ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 4 (2014) No. 4, pp. 27-44

Optimizing Production Scheduling Using Genetic Algorithm in Textile Factory

Harwin Kurniawan [1 +], Tanika D. Sofianti [1], Aditya Tirta Pratama [1], Prianggada Indra Tanaya[1]
[1] Swiss German University, BSD City, South Tangerang, Indonesia

(*Received May 2014, accepted Oct 2014*)

Abstract. This study addresses the production scheduling problem in a textile factory located in Purwakarta, Indonesia. The product completion time of the textile factory exceeds the established deadline frequently. The problem is identified from the total makespan; the total processing time from the first item enters the production line until the last item leaves the production line in the same batch that is considered to be too high. The objective of this paper is to develop an application program to generate production schedules with minimum total makespan. Genetic algorithm is used in this study to find minimum total makespan. The results was tested and proven to be reliable in generating production schedules with lower makespan compared to existing scheduling process in the same factory.

Keywords: Genetic Algorithm Scheduling, Job Shop Scheduling, Textile Scheduling

1. Introduction

Manufacturing is a process of transforming raw material into intermediate products or finished goods through several processing steps (Business Dictionary, 2014). Textile manufacturing is a process of transforming natural and synthetic fibers into fabric which go through several treatments to improve the strength and appearance. The process begins with a knitting process to transform fibers into raw fabric. The finished fabric then goes through a bleaching process to whiten the product before it is dyed to give color based on orders. Some additional processes may incur calendaring which is capable of smoothing the surface or compacting to make the surface uniform (Burrows, Cooper, Hewson, Smith, & Wilson, 1996).

Scheduling the production process in textile industries is really important as to prevent lateness, maintain low inventory level and high utilization of machine. The goals of scheduling are to minimize average cycle time, maximum lateness or makespan (Hopp &Spearman, 2008).

This study is addressed on a factory that produces fabrics in rolls quantity with more than two hundred workers and about seventy machines. The classic scheduling method in would not really effective as it is constructed without taking consideration on agreed deadline time with customers. Some orders from customers still exceed their deadlines. The production scheduling is often not followed because it causes some customers orders to exceed the previously agreed deadline. In order to maintain customer satisfaction, it is important to fulfill customer demand before the agreed dead line. One of alternative option is by using good production scheduling system. However, the current traditional production scheduling system is still far from satisfactory. It needs production scheduling system with program or application that is capable of generating a production schedule in such a manner so that no lateness or minimum lateness occurs.

Genetic Algorithm (GA) is an alternative method to manage production scheduling, an evolutionary search techniques used to identify approximate solutions for optimization problems. It represents a computer simulation of a population of abstract representation (called chromosomes) of the candidate solutions (called individuals) to an optimization problem that evolves toward better solutions (Oprea &Nicoara, 2005). Based on identified problem, the objective of this study is to develop software for production scheduling system in the textile factory by using GA for its optimization algorithm.

2. Literature Review

The literatures of GA are reviewed in this study to elaborate the process of GA for scheduling and the processes of selection, crossover and mutation for finding minimum makespan.

2.1. Genetic Algorithm

GA imitates the principle of natural evolution and selection inspired by biological mechanisms and is adapted to large area of problems (Chircu, 2010). GA shows implicit parallelism and is able to keep useful redundant information from previous generation by its representation in chromosomes of each population. Critical components of past good solutions can be maintained and combined together by means of crossover to form high quality solutions (Wang &Zheng, 2001).

The algorithm of GA starts with a complete or partial randomly generated

population. The evolution is simulated in generations. In GA, the available possible solution is lined together then by using cases inspired by natural occurrences such as: mutation and crossover, then the solution is combined and mixed together to generate the best possible solution (Mitchell, 1996) as in Figure 1. The new population is obtained from the old population by following three important steps (Oprea &Nicoara, 2005): selecting the best individuals to become parents, performing crossover on the parents to obtain new individuals and performing mutation to some very few individuals.

