ABSTRACT


Improving productivity and reducing work in process (WIP) are two activities that can improve project performance, but are often difficult to accomplish. The construction process is complex and consists of a large number of interdependent and sequential tasks. Construction activities are simulated as serial production lines (SPLs) using STROBOSCOPE to investigate the effects that different working strategies (push/pull and balanced/unbalanced) have on productivity and WIP. The sensitivity to task duration variance is also examined for the different working strategies. The simulation results are used to compare the models based on their performance. Results show that in presence of duration variance, push strategies result in higher productivity, while pull strategies result in lower amounts of WIP. Pull strategies are more sensitive to the presence of duration variance. It is also shown that WIP is a function of the coefficient of variation. The results of simulations are used to compare the tradeoffs between strategies. The findings will help construction managers to better understand the effects of working strategies and conditions to decide which working strategy is best suited for their needs.

The Oops game demonstrates the value of a reliable planning for construction activities by providing easy to follow game rules. The game uses two strategies. The first strategy is to build without planning. Although we do not need to pay for planning in the beginning, when oops occurs we will have to pay a higher price in the building phase. The second strategy is to pay for planning before build. This means that we will need to pay for planning upfront, but it saves us more in the building phase because reliable planning reduces the chances of oops. The cost and
value of planning depends on the complexity level of the project, the amount of uncertainty that we need to deal with, and the price for oops. We use computer simulation to show how the value of planning is sensitive to those factors. The results show that the cost for a reliable planning is justified by reducing the total cost of project when there is probability of oops. To validate the results from simulation, a case study was performed by comparing two projects executed by the same mechanical subcontractor. The projects were managed in accordance with the aforementioned strategies. The project with the reliable planning strategy had a 35% higher productivity and a benefit-cost ratio of 13:1. The contribution of the Oops game lies in providing a simple and effective means to show the value of planning to construction practitioners and students. The research findings fill in the gap of body of knowledge in quantifying the value of plan under different circumstance for construction projects.
Effects of Working Strategy and Duration Variance on Productivity and Work in Process
A Simulation-Based Investigation
and
Oops Game: Cost-Benefits Tradeoff Analysis of Reliable Planning for Construction Activities

by
Kiarash Hajifathalian

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Master of Science

Civil Engineering
Raleigh, North Carolina

2011

APPROVED BY:

__________________________________________  ____________________________________________
Dr. Simon M. Hsiang                        Dr. Michael L. Leming
Co-Chair of Advisory Committee

__________________________________________  ____________________________________________
Dr. Min Liu
Co-Chair of Advisory Committee
BIOGRAPHY

Kiarash Hajifathalian was born June 12, 1986 in Tehran, Iran. He received his Bachelor of Science degree in Civil Engineering at University of Tehran (UT) in 2009. During his undergraduate degree he served as teaching assistant in the Civil Engineering Department at UT. He was a member of the team in charge of defining concepts and descriptions of urban land use and urban capitation for Iran’s Urban Planning and Design Code.

In August 2009, he enrolled in the Construction Engineering and Management program at North Carolina State University (NCSU) to pursue his Master of Science Degree. While studying at NCSU he served as research assistant and teaching assistant in the Civil, Construction, and Environmental Engineering Department. During his graduate career he was honored with induction into the Phi Kappa Phi honor society.

Kiarash’s areas of interest are lean construction, construction financial and risk management, and structural engineering. He is a member of the American Society of Civil Engineers and Associated General Contractors of America.
ACKNOWLEDGEMENTS

This thesis would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, my utmost gratitude to Dr. Min Liu, whose sincerity and encouragement I will never forget. Dr. Liu supported me throughout my thesis with her patience, motivation, and knowledge. Without her, this thesis would not have been completed or written.

I offer my sincerest gratitude to Dr. Simon M. Hsiang, who has supported me throughout my thesis with his patience, enthusiasm, and immense knowledge.

My sincere thanks also goes to my committee member, Dr. Michael L. Leming, for the time and effort he has devoted to make this thesis possible.

It is a pleasure to thank my friend, Brad W. Wambeke, who has been my support and inspiration as I hurdle the obstacles in the completion of this research work.

I would also like to thank all my professors in the Construction Engineering and Management program at North Carolina State University for the knowledge and experience they have shared with me. They are worth more than I could ever repay.
Last but not the least, I would like to thank my family: my parents Hadi and Zohreh, and my brother Kaveh for supporting me throughout my life. “I can no other answer make, but, thanks, and thanks.”
TABLE OF CONTENTS

LIST OF TABLES...........................................................................................................vii

LIST OF FIGURES........................................................................................................viii

EFFECTS OF WORKING STRATEGY AND DURATION VARIANCE ON PRODUCTIVITY
AND WORK IN PROCESS: A SIMULATION-BASED INVESTIGATION.........................1

INTRODUCTION.............................................................................................................1

Construction Works as Serial Production Lines.......................................................1

Push vs. Pull..................................................................................................................2

Balanced vs. Unbalanced............................................................................................3

Duration Variance.......................................................................................................3

Examples in Construction Practice.............................................................................4

Underlying Issues.......................................................................................................5

LITERATURE REVIEW.................................................................................................7

Impacts of Variability in Construction Work ...........................................................7

Use of Simulation........................................................................................................8
METHODOLOGY..................................................................................................................8

Simulations and Results...................................................................................................12

Mean-Variance Analysis of the Results.........................................................................23

CONCLUSIONS................................................................................................................27

REFERENCES..................................................................................................................29

OOPS GAME: COST-BENEFITS TRADEOFF ANALYSIS OF RELIABLE PLANNING FOR CONSTRUCTION ACTIVITIES.................................................................33

INTRODUCTION..............................................................................................................33

RESEARCH IN STUDYING THE VALUE OF PLAN........................................................34

A RELIABLE WORK PLAN..............................................................................................35

TEACHING GAMES IN CONSTRUCTION....................................................................36

GAPS IN KNOWLEDGE..................................................................................................37

GAME DESCRIPTION......................................................................................................38

Strategies to Play the Game..........................................................................................40

Items for Group Discussion..........................................................................................43

CLASS SIMULATION......................................................................................................45
COMPUTER SIMULATION .................................................................................. 46

DISCUSSION OF THE RESULTS ........................................................................ 47

Cost of Project .................................................................................................. 47

Performance Reliability .................................................................................. 49

Cost of Project vs. Cost of Oops and Cost of Planning ...................................... 51

Risk Assessment Matrix .................................................................................. 53

VALIDITY CHECK ............................................................................................. 55

CONCLUSIONS .................................................................................................. 59

REFERENCES ..................................................................................................... 62

APPENDICES ...................................................................................................... 65

Appendix A. Stroboscope Simulation Code – 3 Work Station Push ....................... 66

Appendix B. Stroboscope Simulation Code – 3 Work Station Pull ......................... 72

Appendix C. Stroboscope Simulation Code – 5 Work Station Push ....................... 80

Appendix D. Stroboscope Simulation Code – 5 Work Station Pull ......................... 89

Appendix E. MATLAB Code – Oops Game “Plan” Strategy .................................. 98

Appendix F. MATLAB Code – Oops Game “Build” Strategy .................................. 101
LIST OF TABLES

Summary of Results for Productivity Improvement........................................14

Summary of Results for WIP Decrease..........................................................18

Summary of Research Objectives and Simulation Analysis Results..................20

Summary of the Teaching Games in Construction.........................................37

Example of a Completed Score Sheet..........................................................40

Summary of In-class Results of Oops Game..................................................46

Results of Computer Simulations of Oops Game..........................................47

Case Study Projects....................................................................................57
LIST OF FIGURES

SPL Model with 3 Work Stations ................................................................. 9
SPL Model with 5 Work Stations ................................................................. 10
Productivity in Push vs. Pull and Balanced vs. Unbalanced SPLs .................. 13
Productivity sensitivity in Push Balanced systems .......................................... 16
Productivity sensitivity in Push Unbalanced systems ..................................... 16
Productivity sensitivity in Pull Balanced systems .......................................... 17
Productivity sensitivity in Pull Unbalanced systems ..................................... 17
Percent WIP for Push vs. Pull and Balanced vs. Unbalanced SPLs .................. 19
WIP sensitivity in Push Balanced systems ..................................................... 21
WIP sensitivity in Push Unbalanced systems ................................................. 21
WIP sensitivity in Pull Balanced systems ..................................................... 22
WIP sensitivity in Pull Unbalanced systems ................................................. 22
Mean-Variance Frontier for Productivity/ \( \frac{1}{\log_{10} WIP \times 10} \) \( \approx 1 \) ................................................................. 26
Mean-Variance Frontier for Productivity/ \( \frac{1}{\log_{10} WIP \times 10} \) \( \approx 2 \) ................................................................. 26
The Project

Game Configuration and Cost Basis

Flow Diagram “Plan” Strategy

Flow Diagram “Build” Strategy

In-class Results

Cost of Project vs. Number of Cards

Cost Frequency for 4 Card Game

Cost Frequency for 9 Card Game

Cost Frequency for 16 Card Game

Balanced Lines for 4, 9 and 16 Card Game

Risk Assessment Matrix

Risk Assessment for Oops Game

Project Productivity Performance for LPS® and non- LPS® Projects
INTRODUCTION

There are numerous factors, techniques, and procedures intended to improve the performance of construction projects. Conventional methods for improving productivity have focused on the reduction of idle time for workers. Oglesby et al. (1989) conducted activity sampling to observe and identify idle time for workers. They subsequently modified the working procedure to minimize the idle time and improve productivity. More recent studies have investigated productivity while viewing construction as a systematic work procedure consisting of different work stations (Tommelein et al. 1999; Koskela 2002). This research is focused on improving productivity and reducing work in process (WIP) by studying push vs. pull strategies, balanced vs. unbalanced production lines, and task duration variances to determine their impacts on both productivity and WIP. The results are useful for project managers by providing a better understanding of the impacts of working strategies and conditions on the performance of the projects.

