Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

IMPROVING BUSINESS PROJECT PERFORMANCE BY INCREASING THE EFFECTIVENESS OF RESOURCE CAPACITY AND ALLOCATION POLICIES

Peter H. Tag

Department of Industrial and Systems Engineering
Imagine That Inc.
6830 Via Del Oro, Suite 230
San Jose, CA 95119, USA

ABSTRACT

Resource capacity plans and allocation policies have a significant impact on the performance of business projects. This is particularly true in situations where multiple projects compete concurrently for scarce resources. Project management tools have limited ability to analyze the impact of resource allocation policies in systems with variability. Simulation tools are designed for this type of analysis. This paper focuses on simulation analyses of the relation between changes to resource capacity, resource allocation policies, variability, and project performance. Scenarios are simulated for different combinations of changes to resource quantities, work schedule durations, allocation policies, and task duration variability. Each scenario's performance is measured based on total project cycle-times and costs. The results demonstrate how increasing the flexibility of resource allocation policies can increase the effectiveness of resource capacity and significantly reduce project cycle-times without increasing project costs.

1 INTRODUCTION

Business projects range from workforce planning to disaster response to program management. Project management is the process of ensuring that results are delivered as expected while meeting a targeted deadline within a fixed budget. Companies use project management to leverage the best use of resources, reduce time to market, and handle technological complexities. Successful project management efforts result in improved performance and competitiveness (Kuhl 2008). The performance of a project can be measured and assessed using three dimensions: **time** to deliver results, **quality** of the delivered results, and **cost** to produce these results. These dimensions depend strongly on a system's resource capacity, which is a function of the quantity, schedule, and properties of the resources (funds, labor, equipment, etc.) required to perform the project. The time required to perform a project depends on having the right resources available to perform the right task at the right time. The less available required resources are to perform specific tasks, the more tasks are delayed. Also, the less proficient resources are, the longer it takes to perform tasks and the more likely tasks will need to be reworked. The more tasks are delayed and the longer it takes to perform them, the longer it takes and the more it costs to complete projects.

One of the greatest challenges of project management is determining what the resource capacity requirements are for a project. The amount of capacity required depends on the desired cycle-time for a project. A commonly used method for estimating project cycle-time is to create a project schedule network diagram. The schedule network identifies the durations, sequence, and dependencies of all the tasks in a project and allows project managers to identify all of independent work flow routes through a project. The route that dictates the cycle-time of a project is the project's critical path. In order to predict what the cycle-time of a project will be, it is important for project managers to understand the relation between resource capacity and the two main components of the cycle-time for critical path tasks: 1) **task**

execution-time and 2) **resource wait-time**. Task execution-time depends on the nature of the task being performed and the proficiency level of the resource performing the task. This can be difficult to estimate when there is little or no historical data. Resource wait-time depends on the availability of resources required to perform a task. Resource availability can be very difficult to predict when there is variability in the system. Variability is introduced into the system when actual task execution-times deviate from their estimated times, resources experience unscheduled down events, and tasks need to be reworked. System variability increases, sometimes severely, when multiple projects having task execution-time variability concurrently compete for scarce resources. Most project management tools do not address system variability and the unavailability of resources that exists in real-world environments where projects are executed (Deleris 2007). Consequently, these tools are not capable of accurately estimating the resource wait-time component of tasks and thus, project cycle-time. Simulation tools are designed to accurately represent the types of variability that impact resource wait-times. These tools can measure the tradeoffs between reduction in time waiting-for-resources and increases to task execution-time.

This paper uses simulation to create a model of an environment where multiple projects compete concurrently for scarce resources while experiencing variability.

2 LITERATURE REVIEW

The Project Management Institute (PMI) has identified ten knowledge areas for project management, including scope management, cost management, and time management (PMI 2013). A recent literature review (Patanakul 2010) of project management tools and techniques (PMTT) noted that most project management information is found in books, both academic and practical, which focus on the benefits of project management and the process of using PMTT (see Larson 2014). The review also noted that most of the current literature was specific to how PMTT would be used in each of the PMI knowledge areas.

While there is an extensive body of printed information about PMTT, few references to it in conjunction with dynamic simulation were found. Instead most referred to Monte Carlo simulation for risk management (Kwak 2007) or simulation gaming of project management cases for education (Martin 2000). The two standard texts on dynamic simulation (Law 2015, Banks 2010) do not address project management or resource allocation policies. Literature regarding policies for allocating constrained resources focuses on comparing the capabilities of various project management tools (Farid 1996) while acknowledging that the packages have deficiencies when resource scarcity is high or the number of activities is large (Trautmann 2009), as is common in the real world.

3 MODEL DESCRIPTION

A discrete-event simulation model was created to represent a hypothetical system containing the resource contention and task processing dynamics found in most project management environments. The model was created using ExtendSim because its advanced resource management (ARM) capability provides a convenient mechanism for representing task processing complexities and resource allocation policies that conditionally depend on system state (Imagine That Inc. 2015). The results of simulation scenarios were stored in the ExtendSim database (Diamond 2010) and exported to Excel at the completion of each simulation run.