Individual strings are copied according to their maximum objective values or goodness. The process begins with initialization of population which generated randomly from population size. In the selection phase, two parents are selected based on their fitness values. The higher fitness level offspring has a higher probability to be chosen (Negnevitsky, 2005). In the next phase, these two parents will be mated through crossover and mutation process to generate two new offspring which carry hybrid genes from both parents.

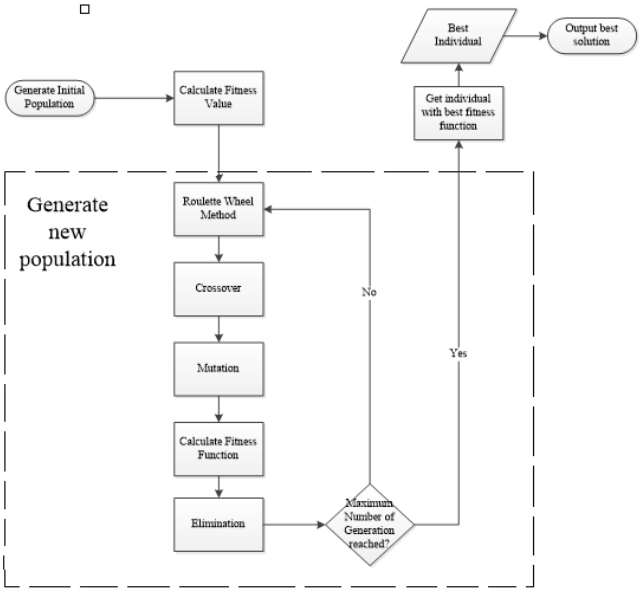

Fig. 1: Genetic Algorithm Cycle (Tamilarasi & Kumar, 2010).

2.1.1. Fitness Value

Fitness value determines the compatibility between the current solution and the desired solution. This compatibility indicates when to stop the mating process in GA. Some fitness measurements that can be used in scheduling are service level, lateness or makespan (Hopp &Spearman, 2008).

Fitness value contained in fitness function has strong correlation with objective

function. Both functions may be related directly or inversely proportional. When they are inversely correlated then some adjustment is necessary to link both functions (Sianturi, 2012). Fitness value is also used to determine which individual to be derived and which one should be eliminated in order to breed individual with best solution (Chircu, 2010). It also serves as an indicator in roulette wheel method to determine the probability value (Negnevitsky, 2005).

2.1.2. Roulette Wheel Method for Selection

The selection of individuals to be assigned as parents in the next generation is an important stage of the algorithm. Each individual in this population has attached fitness function that represents the individual performance based on a number of criteria. Based on the fitness function value attached to each candidate, the individuals are chosen to be parents in order to increase the quality of the solution (Chircu, 2010). A higher fitness level means that it is the better solution (Whitley, 1994).

The reproduction can be presented by using the roulette wheel method. In this method, each string of the population has a slot in the wheel. The size of the slot is based on its fitness level; when the fitness level is high, then the slot is also bigger. During reproduction phase, the wheel is spun based on the number of strings needed. Every time new offspring is needed, the wheel is simply spun (Goldberg, 1953).

In roulette wheel method the string with higher fitness values have bigger probability to be chosen, strings with better solution will have bigger probability to inherit its one or more offspring to the next generation. But other string also has a chance to be chosen though with smaller probability which keep the diversity of the population large enough (Bajpai & Kumar, 2010).

2.1.3. Crossover and Mutation

In crossover step, half of the content from two or more parent solutions is swapped to create a new solution (Buckland, 2004). The purpose of crossover is to create new offspring. The line where the crossover happens or two parent chromosomes are separated is randomly chosen. The results are two new offspring that are expected to be better than their parents. In mutation the chromosome of the parent is changed. The change of genes is aimed to guarantee that the algorithm search will not be trapped in a local optimum. Mutation is represented by a chromosome modification applied to one or more genes (Chircu, 2010).