Construction Works as Serial Production Lines

Precedence networks are commonly used in the construction industry due to the sequential nature of many of the activities. For example, some typical sequential activities associated with a building’s structure may include: excavation, forming and placing a concrete foundation, erecting a structural steel frame, placing and securing the flooring and roof decking,
and enclosing the structure. Most of these activities involve an internal series of sequential tasks that are often repetitive, which liken them to a serial production line (SPL). The SPL is a term used in manufacturing that describes a single-line production system that produces large amounts of standardized products (Womack et al. 1990). This paper is not suggesting that construction as a whole can be characterized by mass manufacturing; however, there are some repetitive tasks within the construction industry that are relatively similar to a SPL. This research uses the SPL while examining the effects push vs. pull strategies, balanced vs. unbalanced lines, and task duration variances have on productivity and WIP.

**Push vs. Pull**

In a “push” working strategy, each crew or “working station” in the SPL works to finish their assignments and has little or no concern about the state of successive working stations (Hopp and Spearman 1996). The schedule drives the requirement for work to be completed in a push system. For example in an excavation process, the excavator may continue to work regardless of whether or not there is a dump truck available. If there was no truck available, the excavator would simply create a stockpile as the excavator is driven by the schedule to excavate a specific area. In a “pull” strategy, work stations perform with respect to their successor’s conditions and needs, and only produce work when their successor needs their product (Hopp and Spearman 1996). The customer’s need drives the requirement for work to be completed in a pull system. Consider the excavation example again. In a pull system, the empty truck could be viewed as the customer; therefore, the excavator would wait until there was a truck to fill instead of creating a stockpile that may need to be moved again in the future. The pull strategy is similar
to the “Just in Time” (JIT) strategy commonly used in manufacturing (Ohno 1987). The JIT strategy implies that units of work do not wait to be worked on by the next working station. The implementation of JIT strategies in manufacturing has resulted in the reduction of cost, hence increasing competitiveness of production companies (Akintoye 1995; Pheng and Chan 1997; Pheng and Tan 1998; Pheng and Hui 1999). This research aims to quantify and investigate the effects of the pull strategy on productivity and WIP and compares it to the push strategy that is commonly practiced in construction.

**Balanced vs. Unbalanced**

The second aspect considered in this research has to do with whether a SPL is balanced or unbalanced. The balancing is in relation to the relative time it takes to complete tasks within the SPL (i.e. task duration). If the tasks for the working stations of a SPL have relatively equal task durations, the SPL is considered balanced. On the other hand, if the work stations are responsible for tasks with different durations, the SPL is considered unbalanced.

**Duration Variance**

The third aspect investigates the effects of duration variance by using STROBOSCOPE simulation models. Koskela (2000) and Tommelein et al. (1999) stated that in reality, activities do not have a deterministic duration and the fluctuation in duration causes variability in the process. The details of how the duration variance is simulated are discussed in the methodology section. In this paper “variability” and “variation” refer to the uncertainty in the process. “Variance” refers to the statistical definition which is the deviations of the measurements from
their “mean” (Ott and Longnecker 2001).

**Examples in Construction Practice**

To have a better understanding of pull and push systems, balanced and unbalanced SPLs, and to see how these different characteristics can exist in construction projects, the following examples are provided:

1. **Pull Balanced Strategy:** Consider the construction of a structural steel building in a congested work site (i.e. the construction of a high rise office building in a downtown area). The management has decided to follow a Pull (JIT) strategy due to limited storage area for steel sections. The steel is scheduled to be delivered to the site on a daily basis. A crew is assigned to attach hooks and rigging devices to lift the steel directly off the delivery truck with a crane. A second crew is waiting at installing points to connect the steel sections to previously erected parts of structure and fasten the initial connection, and a third crew follows to install and tighten all the bolts. Steel is not removed from the truck until all three crews are prepared. The contractor is experienced and skilled in steel erection and knows the production rate of each crew; thus has balanced the crews such that all three have relatively equal production rates and can work at a smooth rhythm with minimal idle time.

2. **Pull Unbalanced Strategy:** Consider an earth moving job site which has a several loaders and dump trucks working together. Due to availability, there are not enough dump trucks (i.e. the customers of the loader) to keep the loader working at a steady pace. The work rates of the loaders and the cycle times for the trucks to dump their loads and return to get filled again are
different; therefore, the system is unbalanced. Since the loader’s work is governed by the need of a customer (i.e. the dump truck), the loader is following a pull strategy.

3. Push Balanced Strategy: Consider an activity of installing dry wall that consists of three major tasks; which are measuring, cutting, and installing the gypsum boards. An experienced sub-contractor may assign one person to measure studs dimensions, one laborer to cut the gypsum boards, and two workers to install them. The workers doing the measuring and cutting might work well ahead of the installers because they are more concerned with their tasks. Carpeting and floor tiling crews are other examples that might use a push strategy. Since this work is often repetitive with minor changes, the crews can perform in a balanced manner.

4. Push Unbalanced Strategy: The typical work sequence for a structural concrete member involves forming the member; cutting, shaping, and placing the rebar; and placing and vibrating the concrete. The task duration for each of these activities is different, thus making the work sequence unbalanced. Each task is done independently from the others, as long as the resources are available and the preceding task has been completed. So, the forming crew follows a push schedule to stay ahead of the rebar and concrete crews. Likewise, the rebar crew follows a schedule to stay ahead of the concrete crew.

Underlying Issues

Several researchers have looked at individual aspects associated with the SPL strategies. However, according to our knowledge, little research has put these areas together to determine
how they behave in conjunction with one another. For example, Tommelein et al. (1999) studied the effects of variability in SPLs for pull strategies and concluded that production rate fluctuation creates a starvation for resources and a noticeable loss in productivity. To complement and understand the contrast, there is a need to evaluate the sensitivity of the four previously described working strategies to the presence of duration variance. A STROBOSCOPE simulation model was developed to represent and investigate the SPLs in order to address the following six research objectives:

1- In terms of productivity:
   a. Does a pull or push SPL result in higher productivity?
   b. Does a balanced or unbalanced SPL result in higher productivity?
   c. How sensitive is the productivity for the four strategies (pull-balanced, pull-unbalanced, push-balanced, and push-unbalanced) to task duration variance?

2- In terms of WIP:
   a. Does a pull or push SPL result in lower WIP?
   b. Does a balanced or unbalanced SPL result in lower WIP?
   c. How sensitive is the WIP for the four strategies (pull-balanced, pull-unbalanced, push-balanced, and push-unbalanced) to task duration variance?

The results of this research will be useful to project managers as they decide which strategy is best for them to implement in their project. Determining which strategies are best suited (where and when) so that the project can be completed expediently is a production
management task, which is typically handled by construction superintendents. While they may appear to be effective at managing certain trades, regrettably, their goal is not necessarily to accommodate the planned productivity rates of all trades involved.

LITERATURE REVIEW

Impacts of Variability in Construction Work

Tommelein et al. (1999) used pull balanced SPLs to study the effects of work flow variability on production variability. They concluded that production lines cannot meet their full capacity in the presence of flow variability since the variability causes the work stations to be starved for resources. This variability can be a result of several items, such as: working on open job sites with different conditions, the uniqueness of each project, and the complexity of construction works (Salem et al. 2006). The variability causes instability in the balanced system as the activity durations are no longer equal. Researchers have also examined the variability of the process itself, defined by Koskela (2000) as the random variation in the processing time or duration and the arrival of resources. The nature of construction work implies that variability in process can be seen in two main types, starting time and task duration. The arrival or starting time variation is the difference between the planned and actual starting time and the task duration variation is the difference between the planned and actual task duration (Wambeke et al. 2010). Due to the sequential nature of activities and the need to have a crew finish its tasks in order for the next crew to start its work, there is an accumulative effect for the variation. Each working crew’s arrival variability is sum of its predecessor’s arrival and duration variability. Thus
duration variation is an important factor in arrival variation. Other factors such as readiness of equipment, material, plans, permits, etc. may influence the starting time of activities and can be highly variable between different projects, trades, managerial methods, etc. These items are all important as they affect the initiation of the working sequence; however, research has not been conducted to determine how their combined effects impact the productivity and WIP of SPLs.

Use of Simulation

Simulation modeling method has been widely used in construction research for almost two decades (Vanegas et al. 1993; Huang et al. 1994; Smith et al. 1995; Shi and Abourizk 1998; Tommelein et al. 1999; Mohamed et al. 2007; Thomas Ng et al. 2009; Wang et al. 2009). The advantages of simulation lie in the generality and flexibility. In addition, simulation is powerful to represent different behaviors of systems under different conditions and it also makes it convenient and feasible to measure and investigate the outputs (Kant 1992; Schelasin and Mauer 1995; Martinez and Ioannou 1997). STROBOSCOPE is a simulation programming language developed by Martinez (1996) and has been widely used for modeling discrete events in construction (Ioannou and Martinez 1996; Martinez and Ioannou 1999; Tommelein et al. 1999). Therefore STROBOSCOPE is selected to build simulation models for this research.

METHODOLOGY

STROBOSCOPE is used to simulate the SPLs and address the objectives of this research. Models were simulated using both three and five working stations to determine if the results
varied depending on the size of the SPL (Figures 1 and 2). Numbers of working stations (3 and 5) were chosen considering the characteristics of specialty works in construction which usually have a number of different tasks in that range. Previous researchers have also used this number of tasks to represent the specialty works in construction (Tommelein et al. 1999). In Figures 1 and 2 the circular shapes represent the queuing area of the resources which can be labor, equipment, material, WIP, or the final product. The rectangular shapes represents working stations (crews) in which the actual work is being performed. Modeled SPLs were each simulated for 1000 iterations to ensure reliable outputs, while considering the stochastic activity durations and the use of the probability distribution functions. To model duration variance for the activities, probability distribution functions (PDFs) is used to represent activity durations. Past studies have investigated different PDFs that can be used for construction activities duration including the Normal (Clemmens and Willenbrock 1978), the Lognormal (Al-masri 1985), the Beta (Farid and Koning 1994), and the Triangular (Gonzalez-Quevedo et Al. 1993). In this research, the Normal distribution was used to model duration variance due to its effectiveness in construction simulation (Maio et al. 2000) and also its generality.

![Figure 1 – SPL Model with 3 Work Stations](Image)
To address the research objectives, both push and pull oriented SPLs are modeled while considering both a balanced and an unbalanced working sequence. Additionally, the activities within the SPLs are subjected to duration variance to determine how sensitive the productivity and WIP are to these conditions.

