Modern Applied Science; Vol. 11, No. 9; 2017
ISSN 1913-1844 E-ISSN 1913-1852
Published by Canadian Center of Science and Education

An Optimization Approach to the Preventive Maintenance Planning Process

Manar Mohammed Altehmazi[1], Saad M. A. Suliman[2] & Yaser Alalawi[2]

[1] Bahrain Petroleum Company, Bahrain

[2] Department of Mechanical Engineering, University of Bahrain, Bahrain

Correspondence: Saad M. A. Suliman, Department of Mechanical Engineering, University of Bahrain, Isa Town, P. O. Box 32038, Bahrain. Tel: 973-1787-6628. E-mail: ssulieman@uob.edu.bh

Received: July 16, 2017 Accepted: July 26, 2017 Online Published: August 5, 2017

doi:10.5539/mas.v11n9p20 URL: https://doi.org/10.5539/mas.v11n9p20

Abstract

Creating a good preventive maintenance schedule is essential to perform an efficient shutdown. This paper is presenting a mathematical non-linear model that is formulated for the turnaround maintenance scheduling problem, and proposing an algorithmic optimization approach that combines the scheduling and workforce allocation in one phase. The strategy used here mainly aims to filter the uncompleted tasks from the tasks set and then to filter again from the resulted uncompleted tasks the ones which are satisfying the precedence constraint. If a task is not completed because of its preceding task, then it is put under hold until the precedence is finished. Once these two conditions are satisfied, the allocation of processors (workers in departments) starts considering the available ones. The algorithmic optimization approach is based on customized objective function and a number of constraints. It is coded in MATLAB format and solved using a modified genetic solver. It provides an optimized or pseudo-optimized schedule and workforce allocation plan, saves time and effort, and as a consequence it improves the efficiency and effectiveness of the maintenance system. The efficiency of the proposed algorithm in terms of computation time is affected mostly by the number of assigned tasks and the branching density of the dependent tasks.

Keywords: preventive maintenance, turnaround, shutdown, planning, optimization, scheduling, un-identical multiprocessors system, genetic algorithm

1. Introduction

The preventive maintenance in oil refineries is a complicated process where a huge amount of work has to be accomplished in limited time and with limited resources. Even though, the current practice of preventive maintenance process is perceived to be of high standard, there is no evidence that this process cannot be improved further or optimized. Consequently, this study is aiming to provide proof of the usefulness of continuous process improvement and optimization by studying the Turn Around Maintenance (TAM) projects implemented in an oil refinery in the middle east (designated by Company A) and focusing on the scheduling process.

Creating a good schedule with good workforce allocation might be quite hard, if it is decided to be optimized. In the case study, the scheduling process depends mainly on the experience of the planners. It is a lengthy process taking time extending from 2 weeks for a small project to around 3 months for a large project, and the final schedule is usually not optimized in terms of time and workforce utilization. There is an obvious need for a user-friendly tool that overcomes most of the current deficiencies encountered with the TAM project scheduling process including the workforce allocation.

The aim of this paper is to minimize the total execution duration of a Turn Around Maintenance (TAM) project by determining the minimum make span that can be achieved and maximizing the workforce utilization. The paper proposes an optimization algorithmic approach based on a non-linear mathematical model that combines the scheduling and workforce allocation in one phase.

The following sections provide a brief on literature review, the scheduling optimization model formulation, the algorithmic solution approach, sensitivity analysis, discussion and conclusion, and suggested future work.

2. Previous Studies Review

A number of researches were made in optimizing the scheduling process in the industrial field. Several of them were studying general mathematical optimization modeling approaches. Smeitink and Dekker (1994) discussed heuristic solution that minimizes the long term average cost. Cooper et al. (1998) presented a modified list scheduling algorithm with different alternative strategies and supported by experiment. Bijvank (2004) presented a mathematical model to schedule the shutdown activities in the most appropriate sequence in order to minimize the involved overall cost and time. A dynamic scheduling method was provided by Srinivasan and Anderson (2005) with an introduction of joiners and leavers in a uniprocessor system without causing delays or missed deadlines. Barroeta (2005) presented a model that minimizes the total cost per unit time during component renewal cycle. Lindquist (2005) focused on both practical and technical methods to quantitatively estimate the maintainable equipment condition utilizing the results from independent inspection techniques.

Rahman et al., (2007) provided a dynamic critical path method based on the workflow. A genetic programming approach was introduced in a multiprocessor scheduling system by Agarwal (2007). Panthi et al. (2008) presented a mathematical model to estimate preventive maintenance cost for long term contracts using reliability concepts, considering the life time before a failure can take place and warranty cost. Hilber (2008) introduced an optimization model that minimized the life cycle cost and optimized the maintenance performance through a multi-objective problem, and as a conclusion he provided a number of heuristic solutions through an optimization algorithm. Moghaddam (2010) presented a multi objectives optimization models that were developed based on a set of economical and engineering considerations.