3.1 Tasks

Simulation projects consist of a set of tasks that have to be performed in a specific order. Each task consists of an ordered sequence of phases: 1) wait for precedents, 2) wait for required resources, 3) process tasks, 4) release resources, and 5) spawn successors. Figure 1 shows how the block set in ExtendSim is used to represent these phases in the simulation model. The phases that govern task cycle-time (wait for precedents, wait for resources and process tasks) are outlined in blue, yellow, and red, respectively. For consistency, this time-mapping color scheme is used throughout this paper,.

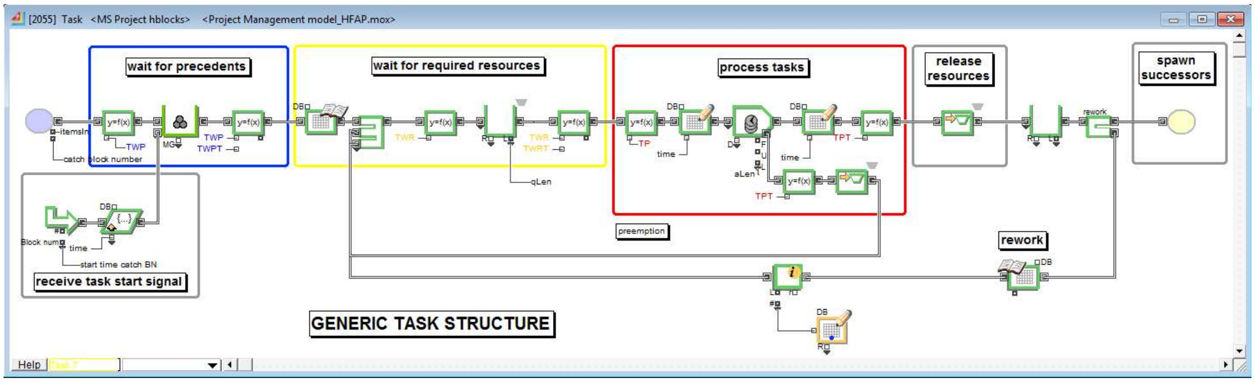

Figure 1: Generic structure used to represent the phases of a task in the simulation model.

To simplify the analysis, a hypothetical schedule network consisting of 9 tasks is used as a template for all project instances. The structure of the schedule network for these 9 tasks is shown in Figure 2.

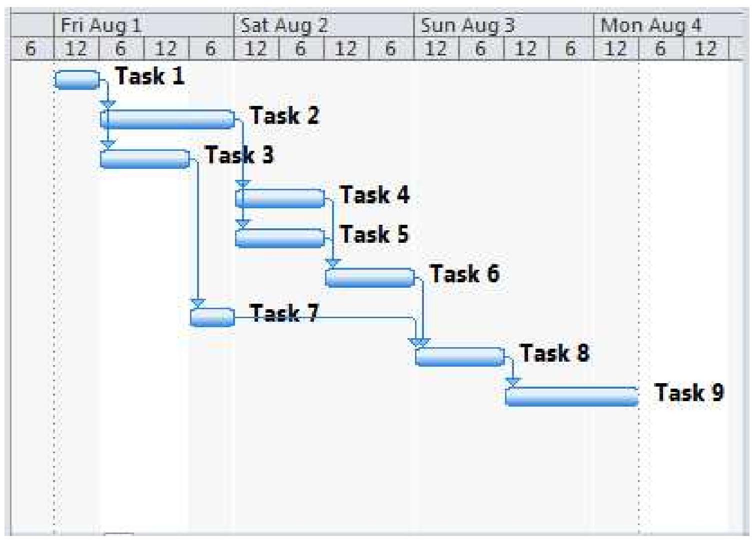

Figure 2: Schedule network.

There are three independent workflow routes through this network as indicated in Table 1.

Table 1: Workflow routes.

Route	Tasks in Route
Route 1	Task 1 - Task 3 - Task 7 - Task 8 - Task 9
Route 2	Task 1 - Task 2 - Task 4 - Task 6 - Task 8 - Task 9
Route 3	Task 1 - Task 2 - Task 5 - Task 6 - Task 8 - Task 9

Each task is assigned a mean duration value, a rework probability, and one resource drawn from a pool (see Table 2). Task durations are stochastic and calculated using log normal distributions. The mean values for these distributions vary as a function of the task type. Standard deviations are expressed as a percentage of the mean. For a given simulation run, this percentage is constant across all tasks. Reworked tasks use the same log normal distribution to determine their duration. Tasks cannot start processing until all precedent tasks have completed.

Table 2: Task assigned properties.

Task	Mean Duration (hrs)	Preferred Resource Pool		Alternate Resource Pools		Rework
		Low Flexibility Allocation Policy	*High Flexibility Allocation Policy*	*Low Flexibility Allocation Policy*	*High Flexibility Allocation Policy*	
Task 1	4	Pool A	Pool A	none	Pool B, Pool C	0.05
Task 2	16	Pool B	Pool B	none	Pool A, Pool C	0.05
Task 3	10	Pool C	Pool C	none	Pool A, Pool B	0.05
Task 4	14	Pool B	Pool B	none	Pool A, Pool C	0.05
Task 5	16	Pool B	Pool B	none	Pool A, Pool C	0.05
Task 6	12	Pool C	Pool C	none	Pool A, Pool B	0.05
Task 7	10	Pool A	Pool A	none	Pool B, Pool C	0.05
Task 8	8	Pool B	Pool B	none	Pool A, Pool C	0.05
Task 9	20	Pool C	Pool C	none	Pool A, Pool B	0.05

To understand the differences between the routes, it is useful to introduce the concept of theoretical route cycle-time. This is the minimum possible time required to perform the tasks on a route. It represents the time to complete a workflow route when there is no variability in the system, no rework, and no delays waiting for resources. For this situation, theoretical route cycle-time is equal to the sum of the mean values of the durations of tasks in the route. Figure 3 shows a graphical representation of the theoretical cycle-times for each workflow route.

