Scheduling of Worker Allocation in the Manual Labor Environment with Genetic Algorithm

Sicong Tan, Wei Weng, and Shigeru Fujimura

***Abstract*—Worker allocation problem of the manual labor factory consists in deciding who does what during the manual labor production. Manual labor factory has concern for reducing the production duration, human cost and overwork. A novel scheduling model of worker allocation problem for manual labor is proposed in this paper. Given a set of jobs and a set of workers, worker allocation problem is to assign the workers into these jobs to reduce human cost, to shorten production duration and control production overwork. Worker allocation problem is a very important issue in the production scheduling, since total budget and human resource involved must be managed optimally. Worker allocation problem is NP-Hard problem to find optimal solution as increasing worker number and enlarging production scale. Genetic algorithm (GA) is considered a promising method so in this paper worker allocation problem is solved by using genetic algorithm. This paper shows it is very flexible and accurate for finding near-optimal solutions in the short time.**

***Index Terms*—Worker allocation, genetic algorithm, manual labor production scheduling.**

I. INTRODUCTION

One problem manual labor factory must face is how to allocate human resource optimally to guarantee manual labor production more efficiently. The high complexity of worker allocation problem demands this research into computer aided tools to properly assign workers into jobs. Current manual labor production needs to control people and processes and allocate resources in order to achieve specific objectives while satisfying a variety of constraints. In short, worker allocation problem consists in defining who are assigned to perform which job. In manual labor factory, the objective is to minimize the production duration, human cost and production overwork, at the same time to maximize the production quality. But these goals are in conflict with each other. In the real factory, the production managers usually allocate workers to each job according their job experience. So they may delay the production duration or waste much money due to their random assignment. The manager wants an automatic scheduling which will reconcile these conflicting goals as far as possible.

To solve these problems, scheduling problem with worker allocation (SPWA) was presented, in Iima's paper, workers skills level to each machine is all the same [1].However in fact each worker has a different skill level for each machine. So Osawa proposed a new model of SPWA consisting of the following three new procedures: shortening of ideal time, modifying infeasible solution to feasible solution and a new selection method of GA [2]. In recent years, modified scheduling problem with worker allocation is presented by Iima, in his proposal, a concept of Module Type GA (MTGA) was proposed, and a design of MTGA is proposed for solving modified scheduling problems with the worker allocation [3] .All of these are related to worker allocation for machines. However, there hardly exists research on the scheduling problem with worker allocation for manual labor.

This paper focuses on the worker allocation for manual labor and proposes a new model which divides the whole production process into several jobs and assigns workers into these jobs according to their skills requirement.

This paper is organized as follows. In Section II the worker allocation problem is defined. Section III describes the genetic algorithm proposed and discusses the representation of the individuals. The experimental study and results are presented in Section IV. Finally, conclusions and future work are outlined in Section V.

II. SCHEDULING PROBLEM WITH WORKER ALLOCATION FOR MANUAL LABOR

Scheduling problem with worker allocation for manual labor (SPWAML) considered in this research can be described as follows. The factory is able to manufacture PQ pieces of one product. PQ denotes the production quantity of it. For producing, there is a need to employ some workers who have skills which are necessary for the production.

A. Worker model

The resource considered is worker who is represented by a numerical identifier, a set of skills, and a salary. These workers have an effort for each job. The effort represents the degree of dedication for one job. Formally, each worker is denoted with w_i where i ranges from 1 to W (i denotes the numerical identifier of a worker). The workers have a set of skills. Let SK be the set of skills, and s_n denotes the n th skill with n varying from 1 to N. The skills of worker w_i will be denoted with $w_i^{skills} \subseteq SK$, and each worker has a salary. The hourly salary of worker w_i denotes with w_i^{salary} .

Manuscript received December 10, 2008.

Sicong Tan is with Graduate School of Information, Production and Systems, Waseda University 2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka, 808-0135 Japan (phone: +81-80-3963-1598; e-mail: danshisou@toki.waseda.jp).