On the other hand, there are a number of researches describing several techniques used to solve the resource allocation problem. In one of these researches an optimization model was introduced, it balances the workforce by considering the skill differences and work mobility (Karabak et al., 2011). Very few researches addressed the combined task scheduling and resource allocation problem using one tool or unified solution like Bianco et al. (1995) and Freeman (2003).

3. Scheduling Optimization Model

The TAM scheduling problem that will be considered within the context of this paper is to set a schedule for a given list of tasks in task set J = (j1, j2, ..., jn) with minimum make span, optimal workforce allocation, and minimum computation time. In order to develop a mathematical model for this TAM scheduling and workforce allocation problem a number of assumptions are required based on a TAM real case study. The assumptions are: the total number of workers is constant and the workers are available all the time; each processor is able to complete his task at the given time; transformation time from one task to another is negligible or equal to zero; all the tasks are predetermined and mandatory; the relaxation of any task is out of the scope. In addition, all the tasks are performed at least with the minimum required quality standard. Therefore, it is assumed that quality is fully applied. Moreover, tasks' durations are known and calculated prior to assigning workforce. However, the total duration of the TAM project is not known and needs to be determined. Finally, once a number of workers from certain processor (department) assigned to perform a task, this number shall not be modified or changed until this task is completed. The parameters and variables used in the model are as follows.

3.1 Input Parameters

J: Set of all tasks $\{j_i;\ i=1,2,\ldots\ldots\ldots\ldots,n\}$, where n represents the maximum number of tasks

A: Set of all available processors $\{a_i;\ i=1,2,\ldots\ldots\ldots\ldots,m\}$, where m represents the maximum number of processors

a^T: The total number of workers in processor $a \in A$

N: An indicator for the initial task precedence input

$N_j = 1$, if task $j \in J$ has a precedence task

$= 0$, if task $j \in J$ has no precedence task

B: Processor ability matrix (Performing ability)

$B_{j,a} = 1$, if processor $a \in A$ has the ability to perform task $j \in J$

$= 0$, if processor $a \in A$ does not have the ability to perform task $j \in J$

$D^T_{j,a}$: The total execution duration of task $j \in J$ before assigning workers from processors $a \in A$, assuming that the task requires only one processor and one worker from that processor.

$D^P_{j,a}$: The preparation duration of task $j \in J$ for execution, which is a fixed time not affected by the increase of assigned workers from processor a ϵ A

3.2 Decision Variables

$X^t_{j,a}$: Allocation of tasks to processors (Allocation ability)

$X^t_{j,a} = 1$, if processor a ϵ A is assigned to complete task j ϵ J at time t

= 0 , if processor a ϵ A is not assigned to complete task j ϵ J at time t

$y^t_{j,a}$ is the number of workers from processor a ϵ A assigned to perform task j ϵ J at time t

$D^{af}_{j,a}$: The duration of the task j ϵ J after assigning workers from processor a ϵ A, $D^{af}_{j,a} \in \mathbb{R}$, where $\mathbb{R}$ is the group of real numbers

3.3 Supporting variables

P: The completion of precedence task. It is a dynamic variable changes with time, only active if Nj = 1.

$P^t_j = 1$, if task j ϵ J has an uncompleted precedence task at time t

= 0, if task j ϵ J has completed precedence task at time t

Q: The task completion. It is a dynamic variable changes with time.

$Q_j = 1$, if task j ϵ J is completed

= 0, if task j ϵ J is not completed

a^{av}_t is the available number of workers in processor a ϵ A at time t

t: Vector start time which increments with time, t ϵ $\mathbb{R}$

$Rt_{j,a}$: The release time for workers assigned from processor a ϵ A from task j ϵ J.

3.4 Objective Function

The objective is to minimize the make span (C) of a certain processor a ϵ A, which is repeated for all processors at all times t resulting into a dynamic optimization model. The number of workers assigned ($y^t_{j,a}$) to the available tasks j at time t, are set to minimize C:

$$\text{Minimize } C = \sum_{j=1}^{Jt} 1 * D^{af}_{j,a} = \sum_{j=1}^{Jt} 1 * (D^C_{j,a} / y^t_{j,a}) \ \forall\ a \in A,\ t \in T \tag{1}$$

