Machine Learning Priority Rule (MLPR) For Solving Resource-Constrained Project Scheduling Problems

Patience Imoh Adamu and Olufemi T. Aromolaran

***Abstract*— This paper introduces a machine learning priority rule for solving non-preemptive resource-constrained project scheduling problems (RCPSP). The objective is to find a schedule of the project's tasks that minimizes the total completion time of the project satisfying the precedence and resource constraints.**

Priority rule based scheduling technique is a scheduling method for constructing feasible schedules of the jobs of projects. This approach is made up of two parts: a priority rule to determine the activity list and a schedule generation scheme which constructs the feasible schedule of the constructed activity list. Different scheduling methods use one of these schemes to construct schedules to obtain the overall project completion time. Quite a number of priority rules are available; selecting the best one for a particular input problem is extremely difficult. We present a machine learning priority rule which assembles a set of priority rules, and uses machine learning strategies to choose the one with the best performance at every point in time to construct an activity list of a project. The one with better performance is used most frequently. This removes the problem of manually searching for the best priority rule amongst the dozens of rules that are available.

We used our approach to solve a fictitious project with 11 activities from Pm Knowledge Center. Four priority rules were combined. We used serial schedule generation scheme to generate our schedules. Our result showed that the total completion time of the project obtained with our approach competes favorably with the completion times gotten with the component priority rules. We then went further and compared our algorithm with 9 other available priority rules. Our results showed that the completion time got using our algorithm compete favorably with the total 13 priority rules available in the literature.

Index Terms— **machine learning, motion planning, network analysis, resource constraints, probabilistic roadmap planners.**

I. Introduction

PROJECT scheduling involves the scheduling of the jobs of a project to obtain a feasible schedule which optimizes desired performance criteria. When the resources required to execute the jobs are constrained, we have the resource-constrained project scheduling problem (RCPSP) and either single-mode RCPSP (Talbot and Patterson [9]) or multi-mode RCPSP (Talbot [8], Hartmann [3]). The Single-Mode RCPSP involves scheduling the jobs of a project and putting into consideration the precedence and the renewable resource availability constraints. When a job is started it is either preemptive or non-preemptive. The typical objective which is what we are considering in this paper is usually to minimize the total completion time of a project.

The multi-mode RCPSP is a generalization of the single-mode RCPSP. Within the multi-mode RCPSP, a job can be performed in one out of a set of execution modes with a specific duration and resource requirements. Each mode represents another way of mixing different levels of resource requirements with a related duration (for example, 2 electricians need 5 days to repair an electrical fault (mode 1), while 3 electricians and 2 unskilled laborers may need 2 days to repair the same electrical fault (mode 2)).

The RCPSP has been shown to be an NP-hard optimization problem (Blazewicz [1]). Exact methods become intractable therefore as the number of activities and the number of modes for each activity increases (Sprecher and Drexl [7]).

Hence, in practice heuristic algorithms to generate near-optimal schedules for larger projects are of special interest.

This problem has found application in many real life applications and industries, such as project management and crew scheduling, construction engineering, production planning and scheduling, fleet management, machine assignment, automobile industry and software development.

Priority rule based scheduling technique is a scheduling method for constructing feasible schedules of the jobs of projects. This approach is made up of two parts: a priority rule to determine the activity list and a schedule generation scheme which constructs the feasible schedule of the constructed activity list. Many priority rules exist (Vanhoucke [10]) for various tasks while basically there are two well-known generation schemes available: the serial schedule generation scheme (SSGS) and the parallel schedule generation scheme (PSGS) [6]. Different scheduling methods use one of these schemes to construct schedules to obtain the overall project makespan (project completion time). This can be seen in [3] who introduced a genetic algorithm approach that used latest finish time priority rule to obtain the activity list for an individual and then used the parallel generation scheme to get the schedule

Manuscript received July 10, 2017; revised July 20, 2017.