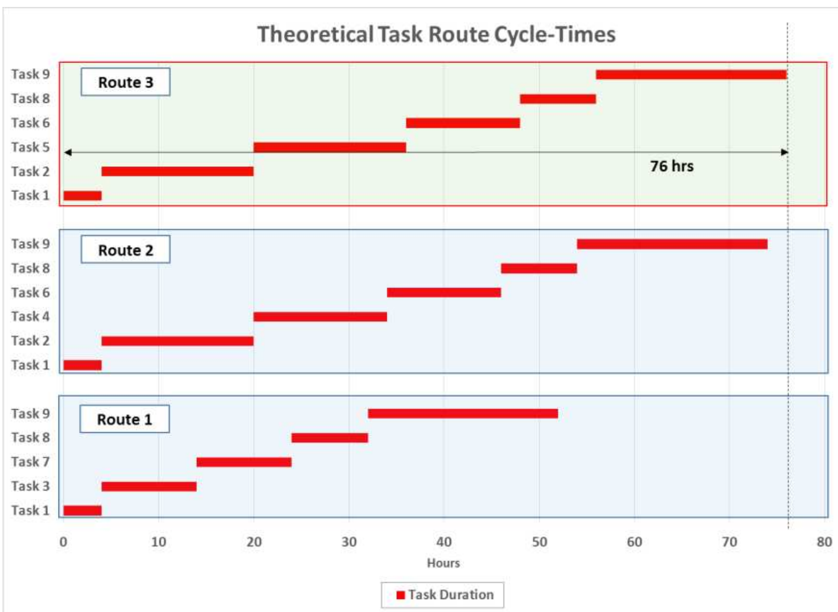

Figure 3: Theoretical cycle-times for each route.

In the theoretical system for this project, Route 3 is the critical path with a cycle-time of 76 hours. However, in the real-world, task durations have variability, rework can occur, and there is contention for scarce resources (Vaziri 2005). Depending on the level of real-world variability, it is possible for any of the three workflow routes to be the critical path of a project. When there is significant variability in the system, project management tools are ineffective for predicting which route will be the critical path. Because the real-world system being modeled in this paper uses simulation technology, it is possible to accurately represent the relation between real-world variability and project cycle-time.

3.2 Resource Capacity

In the simulation model, resource capacity is represented based on resource quantities, work schedules, and resource proficiency levels. Each resource is represented as a distinct entity belonging to a particular pool.

Pools represent resources having specific capabilities. These capabilities dictate whether a resource is capable of performing a particular set of tasks. The simulation consists of three different pools of resources. The more resources there are in a given pool, the greater the capacity of the pool.

Each resource is assigned a work schedule. Work schedules determine how many hours a resource will work each day. For the simulation experiments performed in this analysis, resources work in blocks of either 12 or 24 contiguous hours. Capacity increases as the duration of the number of hours in the work schedule increases.

Proficiency levels govern the rate at which a resource can perform a task. Proficiency levels are expressed as numeric values ranging from a low of 0.6 to a high of 1. The effective duration of a task is calculated by dividing the proficiency level of the resource performing the task into the log normal value of task's duration. Thus, a resource with proficiency level 0.6 takes 67% longer to perform a task than a resource with proficiency level 1.

Proficiency levels are also used to compute effective rework probabilities for tasks. As with effective task duration, the proficiency level of the resource performing a task is divided into the task's rework probability to generate an effective rework rate. Thus, a task having a 5% rework probability being performed by a resource with proficiency level 0.6 would have an effective rework probability of 8.3%.

As proficiency levels increase, the total capacity to perform work increases because the time required to perform the same number of tasks decreases.

3.3 Allocation Policies

As defined in ARM, allocation policies dictate which resources can be allocated to what tasks. Task resource requirements control how resources are allocated to tasks. Task resource requirements are expressed in terms of the quantity required and the pool the resource belongs to. For example, resources in Pool A might be capable of performing Tasks 1 and 7 only. In this case, the ARM requirement for Task 1 would be expressed as follows:

```
1 RESOURCE FROM POOL A
```

If resources in all three pools were capable of performing Task 1, the ARM requirement for Task 1 could be expressed as follows:

```
1 RESOURCE FROM POOL A OR POOL B OR POOL C
```

The simulation attempts to satisfy task resource requirements by searching pools from left to right in the requirement expression. For this requirement, A is the preferred pool and B and C are alternate choices (see Table 2). The order in which pools are listed opens the possibility to experiment with various resource allocation policies. Thus, in the resource requirement above, the simulation would first attempt to allocate resources from Pool A. If no Pool A resources were available, it would next look for available resources in Pool B, and finally Pool C. All simulation scenarios analyzed used task resource requirements where pools were listed in order of decreasing proficiency on a task.