The duration of the task after assigning workers from processor is $D^{af}_{j,a} = (D^T_{j,a} - D^p_{j,a}) / y^t_{j,a}$, where $D^{af}_{j,a}$ represents the duration of the task j after assigning workers from processor a, $D^T_{j,a}$ is the total duration of task j before assigning workers from processors a, $D^p_{j,a}$ represents the preparation duration of task j which is a fixed time not affected by the increase of assigned workers from processor, $y^t_{j,a}$ is the number of workers from processor a assigned to perform task j. Since $D^T_{j,a}$ and $D^p_{j,a}$ are given values, therefore for simplicity the value is calculated and given as $D^C_{j,a} = D^T_{j,a} - D^P_{j,a}$, therefore $D^{af}_{j,a} = D^T_{j,a} / y^t_{j,a}$. Equation 1 represents a customized objective function of dynamic nature.

In the final step to calculate the total execution duration of the TAM project the following expression, which represents the maximum finish time of all the tasks in set J, is used:

$$D_{TAM} = \underset{\forall j}{\text{Max}} (S_j + D^{af}_{j,a}) = \underset{\forall j}{\text{Max}} (S_j + (D^C_{j,a} / y^t_{j,a})) \tag{2}$$

3.5 Problem Constraints

The following constraints need to be observed:

$$X^t_{j,a} = 0 \ \forall\ j \in J,\ a \in A \ \big|\ B_{j,a} = 0 \tag{3}$$

The above equation shows that processor a cannot be assigned to a task unless it has the right matching skill (if $B_{j,a} = 0$, then $X^t_{j,a} = 0$). This constraint as explained above narrows the total binary decision variables needed. So it decreases the needed time for the solution (Freeman, 2003).

$$P^t_j \geq 0 \ \forall\ j \in J \ \big|\ N_j = 1 \tag{4}$$

This expression is to reduce the number of supporting binary decision variables; the decision of having a completed or non-completed precedence task k is only active if task j has a precedence task k.

$$P^t_j = 0 \ \forall\ j \in J \ \big|\ N_j = 1 \text{ and } Q_k = 1 \tag{5}$$

The above equation shows when task j have a completed precedence k. The equation reduces the number of supporting binary decision variables.

$$Q_j = 1 \quad \forall\ j \in J \ \middle|\ D^{af}_{j,a} > 0 \text{ and } y^t_{j,a} > 0 \tag{6}$$

This equation shows when task j is considered completed. It reduces the number of binary decision variables.

$$\Sigma_{a \in A} X^t_{j,a} = 1 \ \forall\ j \in J \tag{7}$$

The above equation specifies that each task j must be assigned once to one processor, i.e. a task cannot be shared by two or more processors (departments) (Freeman, 2003).

$$\mathrm{Min}_j \le y^t_{j,a} \le \mathrm{Max}_j$$
$$\forall\ j \in J,\ \forall\ t \in \mathbb{R} \ \middle|\ P^t_j = 0,\ Q_j = 0,\ B_{j,a} = 1 \tag{8}$$

The ytj,a must not exceed the acceptable upper limit of task's workers capacity as explained earlier, limited by working area size or available machines/tools, and must not be less than the acceptable lower limit of task's workers capacity able to perform task j.

$$y^t_{j,a} \ge 0$$
$$\forall\ j \in J,\ \forall\ t \in \mathbb{R},\ y^t_{j,a} \text{ are integers } \ \middle|\ P^t_j = 0,\ Q_j = 0,\ B_{j,a} = 1 \tag{9}$$

This expression is to make sure that the number of workers from each processor shall be integer and cannot accept negative numbers.

$$a^{av}_t = a^T - \sum_{j=1}^{Jt} 1 * y^t_{j,a} * X^t_{j,a}$$
$$\forall\ j \in J,\ a \in A,\ \forall\ t \in \{t : Rt_{j,a}\} \ \middle|\ P^t_j = 0,\ Q_j = 0,\ B_{j,a} = 1,$$
$$\text{or } \middle|\ N_j = 0,\ Q_j = 0,\ B_{j,a} = 1 \tag{10}$$

This equation is to calculate the available number of workers in each processor after each increment of time, subtracting the utilized workers in all task j from the total number of workers given at the beginning of the process for each processor. But this equation must satisfy the condition of the task being not completed $Q_j = 0$, has no uncompleted precedence $P^t_j = 0$, and the processor a has the ability to perform the task j $B_{j,a} = 1$, or satisfying the condition of the task being not completed $Q_j = 0$, has no precedence task $N_j = 0$, and the processor a has the ability to perform the task j $B_{j,a} = 1$.

If $y^t_{j,a}$ is determined for $D^{af}_{j,a}$, then the number of workers from processor a which assigned to task j ($y^t_{j,a}$) is not available during all the duration of $D^{af}_{j,a}$, so if this was assigned at time t, then the workers are not available until the end of duration $t + D^{af}_{j,a}$. Therefore, the release time for these workers is $Rt_{j,a} = t + D^{af}_{j,a}$.