P. I. Adamu is with the Mathematics Department, Covenant University, Ota, Nigeria.
patience.adamu@covenantuniversity.edu.ng
O. T. Aromolaran is with the Department of Computer and Information Science, Covenant University, Ota,, Nigeria
phemmysmart@gmail.com

for that individual. Also, Kadem and Mane [5] presented a genetic-local search algorithm for RCPSP. This algorithm uses priority base crossover, neighborhood mutation operator and neighborhood search procedure to combine elements from evolutionary and local search procedures. The solution is an activity list which uses SSGS for generation of results.
In the literature, because quite a number of priority rules are available, selecting the best one for a particular input problem is extremely difficult. This is the motivation of this study.

In this paper, we present a machine learning priority rule (MLPR) approach which uses machine learning to dynamically decide which priority rule to use. It assembles a set of priority rules and uses machine learning to identify the best rule automatically when constructing an activity list of a project. This approach is inspired by Hybrid PRM (Hsu et al.[4]), and adaptive neighbor connections (ANC) for PRMS [2].
Hybrid PRM (Hsu et al. [4]) assembles a number of component samplers and uses machine learning approach to identify the best component samplers to use at every stage while ANC for PRMS (Ekenna [2]) assembled a list of neighbor finders and uses the same machine learning approach to identify the best component to use for generating PRM roadmaps in a heterogonous environment.
What MLPR approach essentially does is to find a schedule of a project that minimizes its completion time even though available resources are constrained. It does this by assembling a set of priority rules. Observe their success rates and costs and then choose the one with the best performance to construct an activity list. Having obtained the activity list, a SSGS is then employed to construct the schedule which gives the completion time of the project.
We used an example project network with 11 jobs from Pm Knowledge Center for our experiment. We compared MLPR with 13 priority rules available and it competed favorably with them in finding the completion time of the example project. WE first of all used four of the rules as component rules for MLPR and compared their completion times. MLPR was found to minimize the completion time of the project better than 3 of them and in the same level with the fourth one. When we compared with the completion times using 9 more priority rules, the finished time of all the 13 rules, ranges from 24 – 29 weeks while that of MLPR is 24, the minimum value in the range. Hence instead of looking for the best priority rule to use, MLPR may be used. This implies that it eases the burden of trying to find the best priority rule amongst rules.
The main contribution of this work is to propose a new machine learning priority rule for finding the minimum schedule that minimizes the completion time of a resource-constrained project scheduling problem.

II. Preliminaries Related Works

A. Hybrid PRM

Probabilistic Roadmap Methods (PRMs) are sampling based motion planning algorithms. PRM divides planning into two phases: the learning phase, during which the roadmap is built; and the query phase, during which the user-defined query configurations are connected with the recomputed roadmap. The nodes of the roadmap are configurations and the edges of the roadmap correspond to free paths computed by a local planner. Hybrid PRM brings together a number of sampling strategies and then uses machine learning approach to know the best strategy to use at every point in time in sampling the nodes to generate the roadmap in the learning phase. We use the same approach, but applying it to priority rules for the construction of activity list.

B. Adaptive Neighbor Connections for PRMS

Like the Hybrid PRM, the authors of ANC introduced a strategy that adaptively combines multiple neighbor finding strategies for generating PRM roadmaps. It is mainly for heterogeneous environments which facilitates parallelism. This framework learns which strategy to use by examining their success rates and costs.

C. Priority Rules

A Priority rule approach is a method for constructing activity lists of the activities of a project.
The calculations of priority rules are often based on the four major information of the project and their network. They are as follows:
Activity Information, e.g. duration of the activities: An example of priority rules that use this information is shortest processing time (SPT). It arranges the activities in such a way that the one with minimum duration comes first in the list.
Network Structure Information: Examples are most immediate successors (MIS) and least non-related jobs (LNRJ). MIS puts the activities which has more direct successors first in the activity list while LNRJ puts activities with least number of non-related activities first in the list.
Scheduling information: An example is earliest finish time (EFT). It lists the activities by putting first the ones with minimum earliest finish time in the list.
Resource information: An example is greatest resource work content (GRWC). It puts the activities in a decreasing order of their work content in the list. Work content is duration multiplied by its renewable resource demand (duration x resource demand) of a job.