The essence of GA lies in these two operations. The crossover probability or crossover rate (pc) and mutation probability or mutation rate (pm) determines whether the operations shall be performed or not. Crossover rate controls the capability of GA in reaching local optima. The higher the crossover rate the quicker it exploits the local hills. But if the process is too fast, then the individual will never

reach its local optima. Mutation rate determines the speed of GA to exploits new area. It is usually in very small value compared to crossover rate (Lin, Lee, & Hong, 2003).

3. GA for Production Scheduling

GA is frequently used for resource allocation to finish a predetermined quantity of orders under a minimal makespan. The results are represented by the production sequences. The aim is to allocate the resources that minimize the makespan for certain orders by taking system constraints into account. An individual's performance will be evaluated by his fitness function value, which needs to be minimized. The objective function minimizes the finishing time of tasks n-1 (the last task), and therefore minimizes the makespan. The function value is updated according to both the ending time on the last machine, of the last product and several restrictions on total waiting times at machines and job sequence, by applying a set of specific rules (Chircu, 2010).

To solve a problem in job shop scheduling, an initial population is randomly generated in regard of task sequence. These populations are subjected to Darwinian evolution and must repeatedly undergo crossover and mutation to finally get the fittest individual among them all. Alternatively the GA may be run for the user defined number of generations to then choose the fittest member among them. Some constraints that may be used as fitness function are: total makespan, mean tardiness, maximum tardiness and number of tardy job (Sadeghieh, 2002).