4. Algorithmic Solution Approach

The proposed model (expressions 1-10) is a nonlinear integer programming formulation for the TAM scheduling problem, which is considered as a large sized problem. Exact solution of such problem is expensive in terms of computation time. To overcome this difficulty an algorithmic solution approach is developed. The developed algorithm is solved by a MATLAB genetic solver that can solve constrained and non-constrained optimization problems based on natural selection. The method repeatedly modifies a population of individual solutions to get the optimal solution.

The developed model solution approach as indicated in Figures 1 and 2 starts by identifying the available tasks that can be scheduled (from the task set) for a specific processor. The available tasks include any task with no preceding task or with finished preceding task. These available tasks are then scheduled for that specific processor and workers are allocated by solving the model of equations 1 to 10 by the genetic solver.

Once the available tasks are scheduled and their starting and finishing times are set, the algorithm identifies the task with the earliest finishing time. The time indicator is incremented to that time and the available tasks list is updated, then the available tasks are scheduled and workers allocated to the specific processor. This process is repeated until no further tasks become available for that processor. The same procedure is repeated for the other processors and updated until the entire task set is scheduled and required workforce is allocated. Some processors might be visited more than once based on when the task will be available.

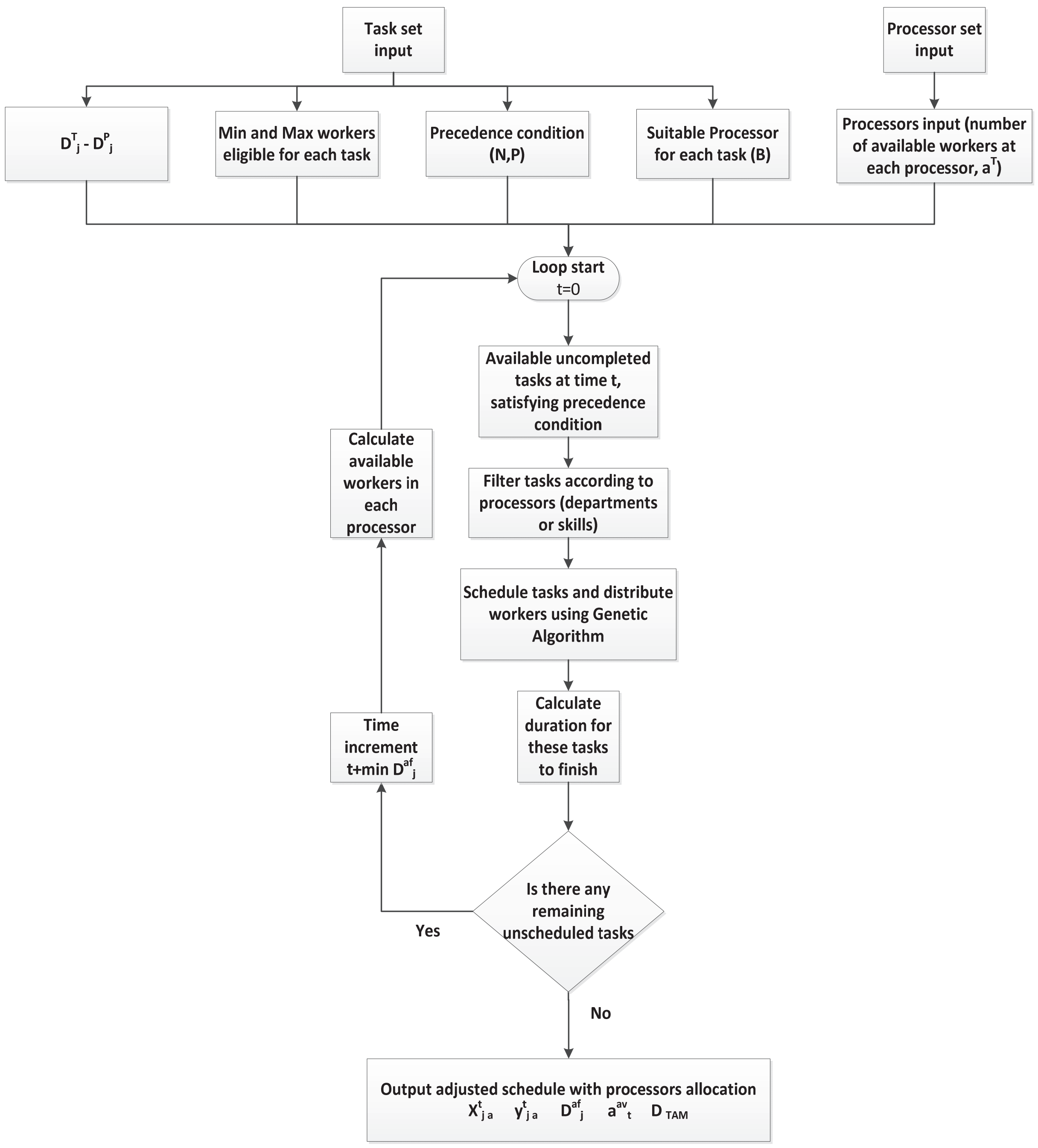
Task set input
Processor set input
$D^T_j - D^P_j$
Min and Max workers eligible for each task
Precedence condition (N,P)
Suitable Processor for each task (B)
Processors input (number of available workers at each processor, a^T)
Loop start t=0
Available uncompleted tasks at time t, satisfying precedence condition
Filter tasks according to processors (departments or skills)
Schedule tasks and distribute workers using Genetic Algorithm
Calculate duration for these tasks to finish
Is there any remaining unscheduled tasks
Yes
No
Time increment t+min D^{af}_j
Calculate available workers in each processor
Output adjusted schedule with processors allocation X^t_{ja} y^t_{ja} D^{af}_j a^{av}_t D_{TAM}