D. Schedule Generation Scheme (SGS)

Quite a number of heuristic algorithms for resource-constrained project scheduling problems (RCPSP) use SGS. It constructs feasible schedule from the priority list of the project activities. This, it does by removing activities from the priority list one at a time and assigning a starting time to each of them (i.e. partial schedule). This removal continues until starting time is assigned to the entire project activities (this is called a schedule). Basically there are two different types of SGS available: The serial schedule generation scheme (SSGS) and the parallel schedule generation scheme (PSGS). The SSGS uses activity-incrementation principle while the PSGS uses time- incrementation principle.

E. Serial Schedule Generation Scheme (SGS)

This scheme uses the activity-incrementation principle which schedules activities of a project one at a time and in

ISBN: 978-988-14048-8-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

its earliest precedence and resource feasible completion time. There are N stages in this scheme if there are N activities. In each stage the scheme scans the priority list, selects the next activity and schedules it at its earliest starting time making sure that the precedence and resource constraints are not violated. There are three sets of activities associated with each stage: the scheduled set, which consist of the already scheduled activities; the eligible set, which consist of all the activities that are eligible for scheduling and the ineligible set, which consist of all the activities not yet scheduled and cannot be scheduled in that stage.

III. MACHINE LEARNING PRIORITY RULE (MLPR)

The proposed priority rule (MLPR) uses a set of priority rules $pr_1, pr_2, pr_3, \cdots, pr_n$ to construct an activity list of the activities of a project. This is done in stages. Each priority rule maintains a probability $p_1, p_2, p_3, \cdots, p_n$. In every stage the MLPR observes the success rates and costs of all the priority rules and chooses the one with better performance to input an activity into the activity list.
How to measure the performance of the priority rules and update the probabilities are for the discussion of the next subsections?

(A) Performance Measures:

The performance measure of the priority rules is the measure of their success rates. Success rates of priority rules are measured by the total rewards received. If the chosen priority rule is able to input a job into the activity list that is being constructed successfully, it receives a reward which is the duration of the scheduled job. Successful in the sense that the job put into the activity list, does not violate any precedence and resource constraints. This makes the probability to increase in the next stage. Otherwise, the priority rule is punished and its probability decreases in the next stage. The cost of a priority rule is the cost of the scheduled job. That is the resource demand of that job. A high cost reduces the probability of a priority rule to be chosen in the next pick.

Algorithm 1: Machine Learning Priority Rule (MLPR)

Input: a) A project with J number of jobs $(j = \{1, 2, \cdots, J\})$.
b) A set of n priority rules, $pr_1, pr_2, pr_3, \cdots, pr_n$ $(i = 1, 2, \cdots, n)$.
Output: A feasible schedule of the Project's makespan (completion time).
Require:
Let $P = \{p_1, p_2, p_3, \cdots, p_n\}$ be a set of probabilities where p_i is the probability of choosing a priority rule.
Initialize $p_i(t = 1) = \frac{1}{n}$ $\forall$ *the priority rules.*
i) Initial weight of each priority rule is one $(w_i(1) = 1)$.
ii) Initial cost of each priority rule is assumed to be one $(c_i(1) = 1)$..
iii) Let AL = $\{j_1, j_2, \cdots, j_r\}; r < J$ be the activity list under construction.
for Step t = 1, 2, … ***do***
1) Write out the set of Eligible activities.
2) Choose a priority rule $pr_i(t)$ using P
3) Run the chosen rule $pr_i(t)$ to get the appropriate activity from the set of eligible activities.
4) Put the activity j_r into the activity list AL that is being constructed.
5) Get the reward of the chosen priority rule.
6) Update success rates
7) Update costs
8) Update Probabilities of all the priority rules.
9) If the activity list is finally got, use SSGS to construct feasible schedule of the activities which gives the completion time of the project.