3.4 Task Execution

Tasks cannot start until all of their precedent tasks are completed and the required resource is available. When the resource becomes available, it is allocated to a task for the full task duration. The execution of a task is preempted if its allocated resource goes off-shift. Preempted tasks must wait for a resource capable of performing the task to become available before the task can continue processing. The duration of continued preempted tasks is set to the time remaining when preemption occurred.

Reworked tasks must wait for required resources and require the full processing duration. Thus, a task that normally takes 10 hours to process would take 20 hours to process if reworked once.

4 ANALYSIS

Simulation experiments were run for several different scenarios, each with 10 replications. The scenarios were based on changes to variables that affect resource capacity and demand for resource capacity. These variables fall into the following categories: allocation policies, resource proficiency levels, work schedules, resource quantities, number of concurrent projects, and task duration variability (see Figure 4).

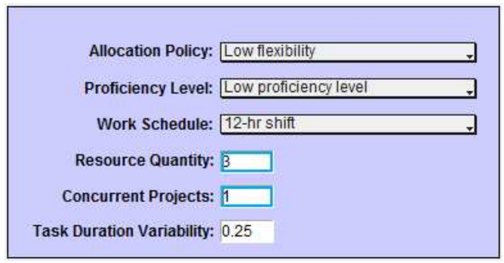

Figure 4: Simulation control panel

For all scenarios, resources are allocated to tasks in order of highest to lowest proficiency. Two levels of flexibility are used in the simulation scenarios. The low flexibility policy requires tasks to have resources allocated from their preferred pool only (see Table 2). The preferred pool has the highest proficiency level for a given task. This policy limits the number of resources available to perform a task, but ensures that the task will be performed at the fastest rate possible. The high flexibility policy allows tasks to have resources allocated from any pool that has resources with any level of proficiency on a given task. This policy increases resource availability, but can increase the time required to perform tasks.

The purpose of these experiments is to find the best project performance for the lowest cost. For this analysis, project performance was strictly measured in terms of the time required to complete projects and total costs to complete the project. To avoid performing an exhaustive full design of experiments for all possible scenario input permutations, a custom graph (see Figure 5) was created to guide the iterative process of tuning model inputs.

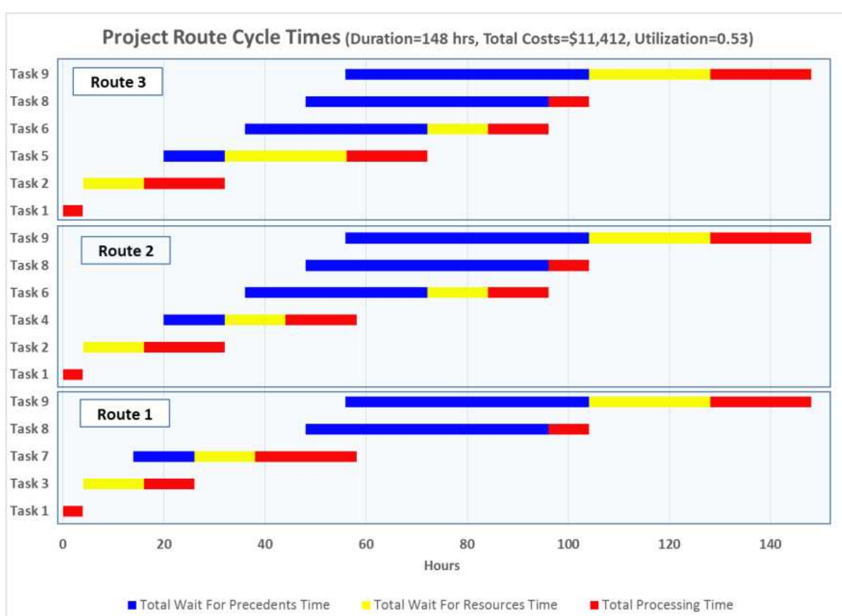

Figure 5: Cycle-time breakdown graph.

This graph illustrates how the cycle-time components of tasks are distributed over the project routes for a given simulation experiment. It represents the time each task is in each of three possible states: 1) waiting-for-precedents (blue bars), 2) waiting-for-resources (yellow bars), and 3) processing (red bars). For instance, it can be used to quickly identify the tasks on the critical path having the most resource wait-time (see the yellow bars in Figure 5).

To record the time tasks spend in the waiting-for-precedents state, each task is assigned a scheduled start time. This time is the derived from the project schedule for the theoretical situation where there is no variability and no resource constraints as shown in Figure 3. Time waiting-for-precedents occurs when a task is unable to start after its scheduled start-time because one or more of its precedent tasks have not completed. A task's waiting-for-resources time begins to accumulate if the required resources are not available and all precedent tasks have completed.

For each simulation run, the graph was used to determine which route was the critical path for a project by observing where the slack-time occurred. Slack-time occurs when a precedent task on a particular route could have finished later without delaying the start of its successor task. This is observed graphically by looking at the state of the successor task when a predecessor task is completed. If the successor task is in a wait-for-precedents state, it is still waiting for other precedent tasks and thus, slack time exists for that particular predecessor task. For the simulation run that produced the graph in Figure 5, it can be seen that Route 3 is the critical path because there is no slack time between the completion of any task and the start of successor tasks.