Figure 1. The general systematic process flow chart of the developed algorithm

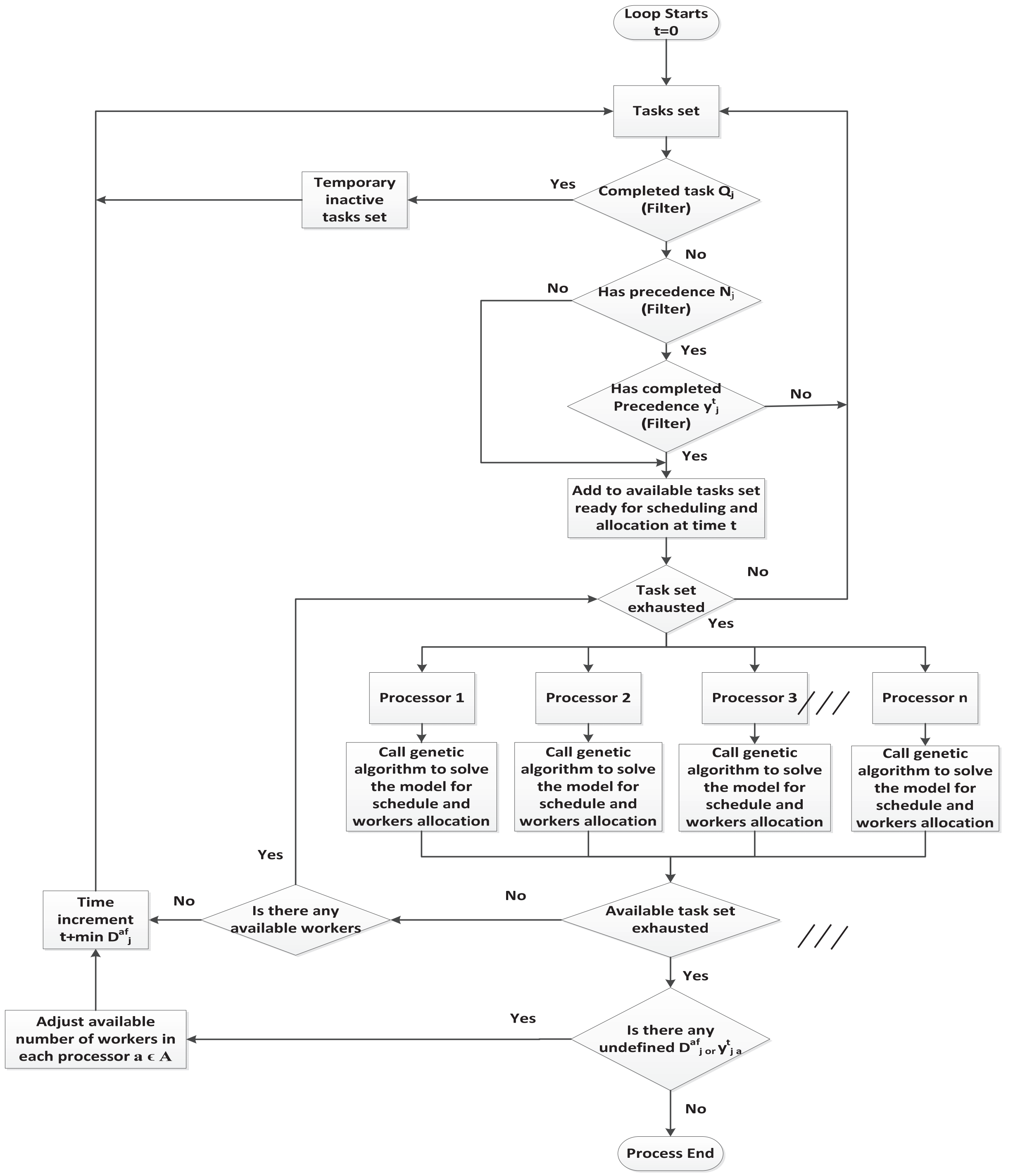
Loop Starts t=0
Tasks set
Completed task Q_j (Filter)
Yes
Temporary inactive tasks set
No
Has precedence N_j (Filter)
No
Yes
Has completed Precedence y^t_j (Filter)
No
Yes
Add to available tasks set ready for scheduling and allocation at time t
Task set exhausted
No
Yes
Processor 1
Processor 2
Processor 3
Processor n
Call genetic algorithm to solve the model for schedule and workers allocation
Available task set exhausted
No
Is there any available workers
Yes
No
Time increment t+min D^{af}_j
Yes
Is there any undefined D^{af}_j or Y^t_{ja}
Yes
Adjust available number of workers in each processor a ϵ A
No
Process End