B Probability update:

In the initial stage of MLPR, all the assembled priority rules are given the same probability of been chosen (i. e. for n number of priority rules, $p_i = \frac{1}{n}$ $\forall$ *the rules*) so that anyone can be randomly chosen for a start. Therefore, in subsequent steps the probabilities have to be updated for subsequent picks. We use the same probability update similar to Hybrid PRM.
To be able to record the past performance of component rules, weights for each of the rules are introduced. Initially, weight for each rule is set equal to one.
A weighted probability p_i^w based on the weights is computed for each of the rules in every step t:

$$p_i^w = (1-\gamma)\frac{w_i(t)}{\sum_{j=1}^{n} w_j(t)} + \gamma\frac{1}{n}, \qquad i = 1, 2, \cdots, n$$

where p_i^w is a convex combination of the *exploitation* and *exploration* probabilities of the priority rules. The first component keeps track of the probability of exploitation of a rule and the second one records the probability of exploration of the rule. The probability of exploitation of a priority rule varies directly with the rate at which that rule is been chosen while the probability of exploration of a rule is the same for all priority rules, giving each, the same chance of being chosen. The weighted probability balances the *exploration* and *exploitation* probabilities in each iteration that is γ *is* 0.5.

Putting the cost of each priority rule into consideration in each step t, the costed probability p_i^c is calculated as follows:

$$p_i^c = \frac{p_i^w / c_i}{\sum_{j=1}^{n} p_j^w / c_j}, \quad i = 1, 2, \cdots, n$$

where c_i is the average of the costs each priority rule incured over time? This is the probability of choosing a priority rule in step t.

C Weight Update:

In step t, the chosen priority rule p_{ri} has a reward $= r_i(t)$, and all the other priority rules p_{rj}, $j \neq i$ have zero rewards
The weighted reward is defined as:
$r_i^w(t) = r_i(t)/p_i^w(t), i = 1, 2, \cdots, n$
Now the weight update for all the rules in the next step is defined as follows:

ISBN: 978-988-14048-8-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

$$w_i(t+1) = w_i(t)\exp\left(\gamma\frac{r_i^W(t)}{n}\right);\ i = 1,2,\cdots,n$$

This implies that;

$$\frac{w_i(t+1)}{w_i(t)} = \exp\left(\gamma\frac{r_i^W(t)}{n}\right);\ i = 1,2,\cdots,n$$

This shows that, the weight of each rule depends on the reward received. The exponential factor of the reward exposes how quickly the weight changes as the reward changes.

D *Reward and Cost update*:

The reward $r_i(t)$ of each priority rule is the duration of the job that has been successfully put in the activity list while the cost $c_i(t)$ of each priority rule is the resource demand of the job. The duration of the scheduled job is chosen as the reward for that priority rule used because the time of completion of the project is directly proportional to the duration of each job. When a job is scheduled, what it gives to the system is its duration. And the cost of the of the priority rule naturally follows as the resource demand of the job.

The new reward (cost) is the average of rewards (costs) as t increases.

$$r_i(t) = \frac{1}{m}\sum_{i=1}^{m} r_i(t);\ t = 1,2,3\cdots$$

$$c_i(t) = \frac{1}{m}\sum_{i=1}^{m} c_i(t);\ t = 1,2,3\cdots$$

where m is the number of times a priority rule is chosen as t increases.

IV. EXPERIMENTS

We used the fictitious project network from PM Knowledge Center (Vanhoucke [10]) as an example to find the completion time of the project and compare our algorithm with 13 other priority rules. It has eleven activities and finish-start precedence relations. We assume each time period to be one week.