The push strategy is modeled by enabling the work stations to produce as long as they are provided with the required resources. Each work station’s output becomes a necessary input resource for its successor’s work. If a successor is busy working, produced units are stored in WaitPiles between the two work stations. In the pull strategy, a work station only works if there is no resource for its successor to work on (i.e. the WaitPile after the work station is empty). When the successor starts working on its last resource (i.e. the WaitPile between the two stations is now empty), the work station starts to work to produce more resources (i.e. needs) for its successor (i.e. its customer).

Normally distributed durations are created in the model to represent the task duration at each of the work stations. The SPL is balanced when all of the work stations have the same average duration; otherwise, the SPL is unbalanced. It is also important to note in both the
balanced and unbalanced SPLs, the total time of the process remains constant.

To investigate how sensitive the various SPL working strategies are to duration variance, each simulation is repeated with different levels of mean duration and duration variance. Duration variance is introduced in the model as a standard deviation from the mean duration of each working station.

The goal of simulation in each combination of strategies and conditions is to collect productivity and WIP results for the SPL. Productivity for the SPL is the total time divided by work units produced in that time. WIP is the time weighted average of WaitPile units that exist during the working period of the SPL. Equations 1 and 2 are provided for determining productivity and WIP respectively. Productivity is a simple calculation of input / output; therefore, a calculation example is not included. Note that in the simulations in order to calculate productivity the total duration to produce 100 units is studied. An example is provided to illustrate how the WIP is calculated.

\[
\text{Productivity} = \frac{T_s}{O_s} \quad \text{(Eqtn 1)}
\]

\[
\text{WIP} = \sum_{i=1}^{N_{ws}} \frac{W_i \times T_{wi}}{T_s} \quad \text{(Eqtn 2)}
\]

Where,

\[ T_s = \text{Total duration to produce } O_s \]
O_s = Total units of output produced in simulation, which is 100

T_{w_i} = Duration in which W_i units exist in WaitPile number i

W_i = units in WaitPile number i for the duration of T_{B_i}

n = Maximum number of units present in WaitPile number i

N_{ws} = Total number of work stations in the SPL

To calculate the WIP of the SPL, a time weighted average of WIP for each WaitPile between two work stations is calculated. These WIPs are then summed to determine the WIP for the entire SPL. For example, if a WaitPile contained 30 units for 4 hours and 70 units for 3 hours, the time weighted average of WIP for the WaitPile would be:

\[ WIP = \frac{30 \times 4 + 70 \times 3}{7} = 47.14 \]

This calculation is performed for all the WaitPiles in the SPL and then the results are summed in order to calculate the total number of WIPs of that SPL. It should be noted that Since WIP value of 5 when the total number of units produced is 20 does not equal the same amount when the total number of units produced is 200, the simulation expresses WIP quantities as percentages of the total number of units produced in order to normalize the WIP quantity.

**Simulations and Results**

The results of the SPL simulations are organized in accordance with the six research
Objectives.

Objective 1a: Does a pull or push SPL result in higher productivity?

Figure 3 shows the productivity of each system. The productivity is represented along the vertical axis of the figure in terms of input/output (i.e. lower is better). The push system has a better productivity than the pull system. The reason is that when variability is present in a SPL, a work station may starve for resources if there are not enough units in the WaitPiles between the two work stations. In a pull system, WaitPiles cannot stock up WIP; thus, resulting in starvation for resources. This is consistent with the research of Rybkowski et Al. (2008). In a push strategy, each work station continues its production independent from its successors needs, which results in piles of units between the work stations. These buffer units enable the work stations to continue to work even when their task durations are subjected to the variability that is inherent in the PDFs upon which they were modeled.

Figure 3 - Productivity in Push vs. Pull and Balanced vs. Unbalanced SPLs
**Objective 1b: Does a balanced or unbalanced SPL result in higher productivity?**

Figure 3 is also used to show that balanced SPLs have better productivity than unbalanced SPLs. If we assumed duration of construction work was deterministic (i.e. activities had no variability), a balanced and unbalanced SPL would result in the same productivity since the total duration of the work would simply be the sum of the durations for each work station. Although assumption of deterministic duration for the activities does not represent the reality, precedence networks commonly use this assumption. This research illustrates the importance of creating balanced work flow in presence of duration variance to achieve greater productivity. Table 1 shows a summary of the productivity differences for push vs. pull and balanced vs. unbalanced. For example push strategies result in 13.54% more productivity in a balanced SPL with 3 work stations.

<table>
<thead>
<tr>
<th>Productivity Improvement (Push vs. Pull)</th>
<th>3 ws Balanced</th>
<th>3 ws Unbalanced</th>
<th>5 ws Balanced</th>
<th>5 ws Unbalanced</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>13.54%</td>
<td>5.38%</td>
<td>24.61%</td>
<td>14.16%</td>
<td>14.42%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Productivity Improvement (Balanced vs. Unbalanced)</th>
<th>3 ws Push</th>
<th>3 ws Pull</th>
<th>5 ws Push</th>
<th>5 ws Pull</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12.56%</td>
<td>4.46%</td>
<td>26.59%</td>
<td>15.97%</td>
<td>14.89%</td>
</tr>
</tbody>
</table>

**Objective 1c: How sensitive is the productivity to task duration variance for the four strategies (pull-balanced, pull-unbalanced, push-balanced, and push-unbalanced)?**

To examine the productivity sensitivity of the different working strategies to duration variance, a more thorough investigation was conducted. Changes in productivity were measured
while the duration variance was reduced in incremental steps. The combined effect of a reduction in duration variance and mean duration was also investigated to examine the productivity trade-off between these two parameters. Figures 4 through 7 depict the productivity sensitivity of each of the four SPLs to duration variance. The strips in the graphs represent a range of productivity improvement. SPLs with productivity improvement strips that are more parallel to the x-axis (i.e. horizontal) represent a lower sensitivity to duration variance. The results show that productivity in push systems (Figures 4 and 5) is less sensitive to duration variance than it is in pull systems (Figures 6 and 7). The results also indicate that unbalanced push systems are the least sensitive to changes in the duration variance. The sloped strips in Figures 6 and 7 indicate the feasibility of improving productivity by reducing the duration variance. As one moves from left to right on the graph, multiple productivity improvement strips are crossed. Reducing the mean duration of the activities improves productivity in both the push and pull strategies, which is fairly intuitive.
Figure 4 – Productivity sensitivity in Push Balanced systems

Figure 5 – Productivity sensitivity in Push Unbalanced systems
Figure 6 – Productivity sensitivity in Pull Balanced systems

Figure 7 – Productivity sensitivity in Pull Unbalanced systems
Objective 2a: Does a pull or push SPL result in lower WIP?
Objective 2b: Does a balanced or unbalanced SPL result in lower WIP?

Pull SPLs generally experience only about 1% WIP regardless of the number of work stations or whether the system is balanced (Figure 8). This is significantly lower than the WIP for a push SPL and not surprising as pull systems are designed to not generate large amounts of WIP. Figure 8 also shows that balanced SPLs have lower WIP than unbalanced SPLs and that the number of work stations impacts the WIP. The WIP increases from about 9% to 28% when the number of unbalanced push work stations increases from three to five. This demonstrates that being unbalanced has an adverse effect on WIP and this effect is magnified as the number of work crews increases. Table 2 shows a summary of the WIP decreases for pull vs. push and balanced vs. unbalanced. For example push strategy results in 471% (4.7 times) more WIP in a balanced SPL with 3 work stations.

<table>
<thead>
<tr>
<th>WIP Increase</th>
<th>3 ws</th>
<th>3 ws</th>
<th>5 ws</th>
<th>5 ws</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push vs. Pull</td>
<td>471%</td>
<td>691%</td>
<td>663%</td>
<td>2730%</td>
<td>1139%</td>
</tr>
<tr>
<td>Unbalanced vs. Balanced</td>
<td>118%</td>
<td>57%</td>
<td>364%</td>
<td>25%</td>
<td>141%</td>
</tr>
</tbody>
</table>

Objective 2c: How sensitive is the WIP to task duration variance for the four strategies (pull-balanced, pull-unbalanced, push-balanced, and push-unbalanced)?

Figures 9 through 12 show the sensitivity of WIP to duration variance and mean duration. Unlike productivity sensitivity, which was a function of how fast (i.e. the mean duration) and how reliable (i.e. the duration variance) each working station performed, WIP sensitivity is a
function of the coefficient of variation (CV). The CV is the ratio between the duration variance and mean duration as shown in Equation 3.

\[
CV = \frac{\text{Duration Variance}}{\text{Mean Duration}} \quad \text{(Eqtn 3)}
\]

This was found to be valid regardless of whether the SPL was push, pull, balanced, or unbalanced. The slopes of WIP strips in the graphs are almost 45 degrees which implies that by decreasing or increasing the mean duration and the duration variance simultaneously, there is no change in the amount of WIP. However, the percent of WIP decreases when the duration variance is reduced or the mean duration is increased or both (i.e. the CV is reduced). Therefore, reducing the CV of the SPL will also reduce the WIP.

Figures 11 and 12 show that pull SPLs, whether balanced or unbalanced, generally have
less than 1% WIP, which was consistent with Figure 8. If the intent is to lower WIP, but implementing a pull strategy is not feasible, the first option should be to balance the process and then attempt to reduce the CV for two reasons. First, balancing the process has a greater impact on reducing the WIP than reducing the CV, as was shown in Figure 8. Second, a balanced system is more sensitive to changes in the CV than an unbalanced system. This can be seen in Figures 9 and 10. The WIP is reduced by about 50% in the balanced SPL (Figure 9) when the CV is decreased. On the other hand, the WIP remains relatively constant in the unbalanced SPL (Figure 10) when the CV is decreased. Since there are many research objectives and multiple figures associated with the analysis, a summary of the results is provided in Table 3 below.