4.1 Analysis Process

After making several simulation runs to explore the performance space, an analysis process emerged that used the cycle-time breakdown graph to restrict input changes to only those which could increase resource capacity for critical path tasks with high resource wait-time. This process consisted of the following sequence of steps:

1. Set input values for the scenario.
2. Run multiple repetitions for the same inputs using different random seed values.
3. Determine which route was the critical path for the scenario.
4. Identify which tasks have the most time waiting for resources on the critical path route.
5. Determine which scenario inputs to change in order to increase capacity for these tasks.
6. Repeat step 1.

For example, in Figure 5 it can be observed that the majority of resource wait-time occurs at Tasks 5 and 9. For this scenario, Task 5 requires one resource from Pool B and Task 9 requires one resource from Pool C (see Table 2). Resource wait-time can be reduced by adding capacity to Pools B and C. This can be done by either adding new resources to Pools B and C or changing the work schedules of existing resources in these pools. Capacity can also be increased by making the allocation policy more flexible to enable Tasks 5 and 9 to select from a larger set of existing resources.

The strategy used for deciding how to adjust resource capacity to improve project performance was to focus on situations capable of producing resource contention. This occurs when tasks requiring the same resources are performed concurrently. From Figure 5, it can be seen that Task 2 and 3 can occur concurrently as can Tasks 4 and 5. However, only Tasks 4 and 5 require the same preferred pool (Pool B). Thus, the analysis focused on making adjustments to the resource capacity of these two tasks.

5 RESULTS

Table 3 contains a summary of results for 36 different simulation scenarios. The input variables for these scenarios consist of the flexibility level of the resource allocation policy, the proficiency level of alternate resources for task resource requirements, the work schedule, number of concurrent projects, and percentage of task duration variability. To highlight the impact of the flexibility of allocation policies, scenarios are grouped by common input values for proficiency level, work schedule, concurrent projects, and task duration variability. Each group has two scenarios: one for high allocation policy flexibility and one for low allocation policy flexibility. The outputs for each scenario consist of three primary performance variables:

1. Critical path duration.
2. Total cost to perform the projects.
3. Utilization of the resources used to during the project timeframe.

The values presented in the table for these variables represent an average across 10 simulation repetitions. Two variables, % task execution and % wait for resources, show how the two primary components of each critical path duration are distributed for each scenario. The final two output variables, % reduction resource wait time and % increase task execution time, are used to compare the high flexibility allocation policy scenario with the low flexibility scenario.

Table 3: Scenario results.

Scenario	Allocation Policy Flexibility	Proficiency Level	Work Schedule	Concurrent Projects	Task Duration Variability	Average of Critical Path Duration	Average of Total Costs	Average of Utilization	% Task Execution Time	% Resource Wait Time	% Reduction Resource Wait Time	% Increase Task Execution Time
Scenario 1a	Low	High	12-hr	1	0%	174.08	$13,512	0.41	43%	57%		
Scenario 1b	High	High	12-hr	1	0%	157.68	$12,132	0.47	51%	49%	23%	9%
Scenario 2a	Low	High	12-hr	1	25%	178.21	$13,771	0.41	42%	58%		
Scenario 2b	High	High	12-hr	1	25%	163.26	$12,789	0.47	52%	48%	24%	13%
Scenario 3a	Low	High	12-hr	2	0%	260.80	$24,151	0.48	28%	72%		
Scenario 3b	High	High	12-hr	2	0%	175.30	$14,590	0.81	43%	57%	47%	5%
Scenario 4a	Low	High	12-hr	2	25%	292.62	$26,448	0.46	26%	74%		
Scenario 4b	High	High	12-hr	2	25%	174.24	$15,251	0.77	45%	55%	56%	4%
Scenario 5a	Low	High	24-hr	2	0%	136.11	$24,422	0.46	51%	49%		
Scenario 5b	High	High	24-hr	2	0%	76.82	$14,440	0.81	87%	13%	85%	-3%
Scenario 6a	Low	High	24-hr	2	25%	139.62	$24,413	0.48	53%	47%		
Scenario 6b	High	High	24-hr	2	25%	85.39	$14,613	0.79	88%	12%	84%	1%
Scenario 7a	Low	Medium	12-hr	1	0%	177.07	$13,781	0.41	43%	57%		
Scenario 7b	High	Medium	12-hr	1	0%	159.23	$12,364	0.48	52%	48%	24%	9%
Scenario 8a	Low	Medium	12-hr	1	25%	167.52	$13,068	0.41	42%	58%		
Scenario 8b	High	Medium	12-hr	1	25%	161.64	$12,546	0.48	52%	48%	20%	19%
Scenario 9a	Low	Medium	12-hr	2	0%	263.22	$23,580	0.49	27%	73%		
Scenario 9b	High	Medium	12-hr	2	0%	181.92	$16,337	0.78	44%	56%	47%	12%
Scenario 10a	Low	Medium	12-hr	2	25%	300.32	$25,448	0.46	25%	75%		
Scenario 10b	High	Medium	12-hr	2	25%	173.36	$15,295	0.77	45%	55%	58%	4%
Scenario 11a	Low	Medium	24-hr	2	0%	133.41	$24,001	0.48	52%	48%		
Scenario 11b	High	Medium	24-hr	2	0%	84.10	$16,020	0.80	86%	14%	82%	4%
Scenario 12a	Low	Medium	24-hr	2	25%	137.29	$23,769	0.48	52%	48%		
Scenario 12b	High	Medium	24-hr	2	25%	98.31	$16,696	0.79	88%	12%	82%	22%
Scenario 13a	Low	Low	12-hr	1	0%	187.68	$14,472	0.41	42%	58%		
Scenario 13b	High	Low	12-hr	1	0%	171.68	$13,512	0.49	52%	48%	25%	14%
Scenario 14a	Low	Low	12-hr	1	25%	185.05	$14,257	0.41	43%	57%		
Scenario 14b	High	Low	12-hr	1	25%	167.03	$12,994	0.50	52%	48%	24%	10%
Scenario 15a	Low	Low	12-hr	2	0%	271.20	$23,940	0.49	26%	74%		
Scenario 15b	High	Low	12-hr	2	0%	221.90	$20,496	0.77	46%	54%	40%	42%
Scenario 16a	Low	Low	12-hr	2	25%	299.06	$25,230	0.47	26%	74%		
Scenario 16b	High	Low	12-hr	2	25%	217.03	$19,576	0.76	45%	55%	46%	26%
Scenario 17a	Low	Low	24-hr	2	0%	138.80	$24,181	0.47	51%	49%		
Scenario 17b	High	Low	24-hr	2	0%	79.73	$20,014	0.75	85%	15%	83%	-4%
Scenario 18a	Low	Low	24-hr	2	25%	133.90	$23,871	0.49	55%	45%		
Scenario 18b	High	Low	24-hr	2	25%	118.03	$21,751	0.75	87%	13%	75%	41%