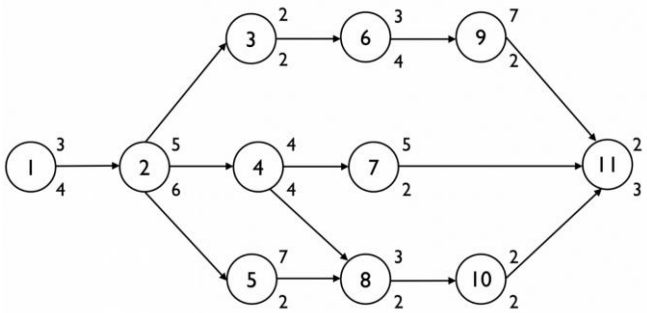

Fig 1: A Project network

The numbers above the nodes are the estimated duration of each job and the number below are the renewable resource demands. The maximum availability of the renewable resource is equal to 6 units.

Experimental Setup

We used EFT, SPT, MIS, GRWC and LNRJ priority rules for our experiment. The following are what we did:

i. We assembled SPT, MIS, GRWC and LNRJ for our algorithm (MLPR), then used each of the priority rules differently to find the activity list of the project.

ii. We used SSGS to find the schedule of EFT's activity list *when the resource is not constrained.* **(**Fig 2). This gives the earliest time the project can be completed.

iii. We then used SSGS to find the schedule of the activities using the other priority rules *when the resource is constrained* (6 units per period). (Fig 3 – Fig 7).

iv. We compared the probabilities **in** each step of the assembled rules as the steps (iterations) increases to see how each of the priority rules learns (Fig 8).

v. We compared the schedule of our algorithm with earliest finished time schedule (Fig 9). To check how well MLPR can minimize project completion times.

vi. We then finally compared the schedule of our algorithm with the schedule of all the component rules. (Fig 10). This is to check how our algorithm compares with its component algorithms in minimizing project's completion times.

vii. In Pm Knowledge Center (Table 1) the authors used 13 different priority rules to find the activity list of this project and they used SSGS and PSGS to find their schedules. The completion times ranges from 24 – 29 units.

V. RESULTS

A. *The Project Schedules*

Below are the project schedules (Fig 2 to Fig 7) of the priority rules EFT, SPT, MIS, GRWC, LNRJ and MLPR of the project network in Fig 1.

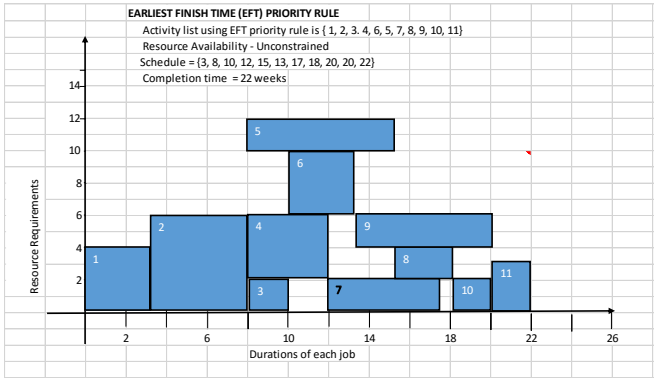

Fig. 2 : ETF priority rule Schedule using SSGS with unconstrained resource

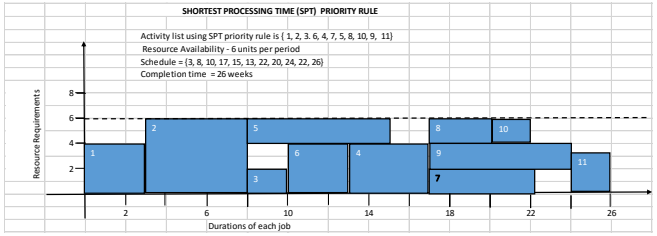

Fig. 3 : SPT priority rule schedule using SSGS with constrained resource level (6 units)