Table 3 – Summary of Research Objectives and Simulation Analysis Results

<table>
<thead>
<tr>
<th>Research Objective</th>
<th>Simulation Analysis Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Does a pull or push SPL result in higher productivity?</td>
<td>Push has higher productivity</td>
</tr>
<tr>
<td>Does a balanced or unbalanced SPL result in higher productivity?</td>
<td>Balanced has higher productivity</td>
</tr>
<tr>
<td>How sensitive is the productivity to task duration variance for the four strategies (pull-balanced, pull-unbalanced, push-balanced, and push-unbalanced)?</td>
<td>Balanced more sensitive than Unbalanced Pull more sensitive than Push</td>
</tr>
<tr>
<td>Does a pull or push SPL result in lower WIP?</td>
<td>Pull has lower WIP</td>
</tr>
<tr>
<td>Does a balanced or unbalanced SPL result in lower WIP?</td>
<td>Balanced has lower WIP</td>
</tr>
<tr>
<td>How sensitive is the WIP to task duration variance for the four strategies (pull-balanced, pull-unbalanced, push-balanced, and push-unbalanced)?</td>
<td>WIP = f(CV) Balnced more sensitive than Unbalanced Push more sensitive than Pull</td>
</tr>
</tbody>
</table>
Figure 9 – WIP sensitivity in Push Balanced systems

Figure 10 – WIP sensitivity in Push Unbalanced systems
Figure 11 – WIP sensitivity in Pull Balanced systems

Figure 12 – WIP sensitivity in Pull Unbalanced systems
Mean-Variance Analysis of the Results:

A Mean-Variance analysis was performed to compare the results of the simulated working strategies based on their productivity and WIP. This analysis was based on combining two measures of performance (i.e. the productivity and WIP) for the four different SPL strategies into one measure of success (i.e. measure of return: $R_p$). The average $R_p$ is calculated as shown below in Equation 4.

$$R_p = X \times \text{Productivity} + (1 - X) \times \frac{1}{\log_{10} WIP \times 10}$$  \hspace{1cm} (Eqtn 4)  \hspace{1cm} (Markowitz 1987)

Where,

$R_p$ = Average measure of return for the simulation, based on Productivity and WIP

Productivity = Average Productivity for different activity durations and standard deviations of duration in one working strategy

WIP = Average WIP for different activity durations and standard deviations of duration in one working strategy

$X$ = Weight of Productivity in calculation of $R_p$

Since $R_p$ is an average, its variance is calculated using Equation 5:

$$\sigma_p^2 = X^2 \times \sigma_{\text{Prod}}^2 + (1 - X)^2 \times \sigma_{\text{WIP}}^2 + 2 \times X \times (1 - X) \times \sigma_{\text{Prod}} \times \sigma_{\text{WIP}} \times \rho_{\text{Prod,WIP}}$$  \hspace{1cm} (Eqtn5)  \hspace{1cm} (Markowitz 1987)
Where,

\[ \sigma_p = \text{Variance of } R_p \]

\[ \sigma_{\text{Prod.}} = \text{Variance of Productivity} \]

\[ \sigma_{\text{WIP}} = \text{Variance of } \frac{1}{\log_{10} WIP \times 10} \]

\[ \rho_{\text{Prod.,WIP}} = \text{Covariance of Productivity and } \frac{1}{\log_{10} WIP \times 10} \]

\[ X = \text{Weight of Productivity in calculation of } R_p \]

To normalize WIP quantities, \( \frac{1}{\log_{10} WIP \times 10} \) is used instead of WIP because the WIP quantities have a vast range and using the Log function constricts that range. Furthermore, since mean variance analysis assumptions are based on measures of success that are intended to be maximized, as opposed to WIP that is intended to be minimized, it makes sense to invert the WIP quantities.

Both \( R_p \) and \( \sigma_p \) were plotted to show the average combined return and the associated uncertainty for each working strategy and condition combination. Since productivity and WIP may not always have the same importance for managers, 2 different graphs are provided for productivity over WIP ratios of 1 and 2 (Figures 13 and 14 respectively). Each line in the graphs represents the relationship between \( R_p \) and \( \sigma_p \) for the corresponding policy (i.e. push balanced, push unbalanced, pull balanced, pull unbalanced). The segment of line that is thicker for each
policy should be focused on for analysis. If there are two possible $R_p$ values with the same variance (i.e. $\sigma_p$), the one with the higher return is preferred. It is important to note that Figures 13 and 14 need to be compared relative to each other. Figure 13, in which WIP is of relatively higher importance, shows that the four policies would be ranked as follows (lines with higher $R_p$):

1. Pull Balanced
2. Pull Unbalanced
3. Push balanced
4. Push Unbalanced

Figure 14, in which WIP is of relatively lower importance, and more emphasis is on productivity depicts that the ranking for policies is changed. In this situation, if we consider the highest return for each policy (i.e. the peak point for each line), the new rankings are as follows:

1. Push Balanced
2. Pull Balanced
3. Push Unbalanced
4. Pull Unbalanced

It can be seen from both Figures 13 and 14 that for the same $R_p$, pull strategies generally have less variance meaning that are more reliable to reach the intended return. The graphical
representation is that pull strategies are to the left of push strategies for a specific $R_p$. The same concept can be seen for balanced versus unbalanced working conditions. Balanced working conditions show more reliability in the return measure ($R_p$).

Figure 13 - Mean-Variance Frontier for Productivity/ $\frac{1}{\log_{10} WIP \times 10} \approx 1$

Figure 14 - Mean-Variance Frontier for Productivity/ $\frac{1}{\log_{10} WIP \times 10} \approx 2$
CONCLUSIONS

This paper investigates the effects different working strategies (push vs. pull and balanced vs. unbalanced) and task duration variance have on productivity and WIP. Additionally, a mean variance analysis was conducted to compare the various working strategies. The results demonstrates that in presence of duration variance, push strategies result in higher productivity (14.4 % in average) at the price of larger amount of WIP. Pull strategies result in considerably less WIP (1139% or 11 times less in average), which subsequently reduces storage space and inventory cost. Sensitivity of the strategies and conditions to duration variance is analyzed and it is demonstrated that productivity in pull strategies is more sensitive to duration variance. Also it is shown that WIP is not sensitive to duration variance; it is sensitive to CV of the duration. Moreover mean-variance analysis is used to summarize the result and to rank the different strategies and conditions. It was shown that when WIP is of relatively more importance pull strategies are better than push, and among each strategy balanced SPLs have higher return. On the other hand if the WIP is not of high importance the ranking is changed and push balanced strategy has the best return, followed by pull balanced.

Common economically driven practice is to minimize the WIP, while maximizing the productivity. The results of the simulations show that in sequential and repetitive activities, working strategies impact these two measures. Moreover results show that there is a trade-off between these two measures (productivity and WIP) that needs to be considered. A better understanding of working strategy’s effect on the performance will help managers in the decision making process. For example the results show that having a balanced working sequence
improves both WIP and productivity, regardless of whether there is a push or pull system in place. This knowledge can be useful to managers because they can improve productivity and reduce their WIP by focusing on balancing the resource allocation for their tasks. Consider forming and ironwork for a shear wall. A 6 man crew finishes 30 sqft/hr (60 sqft of form is needed for 30 sqft of rebars) and a 5 man crew finishes 40 sqft/hr of formwork, this result in 10 sqft of extra rebar in an hour that needs to be formed later. Adding one person from ironwork crew to formwork crew, reduces production rate of ironworkers to 25 sqft/hr and and increases the formworkers productivity to 48 sqft/hr, this balancing will reduce the WIP by 9 sqft/hr and now only 1 sqft of extra work is produced by ironworkers in an hour. This balancing will also improve the combined productivity of the crews as it was shown in the simulations.

Further research work can focus on comparing the working strategies for specific job site conditions and working processes and can be validated by collecting quantitative data from the construction process. This research is valuable because it creates a better understanding of the strengths of the different working strategies. Managers can use the results of this research to determine which system is best suited to meet their goals of improving productivity, while minimizing WIP.
REFERENCES


York.


operations”, Computer aided civil and infrastructure engineering, 13(2), 121-130.


OOPS GAME: COST-BENEFITS TRADEOFF ANALYSIS OF RELIABLE PLANNING
FOR CONSTRUCTION ACTIVITIES

INTRODUCTION

There has been an increasing interest for a paradigm shift in construction planning in the past two decades. In common practice, project managers identify and sequence activities based on what should be done to finish the project assuming the tasks that should be completed are ready to work on when their time is up. Superintendents and foremen are required to commit to plans that are given to them without a thorough consideration of the resources needed and the constraints that challenge them from executing the plans. As a result, it is common that a crew is not able to complete all of its assignments. Researchers have shown that less than 60% of the activities on a weekly work plan are typically completed (Kim and Jang 2005, Ballard et al. 2007, Ballard and Howell 1998, Ballard et al. 1994, Ballard et al. 1996). It is advocated to use a structured process to generate a reliable work plan. For example, the Last Planner System (LPS®) provides a powerful tool to have routine meetings and checking process to make sure constraints are removed before execution of a task (Ballard 1994, Choo et al. 1999, Ballard and Howell 1998). Consequently, productivity and the percentage of task completion rate increases.

There is more time, effort, and cost required in the process for a reliable plan. According to our observations and interviews with construction project managers, superintendents, and foremen, there is a common understanding that we do not need or do not have time to have more planning meetings. It is especially true under the current downturn of economy, construction companies
work under small fees and cannot afford to pay people to have more planning meetings. Therefore, it is important to answer the question: is it necessary to spend additional time in planning and how much does a reliable work plan worth?

The goal of this research is to identify the costs and benefits of a reliable work plan. First, the concept of reliable weekly planning and its requirements are discussed. Next, the “Oops game,” a teaching game designed to compare the plan and build approaches, is described. The results of controlled in-class experiments and computer simulations are presented. In the last section, a case study is investigated which compares two different planning approaches in construction projects. A mechanical subcontractor performed two similar, but separate projects. One project used the LPS®, while the other did not. The research findings add to current body of knowledge in this field by helping project managers and engineers to better understand the benefits of reliable weekly planning; and to further show the real life gains, relevance, and importance of implementing such strategy in the construction industry by providing the data from a case study.