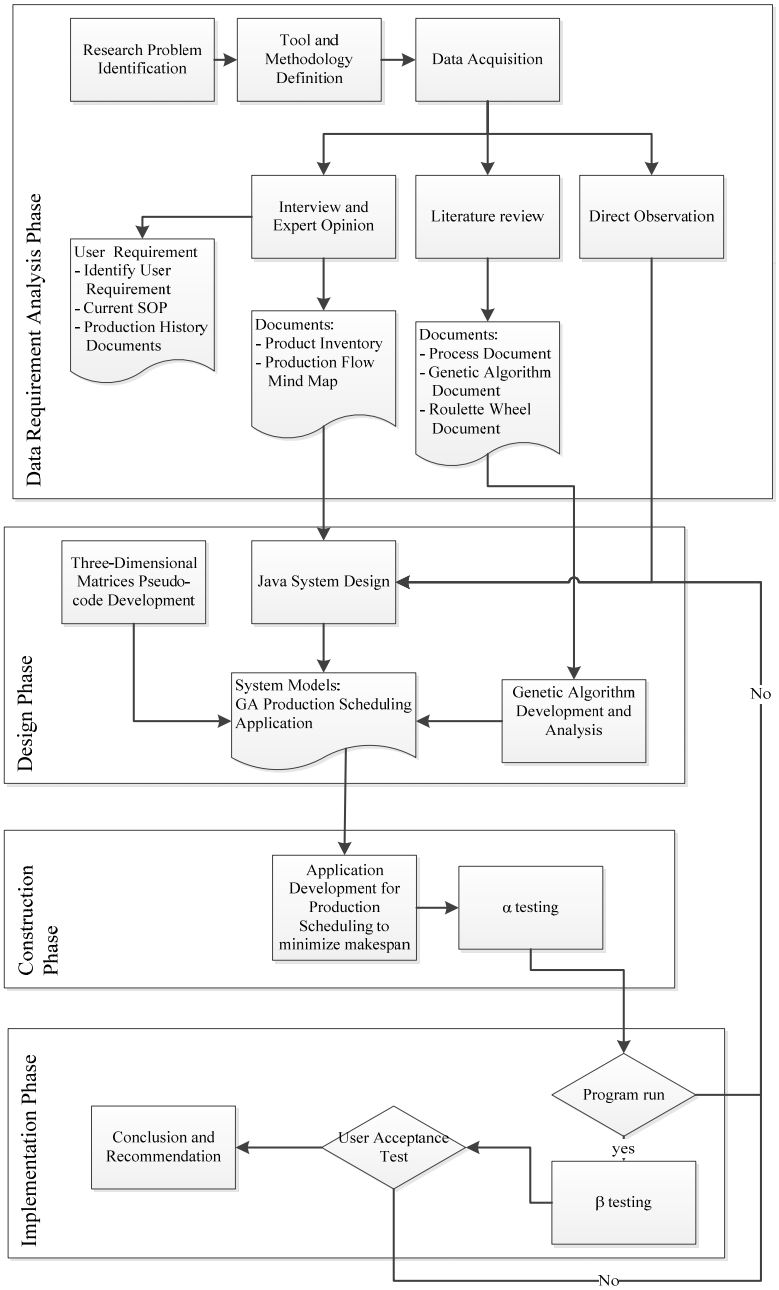

Fig. 2: Methodology of Software Development

The methodology in developing software for production scheduling is divided into four sub-categories or phases: data requirement analysis phase, planning phase, construction phase and implementation phase (Boehm et al., 1998). The overall methodology diagram is shown in Figure 2.

3.1. Production Process at the Factory

The production process at the factory is divided into two main processes, respectively: open width process and tubular process. The flow of working materials in both production processes is slightly different. Since the fabric is produced using circular knitting machine, it leaves the knitting station in tubular form (Catallo & Cohn, 1963). In tubular process the fabric remains in tubular form throughout the whole production line; while in open width process, the fabric is slit open as soon as it leaves the knitting station as to save energy and chemical(Kleinheinz, 2013).

The problem proposed a factory production line with seventy one machines. These machines are positioned in thirteen production cells based on its functions. This system ignores machines breakdown time and transportation time between stations. However, this system takes setup time into account during scheduling. The available tasks for all the stations are described in Table 1.

Tab. 1: Production Station List.

Process Name	Process Location	Processing Time	Setup Time	Station
Knitting	Knitting machine	1.5 hours	10 – 12 hours	Knitting
Greige Inspection	Grey inspection machine	8 minutes	-	Greige Inspection
Preparation	Preparation machine	8 minutes	-	Preparation
Presetting	Pre-set & finishing machine	15 minutes	0.5 - 1 hour	Pre-set & Finishing
Singeing	Singeing machine	5 minutes	30 minutes	Singeing
Dyeing	Dyeing machine	7-12 hours	15 minutes	Dyeing
Slitting /dewatering/Opening	Opening machine	7 minutes	30 minutes	Slitting
Squeezing	Squeezing machine	7 minutes	30 minutes	Squeezing
Drying	Drying machine	10 minutes	30 minutes	Drying
Inspection	Inspect after dye	10 minutes	-	Inspection
Finishing(open width)	Pre-set & finishing machine	10 minutes	0.5 – 1 hour	Pre-set &Finishing
Compacting	Compacting machine	10 minutes	30 minutes	Compacting
Heat Setting	Heat setting machine	10 minutes	30 minutes	Heat Setting

As previously explained, the production line is divided into two main types of process: open width and tubular. The tasks included in both processes are represented in sequence in Table2.

Tab. 2: Production Flow for Open width and Tubular Process.

Process Type	Tasks

Open width	t_1 -t_2 -t_3 - t_4 -t_5 -t_2- t_6–t_7 – t_9 -t_{10} - t_{11} -t_{12}-t_{10}
Tubular	t_1 -t_2 -t_3 -t_2- t_6– t_7 –t_8 – t_9 -t_{10} - t_{12} /t_{13}-t_{10}

Since the production flow is diverse for every products then the process is categorized as a job shop process. In a job shop each batch has a unique time to finish every product and certain method must be used to predict the time. This makes job shop more difficult to be scheduled since each product needs different production time and tools (Hofmann, 2013).