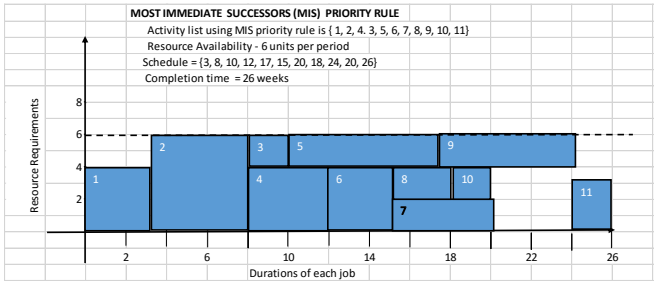

Fig. 4 : MIS priority rule schedule using SSGS with constrained resource level (6 units)

ISBN: 978-988-14048-8-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

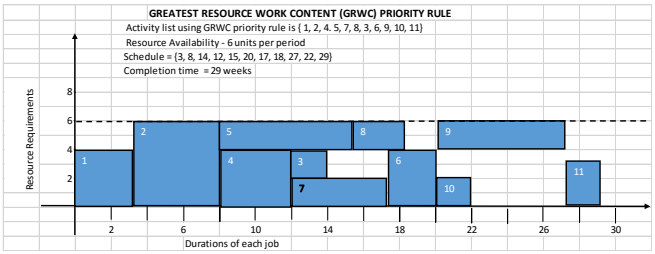

Fig. 5 : GRWC priority rule schedule using SSGS with constrained resource level (6 units)

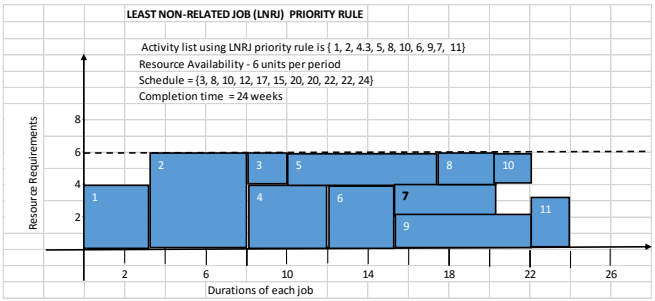

Fig. 6 : LNRJ priority rule schedule using SSGS with constrained resource level (6 units)

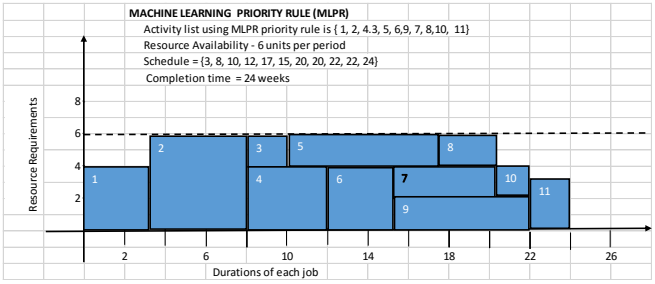

Fig. 7 : MLPR priority rule schedule using SSGS with constrained resource level (6 units)

B. *Checking how the rules learns*

In Fig 8, Series 1 to Series 4 are SPT, MIS, GRWC and LNRJ priority rules respectively.

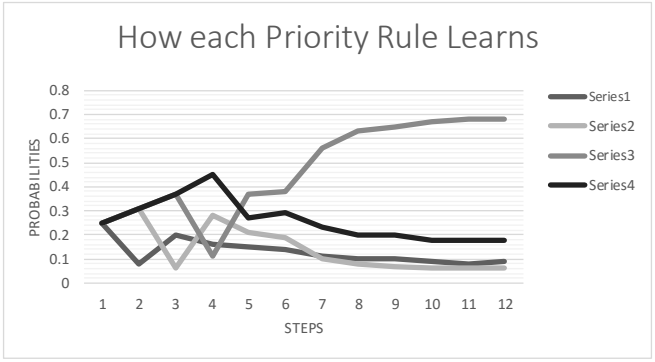

Fig 8: shows the success rates of the rules in each step.