Wei Weng is with Graduate School of Information, Production and Systems, Waseda University 2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka, 808-0135 Japan (e-mail: wengwei@toki.waseda.jp).

Shigeru Fujimura is with Graduate School of Information, Production and Systems, Waseda University 2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka, 808-0135 Japan (e-mail: fujimura@waseda.jp).

It is supposed that a garment factory receive an order which needs to manufacture a lot of embroidery clothes. This factory plans to assign five workers to finish the clothes order. In Fig (a), the different skills of the workers and their hourly salaries are shown. For example, in order to make these clothes, the workers must have five kinds of skills: cut skill, embroidery skill, sew skill, decorating seaming and iron skill. Also for example, worker w_1 who earns \$1000 every hour ,has cut skill and sew skill.

Fig(a): an example of worker information

$w_1^{skill}=\{s_1,s_3\}$ $w_2^{skill}=\{s_2,s_4\}$ $w_3^{skill}=\{s_2,s_3\}$ $w_4^{skill}=\{s_5\}$ $w_5^{skill}=\{s_1,s_4,s_5\}$

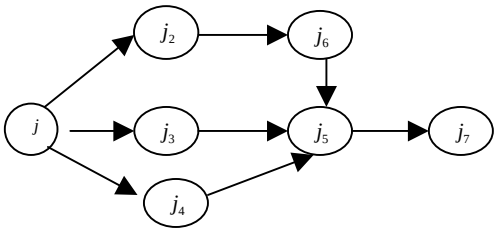
$w_1^{salary}=\$1000$ $w_2^{salary}=\$800$ $w_3^{salary}=\$1200$ $w_4^{salary}=\$700$ $w_5^{salary}=\$1500$

$SK=\{s_1,s_2,s_3,s_4,s_5\}$ s_1: cut skill s_2: embroidery skill

s_3 :sew skill s_4: decorating seaming s_5: iron skill

B. Job model

To plan to manufacture products, the production process is divided into several jobs. Job is modeled as following. A job is represented by a numerical identifier and a set of properties. These properties are described in the following. Jobs are denoted with j_m where m ranges from 1 to M (m denotes a numerical identifier of jobs). Each job j_m has a set of required skills associated with it denoted by j_m^{skill} and each job has production speed j_m^{speed}. Production speed represents how many products can be produced by one worker in an hour .To assign jobs, we need to know the precedence relation of jobs firstly .The jobs must be performed according to a Job Precedence Graph (JPG). A JPG indicates which jobs must be completed before a new job begins. The JPG is an acyclic directed graph G (V, A) with a vertex set $V=\{j_1, j_2.....j_M\}$ and an arc set *A*, where $(j_n, j_m) \in$ A if job j_n must be completed, with no other intervening jobs, before job j_m can start. Let us continue the garment factory example. The production sequence of these jobs is as below. j_1 :cut out a clothes , j_2 :sew the bosom of clothes, j_3 :sew the sleeve of clothes , j_4 :make hats, j_5 : seam clothes , j_6 : do embroidery, j_7 : iron the clothes



$j_1^{skill}=\{s_1\}$ $j_2^{skill}=\{s_3,s_4\}$ $j_3^{skill}=\{s_3,s_4\}$

$j_1^{speed}=20$ $j_2^{speed}=25$ $j_3^{speed}=30$

$j_4^{skill}=\{s_1,s_3\}$ $j_5^{skill}=\{s_3\}$ $j_6^{skill}=\{s_2,s_4\}$ $j_7^{skill}=\{s_5\}$

$j_4^{speed}=20$ $j_5^{speed}=35$ $j_6^{speed}=15$ $j_7^{speed}=25$

Fig(b): Job precedence graph of manufacturing clothes

The information of these jobs which include production speed of each job and the set of required skills are provided. For example job j_1 needs cutting out the clothes in order to be used later by the workers, so this job needs to be assigned to workers with cut skill. And the cut speed is 20 pieces per hour per person. In Fig (b), the job precedence graph of the production is shown. According to this graph, the production sequence can be shown, such as after job j_1 cutting out clothes is completed, job j_2 sewing the bosom of clothes, job j_3 sewing the sleeve of clothes, and job j_4 making hats can be started.