For all of the scenarios in Table 3, the simulation model consisted of one resource in each of the three resource pools. The purpose for this was to focus on the impact of changes to allocation policies on project performance.

The results demonstrate that for all 18 scenario groups, critical path durations and total costs are lower when the most flexible allocation policies are used. This is because high flexibility policies reduce total resource wait time substantially more than they increase task execution time. Even in the worst case scenario where alternate resources have the lowest proficiency levels, the benefits of reducing resource wait-time substantially outweigh the penalty of increasing task execution-time.

Figures 6 and 7 compare the impact of low and high flexibility allocation policies on project performance for resources working 12 hour work schedules. The differences between these policies are most pronounced when multiple projects are concurrently executed. This is because contention for scarce resources sharply increases when two projects concurrently compete for one resource in each pool. The increased resource contention amplifies whatever variability exists in the system (Ignizio 2009).

Figure 6 shows results for one project executed in isolation. The graph illustrates a modest, but consistent, improvement in performance when using high flexibility allocation policies. The impact of proficiency levels is buried within the noise of the variability of the system being modeled.

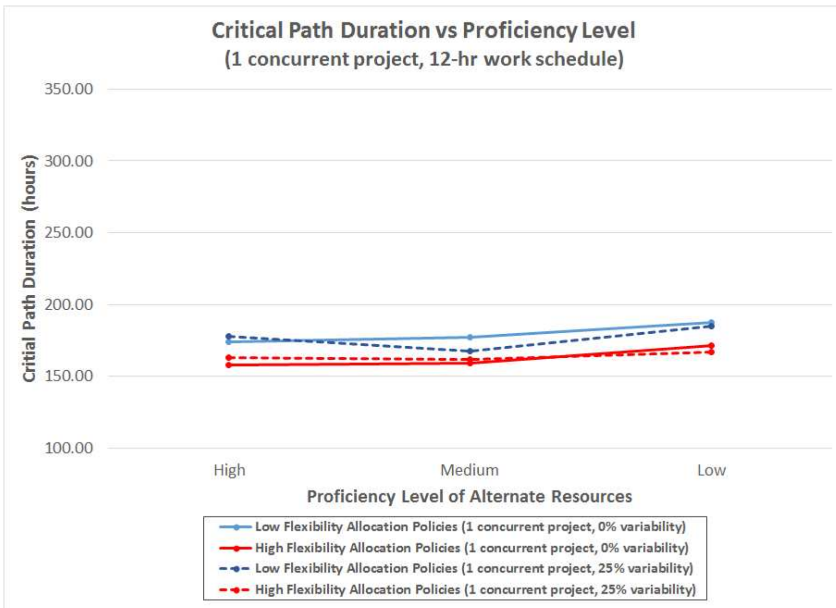

Figure 6: Critical path duration vs. resource proficiency level for one concurrent project.

Figure 7 shows that when variability is amplified, the performance is substantially reduced when low flexibility allocation policies are used. However, high flexibility allocation policies can be observed to actually mitigate the impact of variability. In these situations, high flexibility allocation policies dramatically reduce resource wait-time. This is because high flexibility policies make more resources available to perform all of the tasks in the project at all times. One of the consequences of increased allocation policy flexibility is increased resource utilization. For the system being modeled, the increased resource utilization effectively reduces project cycle-time. This is because the increases to task execution-time are more than offset by the gains from reducing resource wait-time. Even when resource proficiency is low, performance improvement is still substantial.