Figure 2. The detailed decision making process flow chart of the developed algorithm

5. Results

The established algorithm solving program provides the final schedule that specifies for each task the number of assigned workers, duration, start time, and finish time, In addition, it provides network diagram and Gantt chart illustrating the tasks durations, and schedule. The chart can be used for monitoring and follow up the TAM project execution.

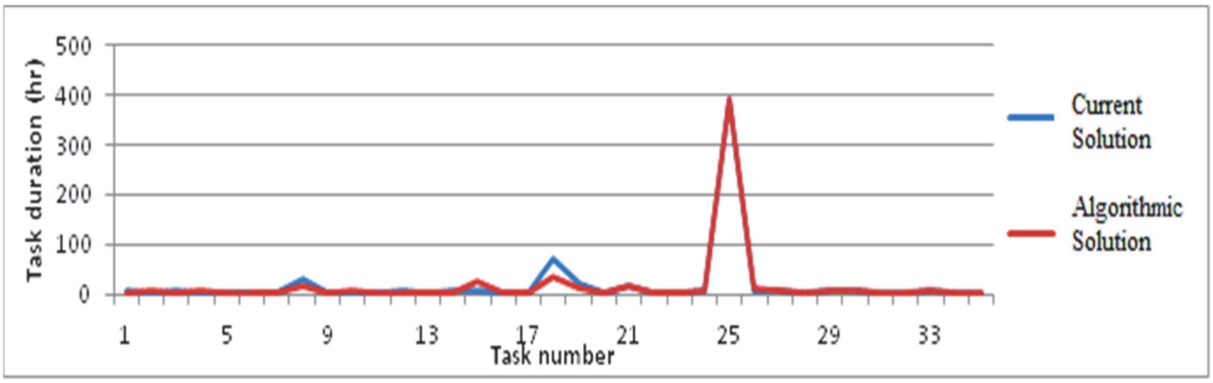

Figure 3. The task duration of the current solution vs. the algorithm solution

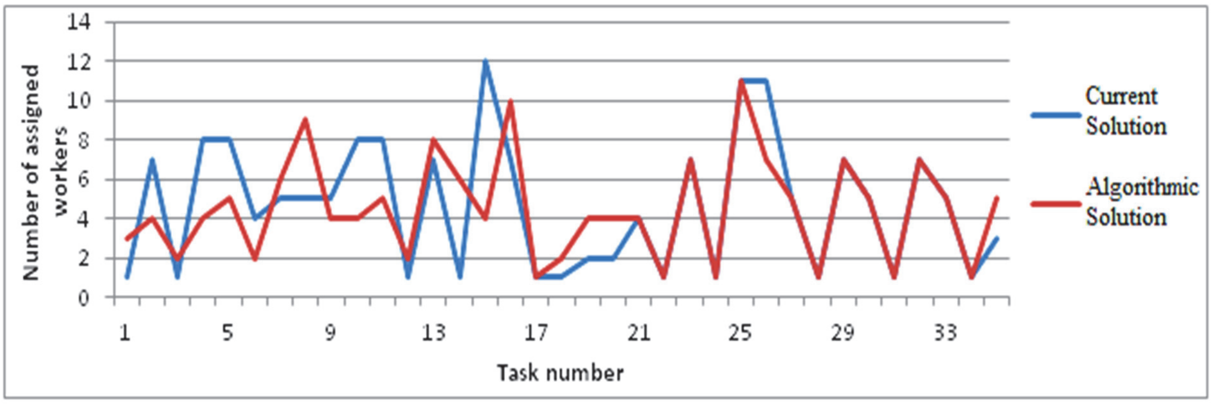

Figure 4. The number of assigned workers of the current solution vs. the algorithm solution

