation (1). Kopčok[1] designed matrix for evaluation of the fitness function.

$$\begin{array}{c} \textit{process} \\ \textit{operation} \\ \textit{state} \\ \textit{machine} \\ \textit{realization time} \\ \textit{priority} \end{array} \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 2 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 3 \\ 3 & 3 & 4 & 4 \\ 2 & 3 & 1 & 4 \end{bmatrix} \tag{2}$$

First row represents number of the process, second operation in that process. Third row represents state for the state of operation. If it is zero then the operation is not evaluating. Fourth row tells us on which machine is operation executing. Fifth row shows how many units time operation will be realized. Last row, as it was mentioned, represent priority for the process. Priority is necessary if the are two operations want to be realized on one machine simultaneously. Each step of the fitness function represents unit time. If state of operation don't equal zero then realization time is reduced by one. Once realization time is zero the column is removed for better clarity. Operations of the processes must be realized in the defined order. Example of the genetic method is in the thesis of Kopčok[1].

3 Parallel computing

Traditionally program has been written as a sequence of computations. Program ran on single computer having a single CPU. Instructions of the program were executed one after another. Some of today's programs try to use multiple CPUs for computations. Programs are broken into parts that can be solved concurrently. Parts are computed on different CPUs. Compute recourses can include single computer with multiple processors, any number of computers connected by network or both. There are a many reasons to use parallel computing, one of them is that it save time and/or money [3]. Some examples of the parallel computing are project like Seti@home, research of cancer, financial and economics modeling, optimization, World community grid, data mining and others [3,4,5].

3.1 Architecture and test functions

For the testing of the parallel computations we chose genetic algorithms. Genetic algorithms are typical example of parallel problem because they compute in every step the same fitness function with changed inputs. Population of genetic algorithm contains several individuals. Individuals can be computed on several client workstations. Such architecture can is expressed on the figure 3. Two problems were computed with parallel computing. First was Rastrigin benchmark function:

$$f(x_1,...,x_n) = 10.n + \sum_{i=1}^{n}(x_i^2 - 10.\cos(2.\pi.x_i)) \tag{3}$$

The second problem was finding optimal operations scheduling described in section 2.2. The test was executed in laboratory for networked control systems. The hardware configuration of the workstations was AMD Athlon64 3800+ computers with frequency 2.41GHz and 1GB operation memory with Windows operating system.

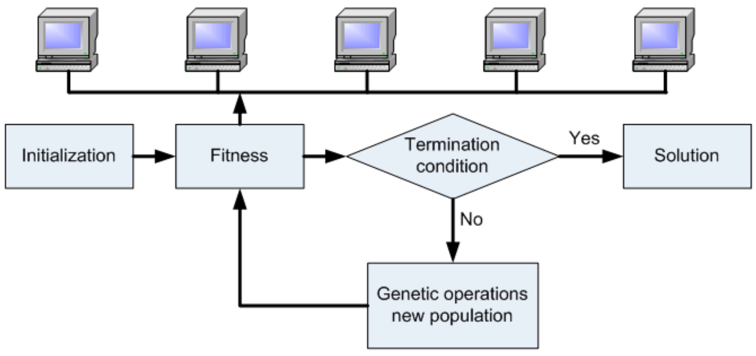

Fig. 3: Parallel computing architecture

3.2 Matlab Parallel Toolbox

Parallel Computing Toolbox software allows you to offload work from one MATLAB session (the client) to other MATLAB sessions, called workers. You can use multiple workers to take advantage of parallel processing. Parallel Computing Toolbox software allows you to run as many as four MATLAB workers on your local machine in addition to your MATLAB client session. MATLAB Distributed Computing Server software allows you to run as many MATLAB workers on a remote cluster of computers as your licensing allows [7,8,9,10].

The bachelor thesis of Hudáček[4] was created one client and the workers computed the fitness function as it was described sooner. In the thesis the parfor function was used for parallel computation. The Rastrigin function benchmark function showed no acceleration of the computations because is too simple for parallel computing. It is because the time for computations was smaller then the sending, receiving and handling of the computed data. The significant improvement shows the computations of the Kopcok's genetic approach for operations scheduling of which we talk in results.

3.3 Java RMI

Java programming language is a high-level language that is platform independent (Windows, Linux, Os X, …). The Java Remote Method Invocation (RMI) system allows an object running in one Java virtual machine to invoke methods on an object running in another Java virtual machine. Java RMI provides for remote communication between programs written in the Java programming language [6].