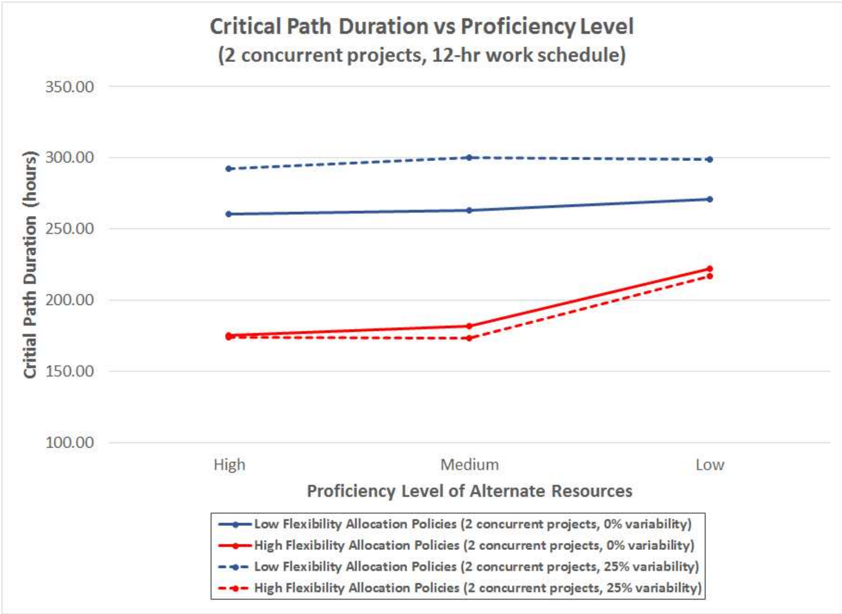

Figure 7: Critical path duration vs. resource proficiency level for two concurrent projects.

The cycle-time breakdown graphs in Figures 8 and 9 clearly illustrate the performance improvement resulting from using high flexibility allocation policies. In these figures, it can be seen that even though high flexibility policies cause task execution times to increase, particularly for Task 9, the dramatic reduction in the time tasks wait for resources ultimately produces much better performance than for low flexibility policies.

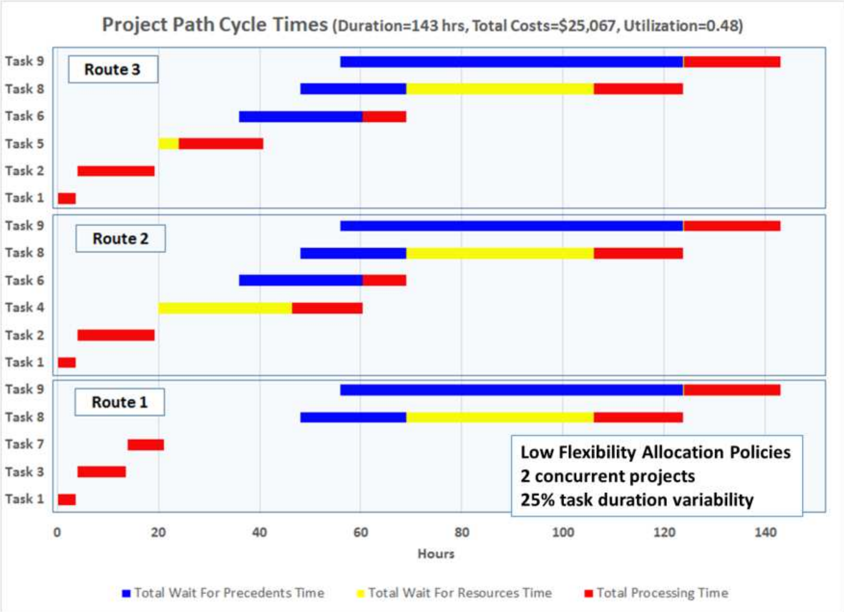

Figure 8: Cycle-time breakdown graph for low flexibility allocation policies.

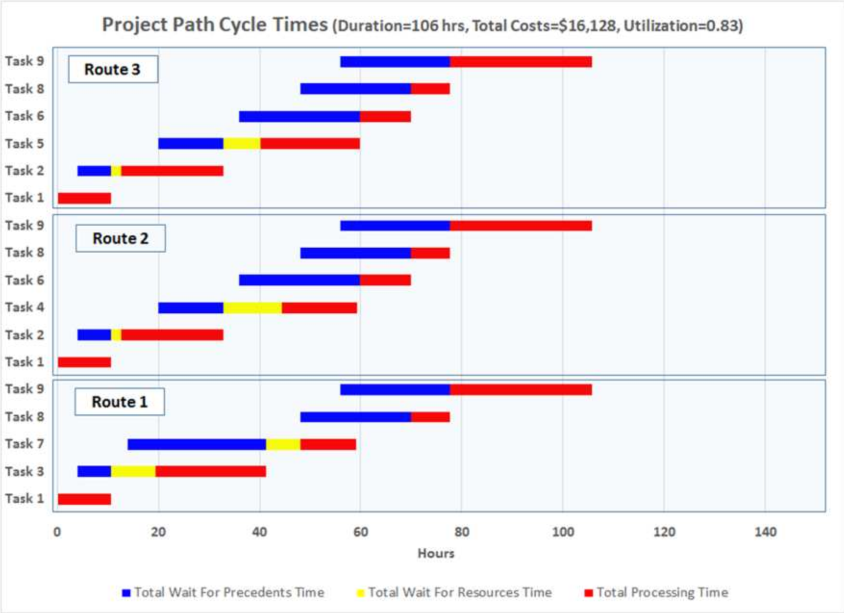

Figure 9: Cycle-time breakdown graph for high flexibility allocation policies.