The objective of job shop scheduling in this study is to produce n jobs in m machines with minimum completion time for all jobs. Each job consists of t tasks that need to be done on certain machines without interruption (Sun, Cheng, & Liang, 2010).

In a factory, orders need to be released before the due date and transformed into jobs. The jobs then have to be worked in by certain machines in a certain sequence and sometimes it has to wait for other jobs in queue. Sometimes there is a high priority job that has to be processed at once. Production scheduling is an important tool to get the operation in order (Pinedo, 1995).

Chircu (2010) also has a same opinion as he explained: a production plan consists of n jobs, and each job consists of mi tasks, each of them having to be processed by a single machine. The completion time of a task can be obtained by adding the processing time to its starting time (Chircu, 2010).

The processes usually have setup time which is different for different job, different due dates and release dates as well as different constraints. The problem's constraints are represented by the operation sequence that is different for each product and by the fact that the machine can operate only one product at the time. The objective is to determine a schedule for the tasks on the machines in minimum time (Chircu, 2010).

The general framework used in this study based on several jobs and machines; the jobs have either identical or different processing times on the given machines. Production scheduling is responsible in assigning jobs to machines so that the completion time or makespan is minimized. The order of the jobs to be executed is completely random.

In classic makespan scheduling each product must visit the machines in a given sequence, but the sequence is different for each products type. For example let t = {0, 1, …, n-1} be the set of task to be scheduled and M = {1,..., m} the set of machines and j is the sequence of the task. The tasks are organized by two constraints (Lourenço, 1994): (1) each task (tj) of a job must be scheduled after all the predecessor tasks in that order are completed (the precedence constraints), and (2) the task j can only be scheduled if the machine it requires is idle. In other words, one machine can only process one task at a time. The objective is to minimize the

makespan".

Determining minimum makespan is an NP-hard (Non-deterministic Polynomial-time hard) problem. That is no currently available computer machine which able to give exact answer to the problem and even the simplest problem must be resolved in polynomial time(Garey &Johnson, 1979). There is no currently exact method that can be used to really fix the problem. One of the promising recently developed methods is from artificial intelligence area with focus in meta-heuristic, including GA(Sun, Cheng, & Liang, 2010).

3.2. Individual Form

The form of individual will be in two dimensional matrices (Dean, 2008). The array row represents the production stations while the array column contains the order numbers. The individual chromosome contains information of sequence of orders to be executed at every production station; the order number at the left side of the column means that order will be executed first.

The objective function of this scheduling is minimum total makespan of the whole orders, the smaller the value of total makespan gives the better result. Sometimes, the fitness function may require a little modification from the objective function (Sianturi, 2012). Since the objective function needs to be minimized, while the fitness function needs to be maximized, then equation 3.1 may be used to help linking both functions.

$$\boldsymbol{F(x) = \frac{1000}{1+f(x)'}} \qquad (3.1)$$

Where:

F(x) = fitness function.

f(x)' = the objective function.

At the beginning the application will produce several parents whose chromosomes are randomly taken from the task matrix. Based on its fitness function, some parents with low fitness value will be eliminated and will not be used to breed new offspring.

The genetic operators used in this application are as following:

- Roulette wheel selection
- Uniform crossover
- Rotation mutation

Roulette wheel selection will randomly choose the parents for the next generation by using the roulette wheel method. The probability value for certain parents to be chosen is based on its fitness value. Random number will be generated between zero and total fitness value of all available individuals to determine which parent to be mated. When the generated number is in the range of certain individual then the

individual will be chosen.

Uniform crossover uses chromosome for each i ∈ {1,…., c) to determine the crossover. The crossover exchanges the genes between two parents in expectation to generate better individual. Crossover occurs only with a probability in crossover rate or crossover probability. When the individual is not subjected to crossover then it remains unchanged(Lin, Lee, & Hong, 2003).