RESEARCH IN STUDYING THE VALUE OF PLAN

Value of planning has been studied in depth in many fields such as socio-economics, world development, operation research, corporate management, and business venturing (Blumstein and Cassidy 1973; Brada et al. 1983; Bock and Hoberg 2007; Camillus 1975; Gruber 2007). However, according to our knowledge, little research has been done in quantifying the value of a reliable plan for construction projects. Austin et al. (1999) studied the value of detailed and reliable planning in design phase. They stated that common planning practice takes little account
of the interdisciplinary and iterative nature of the building design process and this leads to a compromised design process containing inevitable cycles of rework. Subsequently they proposed Analytical Design Planning Technique (ADePT) to generate the project-specific models in an acceptable time scale. However, they did not quantify the value of planning in design phase or measure the value of the proposed technique. Work on the value of planning in construction phase is mainly done in the field of lean construction. Ballard and Howell (1998) gathered over 450 weeks of percent plan completed (PPC) data from seven different companies and found they had an average PPC of 54%; thus illustrating unreliability of performance is certainly present in the construction industry. As a result, Ballard (1994) created the LPS®, in which the last planner (typically the foremen) develops the weekly work plan by using a 6 week look-ahead process to ensure constraints on successful task execution are identified and removed. The benefits of using LPS® have been mainly investigated in terms of improving PPC (Ballard 2000, Ballard et al. 1996, Ballard and Howell 1998, Kim and Jang 2005). Howell et al. (2001) suggested that reducing variation to improve planning reliability from 50% to 70% can theoretically increase productivity by 30% (i.e. from 50% to 65%). But to our knowledge there has been little work done in using real project data to quantify the value of planning and investigating the benefit-cost trade off of a reliable planning for construction projects.

A RELIABLE WORK PLAN

Ballard and Howell (1998) propose that an effective and reliable weekly work plan needs to meet five specific quality requirements: Definition (precise and unambiguous defining of
assignments), Soundness (what is necessary to get the prerequisites and resources ready), Sequence (constructability order, priority order and workable backlog), Size (matching to the productive capability of each crew), and Learning (assignments that are not completed are tracked and the reasons identified). Cohenca-Zall et al. (1994) studied the involvement of different parties in construction projects using three matrices for: information gathering, development of alternatives and choice-making. In each matrix they considered factors such as schedule, cost, cash flow, major equipment, layout and logistics, work methods, manpower and material allocation. They found that superintendents and foremen are by far the most involved person in a project, which highlights the importance of including their opinions to have a reliable work plan.

**TEACHING GAMES IN CONSTRUCTION**

Due to the existence of multiple interrelated factors and variables in case studies it is very difficult to investigate the effects of one single variable. Therefore it is hard to quantify the relative magnitude of benefits of implementing a new strategy or principle (Rybkowski et al. year). As Detty and Yingling (2000) argued, implementing concepts and principles based on previous experience of others (i.e. case studies) requires a “faith based justification.” Therefore, researchers have tried to use other means to study and demonstrate the effects of implementing new methodologies. The most frequent approaches are using computer simulation models, designing teaching games, and a combination of both. Table 1 shows a summary of the teaching games in construction. In this paper, a combination of both a game and computer simulation is
used to address the research objective.

While computer simulations offer a high level of control on models, which enable the results of different approaches to be compared and analyzed, some may argue the validity of such models in accurately representing the reality. Controlled experimentation can be used to test, calibrate, and validate computer models. In this research the Oops game has been played in classroom conditions according to the scientific controlled experimentation guidelines by Bernard (2000) to validate and calibrate the computer simulation.

Table 1 - Summary of the Teaching Games in Construction

<table>
<thead>
<tr>
<th>Games</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Construction management game – Deterministic model</td>
<td>Au et al. (1969)</td>
</tr>
<tr>
<td>Road construction – Simulation game for site managers</td>
<td>Harris and Evans (1977)</td>
</tr>
<tr>
<td>LEAPCON: Simulation of Lean Construction of High-Rise Apartment Buildings</td>
<td>Sacks et al. (2007)</td>
</tr>
<tr>
<td>The Airplane Game As A Lean Simulation Exercise</td>
<td>Rybkowski et al. (2008)</td>
</tr>
</tbody>
</table>

GAPS IN KNOWLEDGE

Since there is additional planning time and effort required to generate a reliable work plan, on construction practice, we often face defensive reaction by management of many companies towards a change in their approach to planning projects. Companies are unsure whether the expected additional costs and efforts involved with a more planning can be justified by the benefits gained. Thus, there is a need for a quantified analytical approach of showing the benefits of reliable planning using measures that are tangible to construction practitioners such as cost.
and productivity. Furthermore, there is a need for an effective while simple tool, a teaching game, to illustrate the benefits and costs of planning and the adverse effects of not doing so. The objectives of this research is to address these two issues by providing Oops game as a means to show the mechanics of effectiveness of reliable planning and to present cost and productivity data based on a case study on a construction activity.

GAME DESCRIPTION

The Oops Game was developed by Greg Howell as a teaching game which is easy to play with student in a classroom setting. Students are divided into teams of three members. Each team represents a contractor in charge of building the “Project.” The project to be constructed is represented by the arrangement of 9 cards as shown in the Figure 1. The project is built in 9 increments. One of the players is in charge of handling the cards in order to build the project, the second player is in charge of keeping the score, and the third player watches for errors. The assignments can be adjusted if there are different numbers of players in each group. The configuration of the project is shown in Figure 2 and the game begins after the cards are well shuffled and placed face down in the “Yard.” The first card is turned over and placed in its appropriate spot on the project. The rule of the game is that after the first card is placed, additional cards can only be placed on the project if they share an adjacent edge with the card(s) already in place. For example if the first card is a 1, then only a 2 or a 4 can be added to the project because they are the only two cards that share an edge with card 1, 5 cannot be added because it does not share an edge. After the first card is placed, and before turning the next card
over, players have two options: 1) to “Build” the card by attempting to place it on the project, or 2) to “Plan” the card by turning it over without necessarily placing it on the project. An “oops” occurs when the players decide to build and the card does not share an adjacent edge with the card(s) already on the project. The cost of each choice and the outcome is shown in Figure 2. A score sheet is provided for the players to easily keep the score of each game played. As it can be seen in Figure 2, if a card is directly built on the project it costs $1, if the card is planned and then built its total cost is $2, and if a cards is attempted to be built but an oops occurs its total cost is $4. Players record which of the three situations occur for each card that is turned over on their score sheet. The total instances of each situation are added and multiplied by its cost for each of the columns in the score sheet. The sum of the three costs represents the total project expense. Table 2 shows an example of a completed score sheet.

![Figure 1 – The Project](image-url)
Strategies to Play the Game

Each team is required to play the game following two different strategies (since this game is governed by randomness, each team is required to play the game 10 times for each of the strategies):
- Plan
- Build

**Plan:** in this strategy teams are asked to plan the cards before attempting to build (after the first card is turned over). This strategy represents a reliable planning process for the activities before execution. The reliable planning in the game is analogous to ensuring that all the prerequisites for an activity are successfully met before attempting to take action. It has to be made clear to the players that in this strategy after a certain point there is no need for planning since all the remaining spots in the project share an adjacent edge with a card already built. After reaching this point, the players are not required to plan and the cards can directly be built. This represents the reality in construction (like in any other activity) that the uncertainty is decreased as we get closer to finish. At the beginning there are a lot of unknowns and uncertainties (i.e. a lot of empty spots in the project) that may affect our performance. As we go further the uncertainty is reduced step by step (after each card is built). Figure 3 shows a flow diagram for the steps that players need to take in the planning strategy. It is recommended to provide a copy of the flow diagram for the players to refer to.
Build: In this strategy teams are asked to attempt to build the cards without any planning, with the risk of an oops occurring. This strategy represents the situation in which a reliable planning process for each step of construction process is not performed; thus resulting in occurrences in which a crew is not capable of performing its duties because of some external factor such as unavailability of equipment or material, poor specifications that need correction or alteration, etc.
Figure 4 shows a flow diagram for the steps in the build strategy.

Figure 4 – Flow Diagram “Build” Strategy

Items for Group Discussion

When all teams have completed their play, the teams should discuss their findings. Issues to discuss are divided into two categories: (1) Analysis of the results and (2) Real life reflections and implications.
Analysis of the Results:

- Which strategy in average results in lower cost of completion for the project? Does it match their anticipation?
- Which strategy has a more predictable outcome? Which strategy has lower variance in outcomes?
- Which factors can change the lower costing strategy from one to another?
- How would changing cost of planning, building and oops impact the results?
- What would changing the number of cards in the game result in? If we play with 4 or 16 cards instead of 9.

Real Life Reflections and Implications:

- How does this game relate to reality? Provide examples of causes in construction that may prevent timely execution of activities.
- How do you interpret the following statistics?
  - Maximum cost for each strategy? (The difference between the maximum cost and the average cost for the strategies represent the amount of possible cost overruns)
  - The average cost of strategies? (In the long run, the difference between the two cost averages is the amount of savings that one can gain using the lower average cost strategy)
  - The standard deviation of the results? (A lower standard deviation in cost shows a more predictable performance. This reliability in performance will result in more
reliable bid estimates and subsequently will help reduce the contingency cost that is considered in the estimates)

- What construction activities would you consider a 4 card game (low probability of oops)? Also provide examples for 9 and 16 card games (higher probabilities of oops)

- In real life what factors impact the cost of planning, building and oops?

- If you are in a decision making position for a project would you choose the planning strategy or building? What factors will you consider in your decision?

**CLASS SIMULATION**

The live simulation results were measured through 76 executions of the game for the planning strategy and 85 executions for the no planning strategy in an in-class controlled experiment. The results of all the groups were entered in an excel file to show the statistics of the results to students. For a better representation it is recommended to provide frequency graphs for students based on their results. The statistics are subsequently used to help the game coordinator and the students in the discussion. Table 3 shows a summary of the result for in class executions of the game. As it can be seen the average cost for completion of the project is lower for planning strategy. Figure 5 shows the histogram for the results of the game and it can be seen that the planning strategy yields more reliable results. A more detailed investigation and discussion of the results is presented in the computer simulation section.
Table 3 – Summary of In-class Results of Oops Game

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Average Cost</th>
<th>Standard Deviation of Cost</th>
<th>Min Cost</th>
<th>Max Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plan</td>
<td>14.14</td>
<td>1.21</td>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>Build</td>
<td>16.31</td>
<td>4.80</td>
<td>9</td>
<td>27</td>
</tr>
</tbody>
</table>

![Figure 5 – In-class Results](image)

**COMPUTER SIMULATION**

While the live results of classrooms can be efficiently used to address the teaching point of the game, computer simulation helps us investigate the problem in depth. The game was simulated for more detailed discussions of the results using MATLAB. Games of 4, 9, 16, 25, 36, 49, 64, 81, 100, and 400 cards are simulated to see if the results follow a trend as the complexity level increases. Simulations are run 1000 times for each strategy of each set of card play due to randomness of the game. Simulation models follow the exact same steps shown in flow diagrams of Figures 3 and 4. Table 4 shows the summary of results of computer simulations for
the Oops game. For each set of card play average, standard deviation and coefficient of variation (CV=STDEV/AVG) of total cost of project is calculated based on 1000 iterations. In order to examine the statistical difference between the results of two strategies, paired t-test (α=0.05) was performed. All the calculated p-values were zero and the two strategies were significantly different for each set of card play.