C. Variable definition

Now the element of a solution to the worker allocation problem is described. A solution can be represented with a matrix of size $W \times M$ where $x_{im} \geq 0$. The element x_{im} is the effort of worker w_i for job j_m. The effort required to complete a job, measured in person-hours, must be known before workers can be assigned to the job .Then the work can be scheduled. If worker w_i performs job j_m with a 0.5 effort, he spends half of his working day on this job. If a worker does not perform a job, he will have an effort of 0 for that job. If the worker has an effort of 1.0 for one job, he spends one working day on this job. The effort of a worker for job is used to compute the duration of each job. And the starting and finishing time of each job can be calculated, i.e., the time schedule of the jobs (Gantt diagram) can be made.

A solution is expressed as a matrix whose element x_{im} (denotes the effort of worker to job) has the value between 0 and 1, i.e., $x_{im} \in [0,1]$.The assignment of personnel to job is constrained to areas that are essential to the prototype. The percent of a worker's labor that can be committed to any given job is constrained to a discrete set of values. The commitment quota is constrained to the set {0, 0.2, 0.4, 0.6, 0.8, 1} for each worker effort that can be assigned to any single job. An example of a solution which allocates the workers to jobs is given as fig(c).

	j_1	j_2	j_3	j_4	j_5	j_6	j_7
w_1	0.8	0.8	0.4	0	0.6	0.8	0.2
w_2	0.6	0.8	0	0.6	0.2	0.6	0.8
w_3	0	0.6	0.8	1	0.8	0	0.2
w_4	0.2	0.2	0.8	0.6	0	1	1
w_5	0.8	0.6	0	1	0.4	0	0.8

Fig(c): a solution of assigning workers to jobs

D. Objective function

The objective is to minimize the human cost, the production duration and production overwork. In order to evaluate quality of a worker allocation solution, how to calculate production duration, human cost and overwork

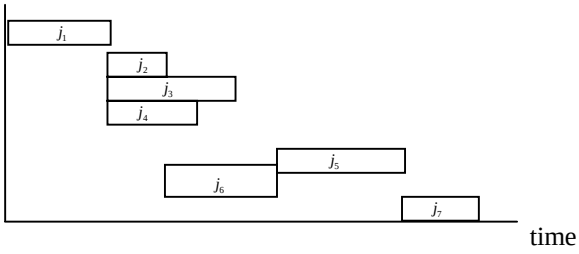
should be thought firstly. To get the production duration, denoted by $p^{duration}$, the duration of each job should be calculated firstly, denoted by $j_m^{duration}$. It can be calculate as below.

$$j_m^{duration} = \frac{PQ}{\sum_{i=1}^{w} x_{im} * j_m^{speed}} \quad (1)$$

For example, the garment factory plans to manufacture 1500pieces clothes. According to the worker allocation solution as Fig(c), as the sum of efforts of all workers for job j_2 is 3.0 and the production speed of job j_2 is 25 pieces per hour per person .The duration of job j_2 is calculated as following.

$$j_2^{duration} = \frac{1500}{3.0*25} = 20$$

According to the duration of each job, the starting time and the finish time of each job are calculated. The production can be scheduled as Fig (d). The production duration which is the maximum finish time of the production also can be calculated.



Fig(d): production schedule

The human cost denoted by w^{cost} is the sum of fees paid to the workers for their efforts on production. These costs are computed by multiplying the salary of the employee by the time spent on the production. The time spent on the project is the sum of the efforts multiplied by the duration of each job. The human cost can be computed as below.

$$w^{cost} = \sum_{i=1}^{W} \sum_{m=1}^{M} w_i^{salary} * x_{im} * j_m^{duration} \quad (2)$$