In diploma thesis of Nagy[5] was designed Java RMI client-server application for remote communication between the client and workers. Several classes for communication and computation were programmed. The first test was on the Rastrigin benchmark function. As well as with Matlab, the Rastrigin function was too simple for parallel computations and there wasn’t any betterment. The benefit of the parallel computation with Java RMI is also to see in the computations of the Kopcok's genetic approach for operations scheduling of which we talk in results.

3.4 Results

Flexible manufacturing system PVS 14 from Kopčok[1] was tested as we mentioned in the Matlab and Java environments. First test compared environments in non-parallel computing. Java environment was significant faster because it use only necessary resources. On the other side Matlab provide easy sets of commands which facilitated work. Java was faster approximately 146 times.

Table 1: COMPARISON OF TWO ENVIRONMENTS – NOT PARALLEL

Environment	1 generation [s]	8000 generations [s]
Matlab	2.775	22198
Java RMI	0.019	156.35

Second test compare Matlab and Java environments in parallel computing. In Java environment (T_2,Z_2) was still faster but our architecture of the program shows that non-parallel computing was faster when we used one and two computers. With three and more computers we can see the advantages of the distributed computing. Matlab environment (T_1,Z_1) show advantages right away. With two computers the betterment was significant, more that 50%. From five or six computers Matlab betterment settles. On the other side Java's environment continue rise more or less equally. Java will probably settle near 80% too. In table 2 last two columns represent betterment from non-parallel approach.

Table 2: COMPARISON OF TWO ENVIRONMENTS – PARALLEL

N	T_1 [s]	T_2 [s]	Z_1 (T_{REF1}=22198s) [%]	Z_2 (T_{REF2}=156.35s) [%]
1	20820	335.83	6.21	-114.85
2	10380	193.66	53.24	-24.23
3	7980	143.96	64.05	8.24
4	5760	119.06	74.05	23.80
5	4800	106.56	78.38	31.71
6	4740	99.76	78.65	35.97
7	4620	89.40	79.19	42.76

4 Conclusion

The genetic algorithms are typical examples of the systems suitable for parallel computing. Parallel computing can accelerate the optimization process. Two environments for parallel computing were presented. Java environment was significant faster but we must write our own functions for genetic operations and distribution of the data. The parallel algorithm wasn't optimal because it was slower with one and two computers then non-parallel algorithm. On the other side Matlab Parallel Toolbox provided tools for parallel computing and we use Sekaj's genetic toolbox so writing application in Matlab environment was faster. Matlab is complex environment so the computations were slower, but betterment was significant only with two computers.

5 Acknowledgement

This work has been supported by Scientific Grant Agency VEGA 1/0544/09.

References

[1] K. Kopčok. *Rozvrhovanie operácií v pružných výrobných systémoch s využitím genetických algoritmov a heuristických prístupov*. Dizertačná práca, FEI STU, Bratislava, 2005

[2] I. Sekaj. *Evolučné výpočty a ich využitie v praxi.* IRIS Bratislava, 2005, ISBN 80-89018-87-4

[3] B. Barney. *Introduction to Parallel Computing*.
[online] https://computing.llnl.gov/tutorials/parallel_comp/, 2009

[4] M. Hudáček. *Výpočtovo náročné úlohy v Matlabe*. Bakalárska práca, FEI STU, Bratislava, 2009

[5] P. Nagy. *Distribuované výpočty pomocou Java RMI.* Diplomová práca, FEI STU, Bratislava, 2009

[6] Sun Microsystems. *The Java Tutorial*. [online] http://java.sun.com/docs/books/tutorial/, 2009

[7] The MathWorks. *Getting Started Guide*. 2008

[8] The MathWorks. *Parallel Computing Toolbox User's Guide.* 2008

[9] The MathWorks. *MATLAB Distributed Computing Server, System Administrator's Guide.* 2008

[10] Posterus.sk. Matlab Tutorials. [online] http://www.posterus.sk/?cat=7, 2009

Ing. Michal Blaho
michal.blaho@stuba.sk

Ing. Martin Foltin, PhD.
martin.foltin@stuba.sk

Ing. Igor Kristijan Kopčok, PhD.
kopcok@eprojekt.sk