Table 4 - Results of Computer Simulations of Oops Game

<table>
<thead>
<tr>
<th># Of Cards</th>
<th>Total Cost (Plan)</th>
<th>Total Cost (Build)</th>
<th>Normalized Cost Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AVG</td>
<td>STDEV</td>
<td>CV</td>
</tr>
<tr>
<td>4</td>
<td>5.33</td>
<td>0.47</td>
<td>0.09</td>
</tr>
<tr>
<td>9</td>
<td>14.22</td>
<td>1.23</td>
<td>0.09</td>
</tr>
<tr>
<td>16</td>
<td>26.95</td>
<td>1.48</td>
<td>0.06</td>
</tr>
<tr>
<td>25</td>
<td>43.04</td>
<td>2.00</td>
<td>0.05</td>
</tr>
<tr>
<td>36</td>
<td>62.90</td>
<td>2.79</td>
<td>0.04</td>
</tr>
<tr>
<td>49</td>
<td>86.39</td>
<td>3.54</td>
<td>0.04</td>
</tr>
<tr>
<td>64</td>
<td>113.69</td>
<td>4.53</td>
<td>0.04</td>
</tr>
<tr>
<td>81</td>
<td>144.83</td>
<td>5.45</td>
<td>0.04</td>
</tr>
<tr>
<td>100</td>
<td>179.39</td>
<td>6.54</td>
<td>0.04</td>
</tr>
<tr>
<td>400</td>
<td>735.54</td>
<td>20.88</td>
<td>0.03</td>
</tr>
</tbody>
</table>

DISCUSSION OF THE RESULTS

Cost of Project

As it can be seen from Table 4 in a 9 card played game, the “Plan” strategy results in an average cost of completion of 14.22, whereas the “Build” strategy has an average cost of 15.99. A paired t-test comparison shows that these two average costs are significantly different (α=0.05). This difference implies that in execution of the project, one would save in costs using a planning strategy. A very important teaching point is to show how complexity of activities would result in
the need for more planning. As it can be intuitively recognized; by increasing the number of cards played in the Oops game, the probability of an oops happening increases. In a 4 card played Oops game after placing the first card, there is a probability of 0.33 for an oops for the second card, whereas in a 9 card played game there is in average a probability of 0.66 for an oops for the second card (depending on where the first card is placed). This probability increases as the number of cards increase. This represents the fact in construction activities that as we increase the complexity of the work, the probability of things going wrong increases. In the Oops game, it can be seen that in a 4 card game (very low complexity), the Plan strategy actually results in a higher cost for the project compared to the Build strategy (Table 4). As it is shown in Figure 6, the cost difference between Plan and Build strategies increases with increase in the number of cards played in the game. In other words as the complexity increases (probability of oops increases), planning results in more savings. A linear regression analysis is performed for both strategies (Plan: \( y = 1.8455x - 3.4594, R^2 = 1 \); Build: \( y = 2.8002x - 13.417, R^2 = 0.9999 \)). The \( R^2 \) values are very close to one which represents a strong linear relationship between number of cards in the game (complexity) and total cost of the project. The regression functions can be used to predict the average total cost when other numbers of cards are in the game.
Performance Reliability

One important factor that is focused on in the Oops game is the reliability and predictability of performance. In the Cost of Project section it was discussed that in average planning strategy result in cost savings, but the Oops game just like reality is driven by some randomness. This randomness obviously results in some plays that the building strategy results in a very low cost with good luck and on the other hand with no luck it has a very high cost. Construction companies would prefer to have more predictable outcomes; therefore, a working strategy that has a wide range of outcomes is generally undesirable. With a reliable performance that has low variance in the outcome, companies will be able to provide a better estimation for bidding the projects as well as scheduling their work. A more reliable performance also means there is less need to include a high amount of contingency in estimates and will subsequently lead to a better control of the financial aspects of work.
Figures 7 to 9 show the histogram of project costs for 4, 9, and 16 cards. Plan strategy always results in a tighter range in cost of projects, regardless of the complexity. When using Plan strategy, the deviation from the average cost is considerably lower which implies that if the estimates are based on average of past works (which in reality they often are) the performance of the new project will be most likely near the estimates in the planning strategy. The larger number of cards played which means the more complicated the project is, the stronger the trend. On the other hand in the building strategy there is a wide range of possible outcomes due to leaving performance to luck.

Figure 7 – Cost Frequency for 4 Card Game

Figure 8 – Cost Frequency for 9 Card Game
Cost of Project vs. Cost of Oops and Cost of Planning

One of the aspects that can affect the outcome of the game is the cost of building, planning and oops. This is also true in reality as well; in real life we intuitively plan more when there are major consequences. On the other hand, when the consequences are minor, we tend to reduce the planning time and go to execution, reasoning that even if things go wrong they can be fixed with little effort and/or cost. Figure 10 shows the balance line between oops and planning costs for 4, 9, and 16 card games. Note that in Figure 10 in order to calculate the balance lines building cost is kept constant at 1 and only the interaction between oops cost and planning cost is investigated, since building cost is the same either with planning strategy or building. Total cost of the project is calculated based on planning cost of 1 to 6 in the planning strategy and then for each plan cost, it is calculated that under what building cost the total cost of project in building strategy will equal the planning strategy. On the balance line the total cost for plan and build strategies are equal and it represents the threshold in which the low costing strategy changes.
from planning to building or vice versa. For example, in the region to the left of the balance line for a 9 card game (i.e. when oops cost is more than 2.25*Planning cost-1.21) the lower costing strategy is planning; otherwise the building strategy results in a lower cost. The slopes of the balance lines are calculated by conducting linear regression analysis between planning cost and oops cost. $R^2$ values (1 or very close to 1) show that the oops and planning cost are linearly correlated. What is important is to note that in reality, the cost of oops is usually a lot higher than the cost of planning, since usually costs of oops include the costs of rework, idle time of labor and material, and delay. It can be seen that the slope of the lines decreases as the number of cards increases which shows that when the probability of oops increases the best strategy changes from building to planning in lower oops/plan cost ratio.

![Figure 10 – Balanced Lines for 4, 9 and 16 Card Game](image-url)
**Risk Assessment Matrix**

In order to make the decision of more planning for a project two major factors may affect the decision maker’s opinion. One is how likely are things to go wrong (probability) and the second is how big of an impact are they going to have (severity). It is very common to only look on one of these factors and overlook the other. For example when there is minor severity for an oops, it may be overlooked neglecting the fact that if it happens frequently it may have as big of an impact as a severe oops that happens rarely. Figure 11 is a risk assessment matrix showing the correlation between Probability and Severity (FM 100-14, 1998).

<table>
<thead>
<tr>
<th>Probability</th>
<th>Certain</th>
<th>M</th>
<th>H</th>
<th>E</th>
<th>E</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likely</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>E</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Possible</td>
<td>L</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Unlikely</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>Rare</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Severity</th>
<th>Low</th>
<th>Minor</th>
<th>Moderate</th>
<th>Major</th>
<th>Catastrophic</th>
</tr>
</thead>
</table>

Key: L=Low; M=Moderate; H=High; E=Extremely High

Figure 11 – Risk Assessment Matrix

The correlation is focused on in the Oops game, pursuing to teach the importance of considering both factors at the same time. Figure 12 shows the risk assessment graph developed using the Oops game simulation. Each strip represents a range for the total cost of the project. As it can be
seen for a constant cost of oops (severity), the higher the number of cards (probability), the higher total cost (i.e. multiple cost regions are crossed when moving parallel to y-axis). This shows that when the complexity level of a project increases, the importance of a reliable plan increases. On the other hand if the number of cards is constant, the higher cost for oops the higher total cost (i.e. multiple cost regions are crossed when moving parallel to x-axis). This shows that when the consequence of oops increases, the importance of a reliable plan increases. Combining the aforementioned observations the effect of two dimensional coupling of severity and probability is derived as shown in Figure 12. It can be seen that frequent but minor and low costing oopses can increase the total cost the same as costly and rare oops. But perception of these two situations is not always right. One may consider being in the region 1 (having frequent but minor oopses) is safer than being in region 2 (having serve but rare oopses). Thus project managers may not necessarily pay enough attention or have good strategies ready to use to reduce the causes of this type of oopses. But in fact total cost is higher in region 1. Because people’s perception on the frequency and severity can be very different from the reality, we need to pay attention to which zone we are in right now and be prepared to use the best strategies to move to a lower cost zone. In Order to reduce the total cost of the project one should:

- Reduce the probability of oops happening (Line 1) by planning to identify potential problems and causes of mishaps which may occur during the work. In order to reduce the probability of oops, we need to ensure constraints on successful task execution are identified and removed;
• Reduction of cost of oops (Line 2) by planning to maintain a workable backlog in cases of oops to utilize workforce and equipment in a productive way instead of keeping them idle waiting for the cause of problems to be fixed. One important requirement for the workable backlog is that completing tasks in the workable backlog should not cause disruption or have negative effect on other tasks in the CPM network. Otherwise, moving workers to work on non-planned tasks may cause more damage than benefits.

• Combination of 1 and 2 (Line 3) which as it can be seen result in higher reduction of total cost.

A thorough understanding of this concept is important: knowing the important factors of risk and not overlooking one while focusing on the other helps evaluating present conditions as well as conceptualizing a path for improvement.