Finally, in order to compute the overwork $p^{overwork}$, there is need to compute the working time (denoted with w_i^{work}) of each worker previously. For each worker w_i, maximum working time as $w_i^{max ed}$ is defined. If his working time is larger than the maximum working time, his overwork (denoted with $w_i^{overwork}$) is calculated by the working time subtract his maximum working time .If his working time is smaller than the maximum working time, his overwork is 0. And the production overwork is calculated as (denoted with $p^{overwork}$)the sum of overwork of all workers.

$$w_i^{work} = \sum_{m=1}^{M} x_{im} * j_m^{duration} \quad (3)$$

$$w_i^{overwork} = \max(w_i^{work} - w_i^{max ed}, 0) \quad (4)$$

$$p^{overwork} = \sum_{i=1}^{W} w_i^{overwork} \quad (5)$$

In order to find feasible solutions, it is checked whether all jobs can be completed. If no worker is assigned to a job, the job couldn't be completed. If the skills of assigned workers for a job can't satisfy skill requirements, the workers can't finish this job. So there are the constrains that each job must be performed by at least one worker; Moreover the set of required skills of a job must be included in the union of the skills of the workers performing the job. It must be checked firstly that all jobs will be performed by somebody, i.e., no job is left undone. That is:

$$\sum_{i=1}^{w} x_{im} > 0 \quad \forall m \in \{1, 2, ..., M\} \quad (6)$$

The second constraint of a feasible solution is that workers performing one job must have the skills required by that job:

$$j_m^{skill} \subseteq \bigcup_{\{i | x_{im} > 0\}} w_m^{skill} \quad \forall m \in \{1, 2, ..., M\} \quad (7)$$

Now the fitness function used in the genetic algorithm is discussed. Firstly, a solution defined is as a matrix whose elements x_{im} is less than 1 and $x_{im} \in [0,1]$ (actually assigned as discrete number from {0, 0.2, 0.4, 0.6, 0.8, 1}).From it, fitness value is calculated.

The objective is to minimize production duration, human cost, and production overwork. According to this objective, the fitness function can be defined as below.

$$fitness = 1/f \quad (8)$$

Here $f = v_{duration} * p^{duration} + v_{cost} * w^{cost} + v_{overwork} * p^{overwork}$ (9)

The term is the weighted sum of the production duration, human cost and production overwork. In this term, $v_{duration}$, v_{cost} and $v_{overeork}$ are values weighting the importance of the three objectives. The minimum value of the term f can be found. i.e., find maximum fitness value.

III. GENETIC ALGORITHM

Generally speaking, the scheduling problem is NP complete, which means there are no known algorithms for finding optimal solutions [4]. Exhaustive search methods can be used to solve scheduling problems, but they require forbiddingly long execution times as the problem size increases. In this paper, a heuristic method is proposed, using genetic algorithm to solve above scheduling problem.

GA is a particular type of evolutionary algorithms introduced in the 1970s by John Holland [6]. Such algorithm attempts to simulate natural evolution and form a populous selection possible solutions by exploring the search space in an attempt to find near-optimal solutions to optimization problems. They do not exhaustively search the entire solution space, but instead start with a small population of possible solutions and then evolve towards better solutions. In the end, fairly good though not necessarily optimal solutions for the optimization problem will be produced. The outline of the algorithm is shown below:

1. Find representations of possible solutions, called genomes, which are composed of genes chosen so that by changing the genes, all possible solutions can be represented.
2. Map each genome to a real number using a fitness function. The goal is to find a genome that maximizes the fitness function.
3. Randomly generate a set of genomes called the initial

population.
4. Repeat the following steps until a stopping criterion is met:
(a) Selection: select a subset of the population for use in the next steps.
(b) Mutation: produce a new genome from an older one by changing some of its genes.
(c) Crossover: produce a new genome by combining genes from two older ones.
(d) Combination: build a new population using some of the old and new genomes.
Some examples of stopping criteria are:
(a) A certain number of generations have been generated.
(b) The best individual in the current generation has a high enough fitness value.
(c) The identity of the best individual has not changed for a number of consecutive generations. GA does not guarantee the discovery of the global optimum.