6 CONCLUSION

Flexible resource allocation policies are necessary when the availability of preferred resources is not sufficient to complete a project within the allowable or expected time frame. Flexible allocation policies increase resource utilization and thereby reduce resource wait-time, a key component of project cycle-time. However, the gains from reduced resource wait-times could be negated by increased task execution times. Simulation provides a means to rigorously measure the impact of this tradeoff for different allocation policies over a range of different conditions and variability levels. Since most project management tools provide users with minimal capabilities for customizing how automatic resource allocation occurs, they are unable to accurately represent many of the policies that can be used in the real-world. Consequently, the effectiveness of these tools for planning resource capacity and predicting project schedules is severely limited. As the results of this paper indicate, one promising direction for the evolution of project management tools is tighter integration with simulation technologies and in particular, the resource management features of these technologies. This integration could open new frontiers in the project management domain for exploring aspects of project performance that were beyond the scope of this paper. In particular, little work has been done to measure the impact of resource allocation policies on the quality of project results. The experiments presented in this paper could readily be expanded to provide the statistical rigor required to enable analysis of new and previously perceived as risky strategies that could significantly improve project performance.

REFERENCES

Banks, J., J. S. Carson II, B. L. Nelson, and D. M. Nicol. 2010. Discrete-Event System Simulation. 5th ed. Upper Saddle River, New Jersey: Pearson Education, Inc.

Deleris, L. A., S. Bagchi, S. Kapoor, K. Katircioglu, R. Lam, and S. Buckley. 2007. “Simulation of Adaptive Project Management Analytics.” In Proceedings of the 2007 Winter Simulation Conference, edited by S. G. Henderson, B. Bille, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton. 2234-2240. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Diamond, B., D. Krahl, A. Nastasi, and P. Tag. 2010. “ExtendSim Advanced Technology: Integrated Simulation Database.” 2010. In Proceedings of the 2010 Winter Simulation Conference, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 32-39. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Farid, F. and S. Manoharan. 1996. “Comparative Analysis of Resource-Allocation Capabilities of Project Management Software Packages.” Project Management Journal. 27:35-44. Newtown Square, Pennsylvania: Project Management Institute.

Ignizio, J. P. 2009. Optimizing Factory Performance: Cost-Effective Ways to Achieve Significant and Sustainable Improvement. New York, New York: McGraw-Hill Professional.

Imagine That Inc. 2015. ExtendSim 9.2 User Guide, San Jose, CA.

Kuhl, M. and R.Tolentino-Peña 2008 “A Dynamic Crashing Method for Project Management Using Simulation-Based Optimization.” In Proceedings of the 2008 Winter Simulation Conference, edited by S.J. Mason, R.R. Hill, L. Monch, O. Rose, T. Jefferson, and J.W. Fowler, 2370-2376. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Kwak, Y. H. and L. Ingall 2007. “Exploring Monte Carlo Simulation Applications for Project Management.” Risk Management 9.1:44-57. Hants, UK: Palgrave Macmillan.

Larson, E. W. and C. F. Gray. 2014. Project Management: The Managerial Process. 6th ed. New York, New York: McGraw-Hill/Irwin.

Law, A. M. 2015. Simulation Modeling and Analysis. 5th ed. New York, New York: McGraw Hill Education.

Martin, A. 2000. “A Simulation Engine for Custom Project Management Education.” International Journal of Project Management, edited by J. R. Turner. 201-213. Amsterdam, Netherlands: Elsevier B.V.

Patanakul, P., B Lewwongcharoen, and D. Milosevic. 2010. “An Empirical Study on the Use of Project Management Tools and Techniques Across Project Life-Cycle and Their Impact on Project Success.” Journal of General Management 35: 41-65. Oxforshire, UK: Braybrooke Press Ltd.

PMI 2013. A Guide to the Project Management Body of Knowledge. Newtown Square, Pennsylvania: Project Management Institute.

Trautmann, N. and P. Baumann. 2009. “Project Scheduling with Precedence Constraints and Scarce Resources: An Experimental Analysis of Commercial Project Management Software.” In Operations Research Proceedings 2008, edited by B. Fleishmann, K-H Borgwardt, R. Klein, and A. Tuma, 165-170. Berlin, Germany: Springer-Verlag.

Vaziri, K., L. K. Nozick, and M. A. Turnquist, 2005. “Resource Allocation and Planning for Program Management.” In Proceedings of the 2005 Winter Simulation Conference, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Johns, 2127-2135. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHY

PETER H. TAG is a simulation architect at Imagine That Inc. He received an M.S. in geophysics with a minor in theoretical and applied mechanics from Cornell University and a B.S. in geology with a minor in philosophy from Lehigh University. Mr. Tag has worked as a consultant for several companies including Apple, Boeing, Florida Power & Light, Intel, Johnson & Johnson, Los Alamos National Laboratory, Pratt & Whitney, Procter & Gamble and Raytheon. His email address is petert@extendsim.com.