Figure 12 – Risk Assessment for Oops Game
Validity Check

A case study was employed to study the use of reliable weekly planning to improve project performance in mechanical related construction tasks. A mechanical contractor that specializes in plumbing, heating, ventilation, and air conditioning participated in the case study. The mechanical contractor was chosen for the study as mechanical related tasks are very common within the construction industry. The company compared the performance of two separate, but similar projects for the case study (Table 5). The work involved in the study was self-performed by the mechanical contractor using labor crews with similar experience and skill levels; however, there could still have been some differences due to learning curve issues. The projects also had the same project manager and superintendent for the first eight weeks of the study. The intent was to keep constant as many aspects as possible, so the differences in performance could be attributed to the differences in the planning processes of the projects. The study was based on data collected over a 16 week period.
The company tracked productivity for the primary activities associated with the project. These activities are shown on the horizontal axis in Figure 13. The first four activities, since they are abbreviated in Figure 13, are spelled out for clarification. They are piping and fitting for underground cast iron, above ground cast iron, steel, and copper. There were three activities (fixtures, commissioning, and shop) that had not been performed at the conclusion of the study; therefore, they do not have a bar associated with them in Figure 13. The vertical axis of Figure 13 is the
performance index the company uses to measure its productivity. It is the ratio of earned to spent man-days for each activity. Ideally, this index is greater than or equal to 1.0. If the ratio is greater than 1.0, this indicates that more man-days were earned (i.e. the company was paid for) than were spent, on a particular activity. In all but one activity (installing equipment), the LPS® project outperformed the non-LPS® project. In terms of overall performance, the LPS® project had about a 35% higher productivity performance index (1.03) than the non-LPS® project (0.76).

Given the low performance index of the non-LPS® project, it could be argued that it was not representative of the company’s capability. The company involved with the case study has a solid reputation and history as a company that performs very well. Based on discussions with the president of the company and the project manager, they felt even though the non-LPS® project’s performance index was lower than desired, it was still appropriate to compare the performance of the two projects because of the similarities in the scopes of work.

The company estimated they spent about an additional $4000 in planning costs over the 16 weeks for the LPS® project. These costs were due to having an extra project manager coordinate and run the weekly LPS® meetings. Access to the specific total planning costs for each projects was not provided; therefore, just the estimated additional costs due to implementing the LPS® were included. The estimated savings for the LPS® project were approximately $52,000, thus resulting in a benefit / cost ratio of 13:1.
A simple and easy to play game was presented to illustrate the impact of reliable planning on performance of construction projects. Two strategies were investigated: (1) Attempting to perform scheduled activities with no additional planning, while accepting the fact that there is a probability of non-completion; (2) Spending more cost and time in order to have a reliable and detailed pull plan to prevent causes of non-completion before execution of activities. The game allows the players to develop a better understanding of several concepts:

- Reliable planning results in reduction of total cost of project and cost saving;
- While it is possible to rarely have a low cost without planning (by being lucky) but due to a large range of possible outcomes the performance reliability is low. On the other hand reliable planning tightens the probable range of total cost and increases performance
reliability which apart from cost saving, is important in bidding competitiveness since there is no need to include high contingency cost in bid estimates;

- Complexity of activities affects the effectiveness of reliable planning. As the complexity increases and there is higher probability of mishaps, the cost benefits from planning increases;
- Severity of mishaps is an important factor in effectiveness of reliable planning. As the cost of each oops increases the cost benefits from planning increases;
- Understanding two dimensional coupling of factors affecting the total cost (probability and severity) helps to properly evaluate the existing conditions of projects and to set up a path for improvement.

The computer simulations of the game for different scenarios (4, 9, 16, 25, 36, 49, 64, 81, 100, 400 card play games) enabled detailed and quantified analysis of the independent and combined effects of reliable planning. It was shown that cost benefits can range from -0.06 to 0.34 percent depending on the complexity of the game. Furthermore, computer simulations were used to calculate balance lines in which the lower costing strategy changes from planning to building based on oops/plan cost ratio. Based on the computer simulation results and a risk assessment tool it was recommended that for optimum results of planning it is important to (1) Identify the problems that can cause non-completions of task and prevent them (reduce probability of oops) (2) Maintain a list of workable backlogs to reduce the cost of mishaps by utilizing labor force and equipment (reduce severity of oops).
To validate the findings of the game, a case study was performed including two projects executed by a mechanical subcontractor using the two aforementioned strategies. It was shown that reliable planning resulted in a benefit-cost ratio of 13:1 and 35% higher productivity.

Oops game can be used as an effective educational tool to study and illustrate the effects of reliable planning. The results of the game and case study helps managers interested in improving project performance by showing the impact of reliable planning and important factors in its effectiveness.
REFERENCES


APPENDICES
Appendix A. Stroboscope Simulation Code – 3 Work Station Push

******************************************************************************
****
/* General section for problem parameters
VARIABLE NrIterations 1000;
SAVEVALUE CurrentIteration* 1;
******************************************************************************
****
/* Definition of resource types
GENTYPE        gen; /GEN
GENTYPE        labor; /LA
******************************************************************************
****
/* Definition of network nodes
COMBI          Act1;
QUEUE          Input gen;
QUEUE          WaitPile1 gen;
COMBI          Act2;
QUEUE          WaitPile2 gen;
COMBI          Act3;
QUEUE          Completed gen;
QUEUE          LEM1 labor;
QUEUE          LEM2 labor;
QUEUE          LEM3 labor;
******************************************************************************
****
/* Definition of network Links
LINK           L1 Input Act1;
LINK           L2 Act1 WaitPile1;
LINK           L3 WaitPile1 Act2;
LINK           L4 Act2 WaitPile2;
LINK           L5 WaitPile2 Act3;
LINK           L6 Act3 Completed;
LINK           LL1 LEM1 Act1;
LINK           LL2 Act1 LEM1;
LINK           LL3 LEM2 Act2;
**Definition of global variables and programing objects**

COLLECTOR cWaitPile1*;
COLLECTOR cWaitPile2*;
COLLECTOR cCompleted*;
COLLECTOR cAct1*;
COLLECTOR cAct2*;
COLLECTOR cAct3*;
COLLECTOR cSimTime*;
COLLECTOR cLEM1*;
COLLECTOR cLEM2*;
COLLECTOR cLEM3*;

***Startup of Act1***

DRAWMAT L1 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
DRAWMAT LL1 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
DURATION Act1 'Normal[7,2.8]';

***Termination of Act1***

RELEASEAMT L2 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
RELEASEAMT LL2 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here

***Entry of resources into Input***

******

****
/* Entry of resources into WaitPile1
*******************************************************************************/

/**************/

/* Startup of Act2
DRAWMAT L3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWMAT LL3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act2 'Normal[5.6,2.24]';
*******************************************************************************/

/**************/

/* Termination of Act2
RELEASEAMT L4 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL4 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
*******************************************************************************/

**************

/* Entry of resources into WaitPile2
*******************************************************************************/

**************

/* Startup of Act3
DRAWMAT L5 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWMAT LL5 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act3 'Normal[8.4,3.36]';
*******************************************************************************/

**************

/* Termination of Act3
RELEASEAMT L6 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL6 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
*******************************************************************************/

**************
/* Entry of resources into Completed */

*************

/* Entry of resources into LEM1 */

*************

/* Entry of resources into LEM2 */

*************

/* Entry of resources into LEM3 */

*************

/* Initialization of Queues, Running the Simulation, Presenting Results */

*************

/* Iteration */

*************

WHILE CurrentIteration<=NrIterations;

CLEAR;
INIT Input 100;
INIT WaitPile1 0;
INIT WaitPile2 0;
INIT Completed 0;
INIT LEM1 1;
INIT LEM2 1;
INIT LEM3 1;
SIMULATEUNTIL Completed.CurCount>=100;
COLLECT cWaitPile1 WaitPile1.AveCount;
COLLECT cWaitPile2 WaitPile2.AveCount;
COLLECT cCompleted Completed.CurCount;
COLLECT cAct1 Act1.TotInst;
COLLECT cAct2 Act2.TotInst;
COLLECT cAct3 Act3.TotInst;
COLLECT cLEM1 LEM1.AveCount;
COLLECT cLEM2 LEM2.AveCount;
COLLECT cLEM3 LEM3.AveCount;
COLLECT cSimTime SimTime;
PRINT StdOutput "Iteration %.0f Duration %.4f \n"
  CurrentIteration SimTime;
ASSIGN CurrentIteration CurrentIteration+1;
WEND;
/***************
/* Reporting
/***************
PRINT StdOutput "cWaitPile1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile1.AveVal
cWaitPile1.SDVal
cWaitPile1.MinVal
cWaitPile1.MaxVal;
PRINT StdOutput "cWaitPile2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile2.AveVal
cWaitPile2.SDVal
cWaitPile2.MinVal
cWaitPile2.MaxVal;
PRINT StdOutput "cCompleted Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cCompleted.AveVal
cCompleted.SDVal
cCompleted.MinVal
cCompleted.MaxVal;
PRINT StdOutput "cAct1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct1.AveVal
cAct1.SDVal
cAct1.MinVal
cAct1.MaxVal;
PRINT StdOutput "cAct2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct2.AveVal
cAct2.SDVal
cAct2.MinVal
cAct2.MaxVal;
PRINT StdOutput "cAct3 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct3.AveVal
cAct3.SDVal
cAct3.MinVal
cAct3.MaxVal;
PRINT StdOutput "cLEM1 Ave |t %8.4f |t SD |t %8.4f |t Min |t %8.4f |t Max |t %8.4f \n"
cLEM1.AveVal
cLEM1.SDVal
cLEM1.MinVal
cLEM1.MaxVal;
PRINT StdOutput "cLEM2 Ave |t %8.4f |t SD |t %8.4f |t Min |t %8.4f |t Max |t %8.4f \n"
cLEM2.AveVal
cLEM2.SDVal
cLEM2.MinVal
cLEM2.MaxVal;
PRINT StdOutput "cLEM3 Ave |t %8.4f |t SD |t %8.4f |t Min |t %8.4f |t Max |t %8.4f \n"
cLEM3.AveVal
cLEM3.SDVal
cLEM3.MinVal
cLEM3.MaxVal;
PRINT StdOutput "cSimTime Ave |t %8.4f |t SD |t %8.4f |t Min |t %8.4f |t Max |t %8.4f \n"
cSimTime.AveVal
cSimTime.SDVal
cSimTime.MinVal
cSimTime.MaxVal;