However in many applications, a near-optimal solution is often acceptable. The representation chosen for the genome is pivotal to the performance of GA [7].

Now it is explained how to use genetic algorithm solving the worker allocation problem. Firstly, the individuals must be initialized. After creating an initial set of solutions, GA can apply a crossover operation to combine the contents of two parents forming a new solution. This will be modified later by the mutation operation which alters some of the contents of the individual. Not all the individuals participate in the reproduction, only the fittest ones are selected from the population by a selection operator such as binary tournament (each parent is selected as the best of two randomly taken individuals). The operators are applied in a stochastic way, thus each one has an associated probability of application in the iterative loop, each step is called a generation. Usually, the best individuals in the present and the newly created generation are combined in order that the best ones can be retained for use in the next step of the algorithm [5]. Fig (e) describes how to use genetic algorithms to solve worker allocation problem as the following flow chart.

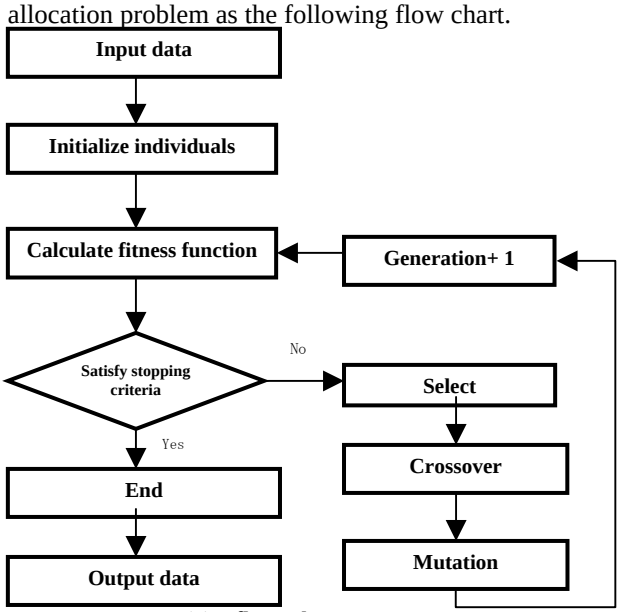


Fig (e): flow chart

The crossover operator can be operated as Fig (f) shows. The two parent solutions create two new offsprings by combining their genes array randomly. Crossover uses both inheritance and variation to improve the performance of the population, while remaining a diverse population

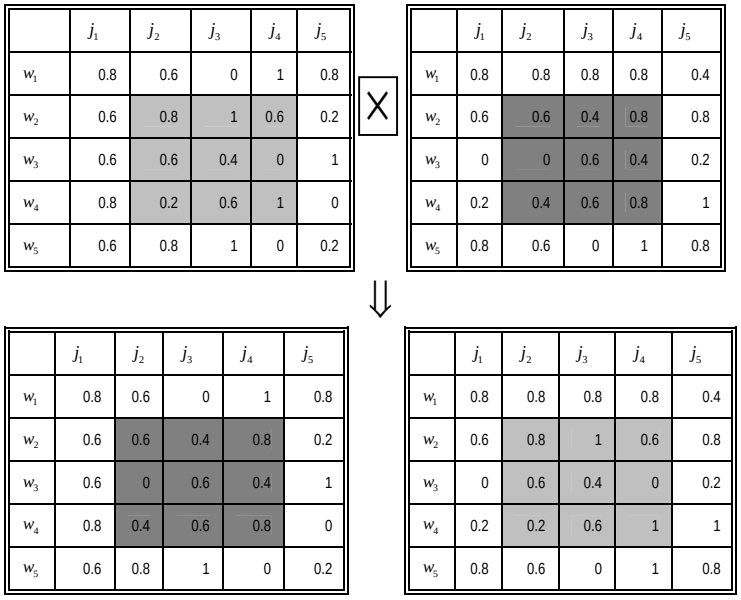
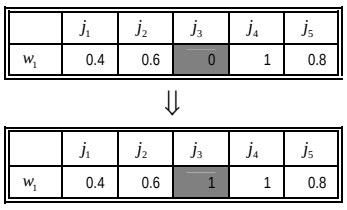


Fig (f): crossover method

The offsprings may be mutated by the mutation operator as Fig (g) illustrates. This is done by modifying a set of values used to encode the offspring. Similar to random mutations in the biological world, this function is intended to preserve the diversity of the population, so as to expand the search space into regions that may contain better solutions.