Fig. 3: Uniform Crossover.

When the crossover is performed a dividing line will be randomly generated in the middle of the individual. The left side of chromosomes from parent1 will be copied to off1 and parent2 to off2, and the right side of chromosomes from parent1 will be copied to off2 and parent2 to off1. To put it simple the chromosomes behind the dividing line will be swapped between two parents as depicted in Figure 3.

The rotation mutation performs a small chromosome modification by inversing the gene order. This modification is made by considering the restrictions imposed by the problem. Like in crossover process, the mutation rate determines whether the mutation is performed or not. Mutation helps add genetic diversity to prevent local optimum trap (Lin, Lee, & Hong, 2003). The example of individual form can be seen in Table 3.

Tab. 3: Individual Form

Knitting	p1OW-123456	p4Tu-234567

Greige Inspection A	p1OW-123456	p4Tu-234567
Preparation	p1OW-123456	p4Tu-234567
Preset	p1OW-123456	null
Singeing	p1OW-123456	null
Greige Inspection B	p1OW-123456	p4Tu-234567
Dyeing	p1OW-123456	p4Tu-234567
Opening - Tubular	p4Tu-234567	null
Opening - Open Width	p1OW-123456	null
Drying - Tubular	p4Tu-234567	null
Drying - Open Width	p1OW-123456	null
Inspection - Tubular A	p4Tu-234567	null
Inspection -Open Width A	p1OW-123456	null
Finishing	p1OW-123456	null
Compacting - Tubular	p4Tu-234567	null
Compacting - Open Width	p1OW-123456	null
Heat Setting	null	null
Inspection - Tubular B	p4Tu-234567	null
Inspection - Open Width B	p1OW-123456	null

Each row in the chromosome represents the available in the workstations listed in Table 1.The upper tasks are performed before the tasks underneath. Each task contains the orders that need to be done in the respective task. The order in the left side of the chromosomes means that the order enters the workstation before the orders on the left. The orders are represented in the format as follow: "pxxx-xxxxxx". The first two digits represent the product code. While the next two digits represents the process type: 'OW' for open width and 'Tu' for tubular. The last six digits of represents the order number.

4. Performance Testing

The crossover rate and mutation rate used is vary between every problem and is determined through trial and error (Lin, Lee, & Hong, 2003). The typical value of pc is between 0.5 and 1.0; while pm is between 0.001 and 0.05.In addition of determining the value of crossover and mutation rate, it is also necessary to specify number of population and maximum number of generation. Different combination of these parameters will cause different results. If the values of both parameters are too small, then the population will not be able to have enough time to evolve which will generate poor results. In contradictory, when the value is too big the calculation time will be too long and not effective (Dean, 2008). Table 4 shows the test result.

Tab. 4: Parameter Test Result.

Parameter Number	Generation Number	Population Size	Running time (seconds)	Best Performance	Worst Performance at Last Generation	Total Makespan (days)	Number of Variants
1.	10	20	10.23	0.03767	0.01518	18.44	16
2.	15	20	13.67	0.03777	0.01563	18.39	17
3.	30	20	22.93	0.03786	0.01570	18.35	25
4.	10	50	19.50	0.03804	0.01430	18.28	27
5.	15	50	28.43	0.03789	0.01404	18.33	30
6.	30	50	73.30	0.03774	0.01407	18.41	30
7.	10	80	31.23	0.03812	0.01361	18.23	51
8.	15	80	50.67	0.03800	0.01353	18.28	45
Parameter with the best result	-	-	1	7	3	5	7