The four priority rules SPT, MIS, GRWC and LNRJ started with probability 0.25.

SPT was chosen in the first step but its cost was too high and it was punished, so its probability was greatly reduced in the next step. MLPR continued *exploring* each of the rules until the fifth iteration when GRWC came out with the best success rate and MLPR *exploited* it to the end. This graph shows that GRWC had higher rewards from the 5th run to the end.

C. *Checking to what extent MLPR can minimize project completion time.*

We compared the schedule of MLPR with that of EFT of the project in Fig. 9. The critical path analysis under which ETF of the project is calculated assumes that resources are available in abundance. Comparing MLPR (with resource unavailability) with ETF (with available resource) is to check how good MLRP is, in minimizing the completion time of the project. Series 1 is the graph of ETF's schedule and Series 2 is the graph of MLPR's schedule.

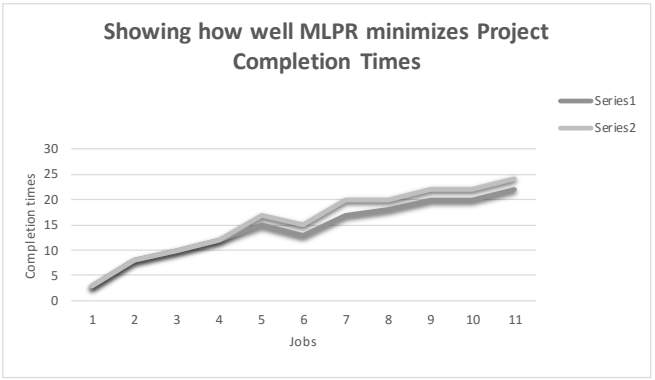

Fig 9: Comparing MLPR with EFT in jobs and project completion time

From the graph of Fig. 9 you see that the first four jobs were completed in their earliest finish times when MLPR was used. And the completion time of this project using our algorithm MLPR with constrained resource is 24 which is close to the earliest finish time of the project which is 22.

D. *Comparing MLPR with Component Rules (SPT, MIS, GRWC and LNRJ)*

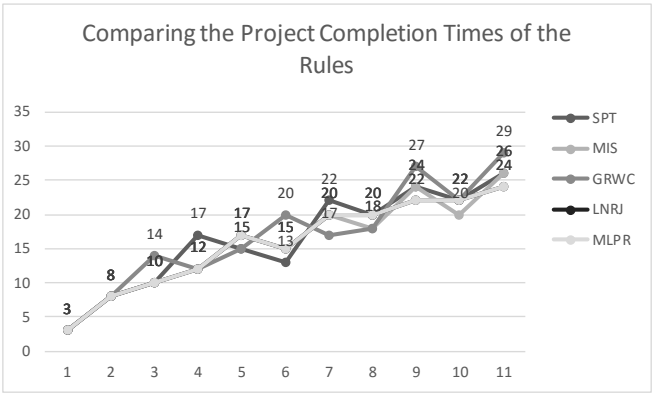

Fig 10: Comparing MLPR with its component priority rules

From Fig 10, the range of completion times of the component rules is from 24 – 29 weeks that of MLPR is 24 weeks. Hence MLPR minimizes the project completion time better than most of the component priority rules. Using MLPR makes the project to finish in 24 weeks which is minimum of all the 4 completion times in the chart. This shows that MLPR competes favorably with its component rules.

ISBN: 978-988-14048-8-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

E. Comparing MLPR with other priority rules

In Pm Knowledge Center, 13 priority rules were used to find the activity list of the project example above (Fig 10) and the jobs were scheduled using serial and parallel generation schemes. The Table 1 below shows their results.