Fig(g): mutation method

IV. Experimental Study and Results

The worker allocation model is implemented in java language using JGAP which is a relatively new toolkit of Genetic Algorithm. It provides basic genetic mechanisms that can be easily used to apply evolutionary principles to problem solutions. Several experiments were conducted to evaluate the correctness and performance of the worker allocation model.

To implement the worker allocation model, firstly fit parameters are turned such as the population, crossover rate, mutation rate and generation. Because these parameters not only influence the time required to perform GA but also affect the quality of the result. The influence of parameter on the near-optimal solutions are examined in order to determine if the algorithms exhibited any unexpected behaviors, as well as to determine the tests scale. According to the test, genetic algorithm parameters are set as Fig(h) shows.

Population	64
Crossover rate	0.35
Mutation rate	0.64
Generation	2000

Fig(h): genetic algorithms parameter

Sufficient experiments have been conducted. One consisted of 10 jobs for which 10 workers are available. The workers, in turn, each possess at least one skill. Each of the five skills is needed for at least one task and many tasks required multiple skills.

It can find feasible and near-optimal solutions immediately. In particular, it can be seen that the Genetic Algorithm always vastly improved the fitness of the solution from the beginning to the end, and the final solution can satisfy the skill restrictions. The detailed scenario and the results of this experiment are given as follows.

Simulate to manufacture 10000 pieces of products. The production process is divided into 10 jobs and 10 workers is assigned into these jobs. The related job information is shown in Table 1. The worker properties are shown in Table 2. The job precedence graph is shown in Fig(i). The algorithms using composite objectives which focus on finding satisfying solutions can be evaluated as below.

Table 1: job information

Job ID	Speed	Skill Number	Skill
j_0	20	2	0,3
j_1	20	2	0,4
j_2	30	2	1,2
j_3	40	2	1,2
j_4	30	2	2,4
j_5	20	2	0,3
j_6	30	3	0,2,3
j_7	20	1	3
j_8	40	3	0,1,3
j_9	20	2	1,4

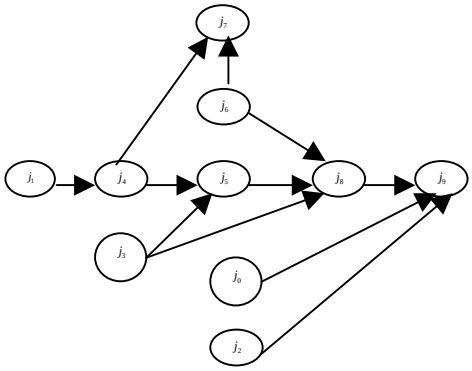
This proposal is tested according to above information. The objective is to find the optimum fitness which satisfies the minimum human cost, production duration and production overwork. According to the production order requirement, the weighted values are set as Table3.

Table 2: worker information

Worker ID	Skill Number	Skill	Maximum Working Time	Salary
w_0	2	0,1	500	5600
w_1	2	0,3	500	4900
w_2	2	0,1	500	6200
w_3	2	0,4	500	4000
w_4	2	0,3	500	5100
w_5	3	0,1,3	500	6600
w_6	2	1,3	500	5800
w_7	2	0,3	500	4200
w_8	2	0,2	500	3500
w_9	2	0,4	500	6000