Currently the parameter setting for this application is crossover rate (pc) 0.60, mutation rate (pm) 0.33 and crossover type of crossover array. However, the population number and generation number is still not clear. In order to get best parameter, correlation test and discriminant analysis are also need to be conducted. Correlation analysis concludes that the best parameter setting occurs in the parameter number 7 by using generation number (gen) of 10 and population number (pop) of 80. Crossover rate (pc) 0.60, mutation rate (pm) 0.33 is selected. The significance level of running time is far below 0.02 which means a change in generation number and population size would highly affect the total running time. Meanwhile the significance level of total makespan is above 0.02 which has small effect on the result when the generation number and population size is changed. The results support the hypothesis, based on those two analyses; it is suggested not to increase the generation number as it could make the total makespan even higher, it would be better to increase the population size instead. The application program is tested to find its performances. The result of the test can be found in Table 5, and the comparison of the result obtained from application program using GA and the existing methodology used in the factory can be seen in Table 6.

Tab. 5: GA Scheduling Test Result.

No.	Order No.	Quantity (rolls)	Process Type	Production Time in GA (days)	Real Production Time (days)
1	042270	4	Tubular	15	19
2	042272	6	Tubular	18	19

3	042274	2	Tubular	17	13
4	042271	3	Tubular	15	14
5	042273	7	Tubular	16	25
6	042276	1	Tubular	16	3
7	042278	1	Tubular	18	3
8	042279	1	Tubular	17	3
9	042277	2	Tubular	18	6
10	042284	12	Open width	18	23
11	042283	1	Open width	15	25
12	042285	47	Open width	18	17
13	042286	48	Open width	17	26
14	042289	71	Tubular	15	13
15	042290	47	Tubular	16	13
16	042291	44	Tubular	16	13
17	042292	28	Tubular	18	26
18	042293	72	Tubular	17	26
19	042294	48	Tubular	18	32
20	042295	72	Tubular	18	26

Tab. 6: GA and Existing Scheduling Method Comparison

	Scenario 1		Scenario 2		Scenario 3	
	Using GA Application	Using Traditional Schedule (manual)	Using GA Application	Using Traditional Schedule (manual)	Using GA Application	Using Traditional Schedule (manual)
Time to Find Solution (average)	0.0087 hrs. (31 sec)	6 hrs.	0.0058 hrs. (24 sec)	6 hrs.	0.0072 hrs. (26 sec)	6 hrs.
Total Makespan (average)	18 days	32 days	21 days	25 days	17 days	29 days
Days Saved	43.8 %		16 %		41.38 %	

5. Conclusion

Based on the test result in Table 5, the production schedule created by this application results in better total makespan compared to the schedules created by traditional method. However there are some orders whose finish date is slightly longer compared to traditional scheduling method. The whole orders are finished before the agreed deadline with the customers.

Based on comparison between scheduling with GA and existing method depicted in Table 6 it is concluded that the schedule produced by GA is better than one produced by manual scheduling, although some of the order has slightly longer estimated finish time. In general, the estimated finish time produced by the program for production scheduling using GA is better than one that produced by manual scheduling. The other advantage is that the entire schedule does not exceed the agreed deadline.

Table 6 shows that total makespan of the whole schedule produced by the program is about 40%shorterthan the one that manually scheduled. It means that the production process finish in about two weeks earlier than the schedule produced manually.

The manual scheduling took about six hours and need more than one person to determine the production schedule. By using the application program developed in this study, only one operator is needed and it takes less than a minute to generate the production schedule. The GA implemented in the scheduling program is able to reduce the order delay in the factory, thus it improves customers' satisfaction.

References

Bajpai, P., & Kumar, M. (2010). Genetic Algorithm – an Approach to Solve Global Optimization Problems. *Indian Journal of Computer Science and Engineering*, 199-206.

Boehm, B., Egyed, A., Kwan, J., Port, D., Shah, A., & Madachy, R. (1998). Using the WinWin Spiral Model: A Case Study. *Computer*, 33 - 44.

Buckland, M. (2004, November). Ai-junkie programming. Retrieved from *www.ai-junkie.com*

Burrows, R., Cooper, P., Hewson, M., Smith, C., & Wilson, H. (1996).