Table 1: Priority rules and their completion times using SSGS and PSGS for the example project of Fig. 1

No.	Priority Rules	Finish Times of Project in Fig 10	
		SSGS	PSGS
1	Shortest Processing Time (SPT)	24	24
2	Longest Processing Time (LPT)	29	29
3	Most Immediate Successors (MIS)	24	24
4	Most Total Successors (MTS)	24	24
5	Least Non-Related Jobs (LNRJ)	24	24
6	Greatest Rank Positional Weight (GRPW)	27	27
7	Earliest Start Time (EST)	24	24
8	Earliest Finish Time (EFT)	24	24
9	Latest Start Time (LST)	24	24
10	Latest Finish Time (LFT)	24	24
11	Minimum Slack (MSLK)	26	26
12	Greatest Resource Work Content (GRWC)	29	29
13	Greatest Cumulative Resource Work Content (GCRWC)	26	27

We see from Table 1 that the completion times of the project using SSGS and PSGS with the 13 priority rules range from 24 – 29 weeks. This implies that the minimum time this project can be completed with 6 units' available renewable resource is 24 weeks. Fig 7 shows that the completion time of the same project using SSGS with MLPR is 24 weeks. This shows that MLPR competes favorably with other priority rules in the literature.

VI. Conclusion

This paper introduces a machine learning priority rule (MLPR) which assembles a set of priority rules and intelligently combines them using their success rates and costs. Results show that our algorithm competes very favorably with its component priority rules (Fig 10), in terms of finding the schedule that minimizes the completion time of a project. Further results got in comparing our algorithm with 9 other priority rules, making a total of 13 priority rules available (Table 1) also shows that our algorithm competes favorably with a total of 13 available priority rules in literature. Hence MLPR removes the burden of deciding which method to use because it leverages each priority rule's strengths, and is extendable to include more rules.

Acknowledgment

The paper is a benefit of sponsorship from Covenant University. The constructive suggestions of the reviewers are wholeheartedly appreciated.

References

[1] J. Blazewicz, J. K. Lenstra,, and A. H. G. Rinnooy Kan,, "Scheduling subject to resource constraints: Classification and complexity", *Discrete Applied Mathematics*, vol. 5, pp. 11–24, 1983.

[2] C. Ekenna, S. A. Jacobs, S. Thomas, N. M. Amato, "Adaptive Neighbor Connection for PRMs: A Natural Fit for Heterogeneous Environments and Parallelism", *In Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS), Tokyo, Japan,* Nov 2013.

[3] S. Hartmann,"Project scheduling with multiple modes: A genetic algorithm", *Manuskripte aus den Instituten fur Betriebswirtschaftslehre der Universit at Kiel,* no. 435, 1997.

[4] D. Hsu, G. Sa´nchez-Ante, and Z. Sun."Hybrid PRM sampling with a cost-sensitive adaptive strategy". *In Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3885–3891, 2005

[5] S. U. Kadam and S. U. Mane, "A Genetic-Local Search Algorithm Approach for Resource Constrained Project Scheduling Problem", *International Conference on Computing Communication Control and Automation*, Pune, 841-846. (*2015)* doi: 10.1109/ICCUBEA.2015.168

[6] R. Kolisch, "Serial and Parallel resource-constrained project scheduling methods revisited – Theory and Computation", *European Journal of Operations Research*, 90, 320 – 333, 1996.

[7] A. Sprecher, and A. Drexl**,** "Multi-Mode Resource-Constrained Project Scheduling Problems by A Simple General and Powerful Sequencing Algorithm", *European Journal of Operational Research.* 107, 431 – 450. 1998.

[8] F.B. Talbot, "Resource-Constrained Project Scheduling With Time – Resource Trade-Offs: The Nonpreemptive Case",. *Management Science*, 28(10):1197-1210. 1982.

[9] F.B. Talbot, J.H. Patterson," An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems", *Management Science* 24 (1978) 1163-1174.

[10] Mario Vanhoucke.(2012). Optimizing regular scheduling objectives: Schedule generation schemes. *Pm Knowledge Center*. [Online]. Available: http://www.pmknowledgecenter.com/dynamic_scheduling/baseline/optimizing-regular- scheduling-objectives-schedule-generation-schemes.