The established model and its coded algorithm are used to solve a small TAM scheduling project provided by company A (real data). The results provide almost similar results of the current available solution regarding the number of assigned workers and duration, as indicated in Figures 3 and 4. However, an improvement in the total TAM execution duration is noticed. The proposed algorithm shows a reduction of 4.8% in the project total execution duration, which is achieved by a better utilization of the workforce resources. The most important improvement is the reduction in the time required for solving the TAM scheduling problem (elapsed time). The solution of the TAM project by the developed algorithm is achieved in 51.34 seconds only (elapsed time) whereas it used to be achieved in around 40 hrs by the currently used solution approach. After all, the algorithm provides a better schedule, than the currently available one, based on the given data, and reduces the solution time.

The developed algorithm is also used in solving a real application TAM project given by the same Company A, which proves the workability of the algorithm for large scale TAM projects.

5.1 Sensitivity Analysis

The algorithm is assessed for its performance in terms of elapsed time. The assessment is done by conducting one-way ANOVA sensitivity analysis for the algorithm decision parameters including, branching density (precedence relationship), number of processors, and number of tasks. The results of Tables 1-3 show that:

- changing the density of branching affects the algorithm performance in terms of computation elapsed time. Lesser branching projects has shorter elapsed time, while higher branching density and more complex precedence relationships of projects' tasks have longer elapsed time.
- changing the number of assigned processors (departments) is not affecting the algorithm performance in terms of elapsed time, increasing or decreasing the number of assigned departments or skills have no effect on the elapsed time.
- changing the number of assigned tasks affects the algorithm performance in terms of elapsed time. Lesser number of tasks has shorter elapsed time, while greater number of tasks has longer elapsed time.

The assessment is supported by Two-way ANOVA analysis to test the interaction of the parameters on the algorithm performance. The results show that all the parameters affect the algorithm performance but to different extents. The most significant effect is referred to density of branching and number of tasks, and their interaction (Table 4).

Table 1. The computed elapse time and elapsed time percentage error* of different branching density levels

Branching	Elapsed time**		Elapsed time percentage error***	
density	Range	Average	Range	Average
2	26.47 – 28.84	27.897	-3.38 – 5.11	0
5	33.13 – 33.55	33.347	-0.60 – 0.65	0.004
7	49.34 – 50.08	49.652	-0.86 – 0.62	0.002
10	65.18 – 68.85	65.916	-4.45 – 1.11	0.005
12	81.73 – 83.71	82.35	-1.65 – 0.75	0.002

Table 2. The computed elapse time and elapsed time percentage error* of different number of processors

Branching	Elapsed time**		Elapsed time percentage error***	
density	Range	Average	Range	Average
4	24.25 – 25.22	24.537	-2.78 – 1.16	0.005
6	24.15 – 24.63	24.262	-1.51 – 0.46	0.002
8	24.02 – 24.98	24.392	-2.41 – 1.52	0.002

Table 3. The computed elapse time and elapsed time percentage error* of different number of tasks

Branching	Elapsed time**		Elapsed time percentage error***	
density	Range	Average	Range	Average
15	11.8 – 12.61	12.225	-3.14 – 3.47	0
30	32.05 – 32.57	32.242	-1.01 – 0.59	0.001
60	100.57 – 101.99	101.371	-0.61 – 0.61	0
90	215.5 – 222.96	219.071	-1.77 – 1.63	0

* Percentage error = (Average error – error / Average error) *100

** The values are for 10 trials

*** Elapsed time percentage error is relative of the 10 trials

Table 4. The results of two-way ANOVA analysis

Source	$F_\circ$
A (Tasks)	116810
B (Dept.)	98.41822
C (Branching)	58215.24
AB	70.01156
AC	4734.555
BC	103.4087
ABC	65.91624
$F_{critical}$ (F_α) = 4.2597 for $\alpha = 0.05$	

6. Conclusion

This study is an extension of the scheduling optimization studies of the non-identical multiprocessor and dependent tasks. It provides a tool that combines the scheduling and workforce allocation in one phase. The basic idea of the adopted approach is to optimize the scheduling process of TAM project by determining the minimum total duration (make span), which can be achieved by maximizing the workforce utilization. The study addresses the TAM scheduling process for a number of tasks, deciding the suitable order of these tasks, and allocating the right workforce size for each task which provides the project minimum make span.