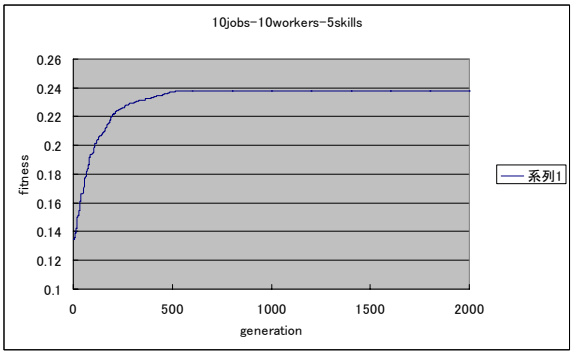
Table3: weighted value

v_{cost}	0.01
$v_{overwork}$	0.001
$v_{duration}$	10^{-8}



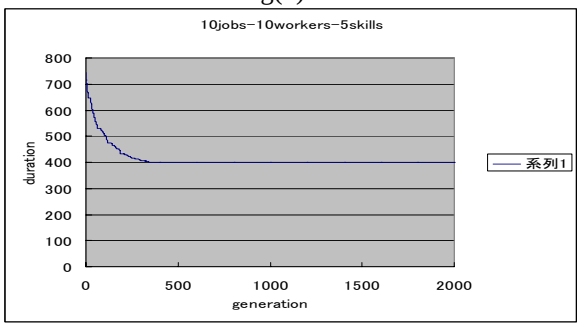
Fig(i): Job precedence graph

The test result is shown as Fig (j). The evolution of the solutions which assign 10 workers to 10 jobs is illustrated. The evolution process can reach its optimal level after around 500 generations. And the optimal fitness value is 0.2377.



Fig(j): genetic evolution of composite objective

Not only the composite fitness, but also the minimum value of duration , human cost and production overwork can be found by setting the weighted value. To find the minimum production duration, the human cost weighted value equals to 0, and the production overwork weighted value equals to 0. The result is shown as fig(k).



Fig(k): genetic evolution of duration as primary objective

The minimum duration is 400 hours, and the evolution process reached its minimum level after around 400 generations. The minimum human cost and the minimum production overwork also can be gotten with setting the appropriate weighted value.

V. CONCLUSION

In this paper, the worker allocation problem has been tackled with genetic algorithm. This problem is essential for the manual labor manufacture nowadays and automatically finding optimal solutions to help factory save cost and time. A production manager can study different scenarios with such an automatic tool in order to make appropriate decisions on the optimal production. In addition, according to this approach, he can adjust the weighted value to adapting production requirements. The worker allocation is combinatorial optimization problem and any exhaustive search can take too much time to get a solution. The contribution to the worker allocation is an automated tool based on the genetic algorithms that can be used to assign people to jobs in a near optimal way.

The result shows that the proposed model could find effective solutions with shortening the project period, reducing the project cost and controlling the production overwork.

In the future, new instances will be added to study the influence of the instance parameter .In addition; the algorithm will be applied to real world data in order to illustrate how to use this technique in a real production. Also extend the model to be used in the real manufacture factory.

REFERENCES

[1] H.Iima ,N.Sannomiya, "Proposition of Module Type Genetic Algorithm and Its Application to Modified Scheduling Problems with Worker Allocation" IEEEJapan ,Vol.122-C, pp.409-416,2002

[2] A.Osawa , K. Ida ,"Scheduling Problem with Worker Allocation using Genetic Algorithm" , Japan-Australia workshop on intelligent and evolutionary systems, pp.1-8 ,2005

[3] H. Iima ,N.Sannomiya, "Module Type Genetic Algorithm for Modified Scheduling Problems with Worker Allocation" Proceedings of the American Control Conference Arlington, VA, 2001

[4] C.Chang "Genetic Algorithms for Project Management" Annals of Software Engineering , Vol.11, pp 107-139,2001

[5] E.Alba , J. F. Chicano," software project management with GAs" Information Sciences, Vol.177, pp.2380-2401 , 2007

[6] J. Holland, "Adaptation in natural and artificial systems", University of Michigan Press, Ann Arbor, 1975.

[7] D. A. Coley, "An Introduction to Genetic Algorithms for Scientists and Engineers", New Jersey, 1999.

[8] C.K. Chang, H.Y Jiang, "Time-line based model for software project scheduling with genetic algorithms" information and Software Technology , Vol.50 pp1142–1154,2008