Business Dictionary. (2014). Retrieved from *http://www.businessdictionary.com/definition/manufacturing.html.*

Catallo, F., & Cohn, S. (1963). Apparatus for converting tubular knitted fabric to open width form. *New York*.

Chircu, F. A. (2010). Using Genetic Algorithms for Production Scheduling. Ploiesti: *Petroleum-Gas University of Ploiesti, Informatics Department*.

Dean, J. S. (2008). Staff Scheduling by a Genetic Algorithm with a Two-Dimensional Chromosome Structure. Parkville: *Park University, Information and Computer Science Department.*

Garey, M. R., & D. S. Johnson. (1979). Computers and Intractability: A Guide to the Theory of NP-Completeness. *W.H. Freeman*,

Goldberg, D. E. (1953). Genetic Algorithm. *Addison Wesley Longman*.

Goldin, D. (2013, August 2). dangoldin.com. Retrieved from *http://dangoldin.com/2013/08/02/a-brief-history-of-manufacturing/*

Hofmann, L. (2013). Retrieved from *http://hofmann.tcnj.edu/courses/360%20ppt/20 Scheduling.ppt*

Hopp, W. J., & M. L. Spearman. (2008). Factory Physics Third Edition. New York: *MC Graw Hill*.

Kleinheinz, J. (2013). Open-width treatment of knitwear. *Pakistan Textile Journal*, 54-55.

Lin, S.-C., Goodman, E. D., & Punch, W. F. (1997). A Genetic Algorithm Approach to Dynamic Job Shop Scheduling Problems. *East Lansing: Michigan State University*.

Lin, W. Y., Lee, W. Y., & Hong, T. P. (2003). Adapting Crossover and Mutation Rates in Genetic Algorithm. *Journalof Information Science and Engineering*, 889-903.

Lin, W. Y., Lee, W. Y., & Hong, T. P. (2003). Adapting Crossover and Mutation Rates in Genetic Algorithm. *Journalof Information Science and Engineering*, 19, 889 - 903.

Lourenço, H. R. (1994). Local Optimization and the Job Shop Sheduling Problem. *European Journal of Operational Research*.

Mitchell, M. (1996). An Introduction to Genetic Algorithm. Cambridge: *MIT Press.*

Negnevitsky, M. (2005). Genetic Algorithm, artificial intelegence. *Addison Wesley*.

Oprea, M., & S. Nicoara. (2005). Artificial intelligence. Ploiesti: *Petroleum–Gas University of Ploiesti*.

Pinedo, M. (1995). Scheduling, Theory, Algorithm and System. New jersey: *Prentice-Hall, Inc.*

Sadeghieh, A. (2002). Optimization in spreadsheet model for network problem. *Far East J. Math. Sci.*, 4, 100-115.

Sianturi, A. L. (2012). Optimasi Penjadwalan Karyawan Pembangunan Kapal dengan Mengunakan Algoritma Genetika. Depok: *Universitas Indonesia*.

Sun, L., Cheng, X., & Liang, Y. (2010). Solving Job Shop Scheduling Problem Using Genetic Algorithm with Penalty Function. *International Journal of Intelligent Information Processing*.

Tamilarasi, A., & kumar, T. A. (2010). An Enhanced Genetic Algorithm with Simulated Annealing for Job-shop Scheduling. *International Journal of Engineering, Science and Technology,* 2(1), 144-151.

Tanjaya, I. C. (2013). the computerized distribution of route planning of Pt Lea Sanent using genetic algorithm and analytical hierarchy process. *South Tangerang: Swiss German University*.

Wang, L., & D. Z. Zheng. (2001). An elective hybrid optimization strategy for job-shop scheduling problem. *Computers & Operations Research*, 28, 585 - 596

Whitley, D. (1994). A Genetic Algorithm Tutorial.