The results show that the proposed algorithmic model provides good workforce deployment and produces a detailed schedule specifying when each task starts and when it is expected to end, and how many workers are assigned for each task. The model takes in consideration the un-identical multiprocessors system (different departments with different skills). The major conclusion is that using the proposed model saves time and effort, provides an optimized schedule and workforce allocation plan, and as a consequence it improves the efficiency and effectiveness of maintenance systems. The efficiency of the proposed algorithm in terms of computation time is affected mostly by the number of assigned tasks and tasks' branching density of the dependent tasks.

The proposed algorithm performs well and gives a feasible solution which is highly dependent on the problem objective function and constraints. It provides better time deployment, reduces the total TAM execution duration, reduces solving time, and provides a semi-optimal if not optimal schedule and workforce allocation, without compromising the quality of the work.

For Future work, it is recommended to investigate task durations of probabilistic nature rather than deterministic.

References

Agarwal, A. (2007). A neurogenetic approach for multiprocessor scheduling. *Eugene Levner International Journal, 2*, 121-136. Retrieved from http://anuragagarwal.com/ResearchPapers/2006PMPSMultiprocessorSchedulingNGApproachSchedulingAgarwal.pdf

Barroeta, C. (2005). *Risk and economic estimation of inspection policy for periodically tested repairable components.* Master thesis, University of Maryland, Maryland, USA. Retrieved from http://drum.lib.umd.edu/bitstream/handle/1903/2921/umi-umd-2712.pdf;jsessionid=18605479E35EA3C43ECC4F440CA78CCD?sequence=1

Bianco, L., Blazewicz, J., Olmo, P. D., & Drozdowski M. (1995). Scheduling multiprocessor tasks on a dynamic configuration of dedicated processors. Master thesis, Instytut Informatyki Politechniki Poznanskiej, Poland. Retrieved from http://www.cs.put.poznan.pl/mdrozdowski/txt/SMTDCDP-AOR.pdf

Bijvank, M. (2004). Shutdown scheduling a practical approach to handle shutdowns at refineries. Master thesis, Vrije University, Amsterdam, Netherlands.

Cooper, K., Schielke, P., & Subramanian, D. (1998). An experimental evaluation of list scheduling. Master thesis, Rice university, Texas, USA. Retrieved from https://pdfs.semanticscholar.org/592c/49d6f72eba46a27629931c37d93d2d2b380e.pdf

Hilber, P. (2008). Maintenance optimization for power distribution systems. Doctoral thesis, Royal Institute of Technology, Stockholm, Sweden. Retrieved from https://pdfs.semanticscholar.org/231f/96ca971f856ec78a23dcafb351ac308cb06f.pdf

Karabak, F., Güner, N., Satır, B., Kandiller, L., & Gürsoy, İ. (2011). An optimization model for worker assignment of a mixed model vehicle production assembly line under worker mobility. Proceedings 41st International Conference on Computers and Industrial Engineering, Los Angeles, USA. Retrieved from https://pdfs.semanticscholar.org/d1c9/1a30e568a646ead914b67db0cd80ef822297.pdf

Lindquist, T. (2005). Reliability and maintenance modeling of aging equipment in electric power systems. *KTH Journal, 6*, 42.

Moghaddam, K. (2010). Preventive maintenance and replacement scheduling models and algorithms. Doctoral thesis, University of Louisville, Kentucky, USA.

Panthi, K., Farooqui, R., & Ahmed, S. (2008). Reliability based model for estimating long term pavement maintenance contracts under performance specifications. Proceeding of the Proceedings of the First International Conference on Construction in Developing Countries(ICCIDC-I), Karachi, Pakistan (August 4-5, 2008).

Rahman, M., Venugopal, S., & Buyya, R. (2007). A dynamic critical path algorithm for scheduling scientific workflow application on global grids. Proceedings of the Third IEEE International Conference on e-Science and Grid Computing, USA.

Semitink, E., & Dekker, R. (1994). Preventive maintenance at opportunities of restricted duration. *Naval Research Logistics journal, 4*, 335-353.

Srinivasan, A., & Anderson, J. H. (2005). Fair scheduling of dynamic task system on multiprocessors. *Systems and Software Journal, 77*, 67-80.

Tompkins, M. F. (2003). *Optimization techniques for task allocation and scheduling in distributed multi-agent operation. Master thesis, Massachusetts institute technology, Massachusetts, USA.* Retrieved from https://www.researchgate.net/file.PostFileLoader.html?id=587f67763d7f4b421712fc45&assetKey=AS%3A451865612689408%401484744565718