Appendix B. Stroboscope Simulation Code – 3 Work Station Pull

/*******************************************************************************
**************
/* General section for problem parameters
VARIABLE NrIterations 1000;
SAVEVALUE CurrentIteration* 1;
/*******************************************************************************
**************
/* Definition of resource types
GENTYPE gen; /GEN
GENTYPE labor; /LA
GENTYPE ticket; /TI
/*******************************************************************************
**************
/* Definition of network nodes
COMBI Act1;
QUEUE Input gen;
QUEUE WaitPile1 gen;
QUEUE WaitPile2 gen;
COMBI Act2;
COMBI Act3;
QUEUE Completed gen;
QUEUE LEM1 labor;
QUEUE LEM2 labor;
QUEUE LEM3 labor;
QUEUE Output1 gen;
COMBI Batching1;
QUEUE Kanban1 ticket;
QUEUE Output2 gen;
COMBI Batching2;
QUEUE Kanban2 ticket;
/*******************************************************************************
**************
/* Definition of network Links
LINK L1 Input Act1;
LINK L2 Act1 Output1;
LINK LL1 LEM1 Act1;
LINK LL2 Act1 LEM1;
LINK LL3 LEM2 Act2;
LINK LL4 Act2 LEM2;
LINK LL5 LEM3 Act3;
LINK LL6 Act3 LEM3;
LINK L3 Output1 Batching1;
LINK L4 Batching1 WaitPile1;
LINK L5 WaitPile1 Act2;
LINK L6 Act2 Output2;
LINK L7 Output2 Batching2;
LINK L8 Batching2 WaitPile2;
LINK L9 WaitPile2 Act3;
LINK LK1 Kanban1 Act1;

LINK LK3 Kanban2 Act2;
LINK L10 Act3 Completed;

/***************************************************************************/
*******
/* Definition of global variables and programing objects */
COLLECTOR cWaitPile1*;
COLLECTOR cWaitPile2*;
COLLECTOR cCompleted*;
COLLECTOR cAct1*;
COLLECTOR cAct2*;
COLLECTOR cAct3*;
COLLECTOR cSimTime*;
COLLECTOR cLEM1*;
COLLECTOR cLEM2*;
COLLECTOR cLEM3*;
COLLECTOR cKanban1*;
COLLECTOR cKanban2*;

/***************************************************************************/
*******
/* Startup of Act1 */
SEMAPHORE Act1 'WaitPile1.CurCount==0';
DRAWAMT L1 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
DRAWAMT    LL1 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT    LK1 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION    Act1 'Normal[7,2.8]';
/*Termination of Act1*/
RELEASEAMT  L2 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT  LL2 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here

/* Entry of resources into Input*/
/* Entry of resources into WaitPile1*/
/* Entry of resources into WaitPile2*/
/* Startup of Act2*/
SEMAPHORE    Act2 'WaitPile2.CurCount==0';
DRAWAMT      LL3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT      L5 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT      LK3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION    Act2 'Normal[5.6,2.24]';
/* Termination of Act2*/
RELEASEAMT  LL4 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT   L6 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
*************************************************************************/
**************
/* Startup of Act3
DRAWAMT       L5 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT       L9 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION      Act3 'Normal[8.4,3.36]';
*************************************************************************/
**************
/* Termination of Act3
RELEASEAMT    LL6 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT    L10 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
*************************************************************************/
**************
/* Entry of resources into Completed
*************************************************************************/
**************
/* Entry of resources into LEM1
*************************************************************************/
**************
/* Entry of resources into LEM2
*************************************************************************/
**************
/* Entry of resources into LEM3
*************************************************************************/
**************
/* Entry of resources into Output1
/************************************************************************************
**************
/* Startup of Batching1
DRAWMHT L3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Batching1 '0';
/*******************************************************************************
**************
/* Termination of Batching1
RELEASEAMT L4 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
******************************************************************************

**************
/* Entry of resources into Kanban1
******************************************************************************

**************
/* Entry of resources into Output2
******************************************************************************

**************

/* Startup of Batching2
DRAWMHT L7 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Batching2 '0';
/*******************************************************************************
**************
/* Termination of Batching2
RELEASEAMT L8 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
******************************************************************************

**************
/* Entry of resources into Kanban2
******************************************************************************

**************
/* Initialization of Queues, Running the Simulation, Presenting Results
/***************
/* Iteration
/***************
WHILE CurrentIteration<=NrIterations;
CLEAR;
INIT Input 100;
INIT WaitPile1 0;
INIT WaitPile2 0;
INIT Output1 0;
INIT Output2 0;
INIT Completed 0;
INIT LEM1 1;
INIT LEM2 1;
INIT LEM3 1;
INIT Kanban1 100;
INIT Kanban2 100;
SIMULATEUNTIL Completed.CurCount>=100;
COLLECT cWaitPile1 WaitPile1.AveCount;
COLLECT cWaitPile2 WaitPile2.AveCount;
COLLECT cCompleted Completed.CurCount;
COLLECT cAct1 Act1.TotInst;
COLLECT cAct2 Act2.TotInst;
COLLECT cAct3 Act3.TotInst;
COLLECT cLEM1 LEM1.AveCount;
COLLECT cLEM2 LEM2.AveCount;
COLLECT cLEM3 LEM3.AveCount;
COLLECT cSimTime SimTime;
COLLECT cKanban1 Kanban1.AveCount;
COLLECT cKanban2 Kanban2.AveCount;
PRINT StdOutput "Iteration %.0f Duration %.4f \n"
  CurrentIteration SimTime;
ASSIGN CurrentIteration CurrentIteration+1;
WEND;
/***************
/* Reporting 
***************
PRINT StdOutput "cWaitPile1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile1.AveVal
cWaitPile1.SDVal
cWaitPile1.MinVal
cWaitPile1.MaxVal;
PRINT StdOutput "cWaitPile2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile2.AveVal
cWaitPile2.SDVal
cWaitPile2.MinVal
cWaitPile2.MaxVal;
PRINT StdOutput "cCompleted Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cCompleted.AveVal
cCompleted.SDVal
cCompleted.MinVal
cCompleted.MaxVal;
PRINT StdOutput "cAct1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct1.AveVal
cAct1.SDVal
cAct1.MinVal
cAct1.MaxVal;
PRINT StdOutput "cAct2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct2.AveVal
cAct2.SDVal
cAct2.MinVal
cAct2.MaxVal;
PRINT StdOutput "cAct3 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct3.AveVal
cAct3.SDVal
cAct3.MinVal
cAct3.MaxVal;
PRINT StdOutput "cLEM1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cLEM1.AveVal
cLEM1.SDVal
cLEM1.MinVal
cLEM1.MaxVal;
PRINT StdOutput "cLEM2 Ave %8.4f SD %8.4f Min %8.4f Max %8.4f \n"
cLEM2.AveVal
 cLEM2.SDVal
 cLEM2.MinVal
 cLEM2.MaxVal;
PRINT StdOutput "cLEM3 Ave %8.4f SD %8.4f Min %8.4f Max %8.4f \n"
cLEM3.AveVal
 cLEM3.SDVal
 cLEM3.MinVal
 cLEM3.MaxVal;
PRINT StdOutput "cKanban1 Ave %8.4f SD %8.4f Min %8.4f Max %8.4f \n"
cKanban1.AveVal
 cKanban1.SDVal
 cKanban1.MinVal
 cKanban1.MaxVal;
PRINT StdOutput "cKanban2 Ave %8.4f SD %8.4f Min %8.4f Max %8.4f \n"
cKanban2.AveVal
 cKanban2.SDVal
 cKanban2.MinVal
 cKanban2.MaxVal;
PRINT StdOutput "cSimTime Ave %8.4f SD %8.4f Min %8.4f Max %8.4f \n"
cSimTime.AveVal
 cSimTime.SDVal
 cSimTime.MinVal
 cSimTime.MaxVal;
Appendix C. Stroboscope Simulation Code – 5 Work Station Push

/************************************************************************************
**************
/* General section for problem parameters
VARIABLE NrIterations 1000;
SAVEVALUE CurrentIteration* 1;
/************************************************************************************
**************
/* Definition of resource types
GENTYPE gen; /GEN
GENTYPE labor; /LA
/************************************************************************************
**************
/* Definition of network nodes
COMBI Act1;
QUEUE Input gen;
QUEUE WaitPile1 gen;
COMBI Act2;
QUEUE WaitPile2 gen;
COMBI Act3;
QUEUE WaitPile3 gen;
COMBI Act4;
QUEUE WaitPile4 gen;
COMBI Act5;
QUEUE Completed gen;
QUEUE LEM1 labor;
QUEUE LEM2 labor;
QUEUE LEM3 labor;
QUEUE LEM4 labor;
QUEUE LEM5 labor;
/************************************************************************************
**************
/* Definition of network links
LINK L1 Input Act1;
LINK L2 Act1 WaitPile1;
LINK L3 WaitPile1 Act2;
LINK L4 Act2 WaitPile2;
LINK L5 WaitPile2 Act3;
LINK L6 Act3 WaitPile3;
LINK L7 WaitPile3 Act4;
LINK L8 Act4 WaitPile4;
LINK L9 WaitPile4 Act5;
LINK L10 Act5 Completed;
LINK LL1 LEM1 Act1;
LINK LL2 Act1 LEM1;
LINK LL3 LEM2 Act2;
LINK LL4 Act2 LEM2;
LINK LL5 LEM3 Act3;
LINK LL6 Act3 LEM3;
LINK LL7 LEM4 Act4;
LINK LL8 Act4 LEM4;
LINK LL9 LEM5 Act5;
LINK LL10 Act5 LEM5;

******************************************************************************
*************
/* Definition of global variables and program objects */
COLLECTOR cWaitPile1*;
COLLECTOR cWaitPile2*;
COLLECTOR cWaitPile3*;
COLLECTOR cWaitPile4*;
COLLECTOR cCompleted*;
COLLECTOR cAct1*;
COLLECTOR cAct2*;
COLLECTOR cAct3*;
COLLECTOR cAct4*;
COLLECTOR cAct5*;
COLLECTOR cSimTime*;
COLLECTOR cLEM1*;
COLLECTOR cLEM2*;
COLLECTOR cLEM3*;
COLLECTOR cLEM4*;
COLLECTOR cLEM5*;
/* Startup of Act1 */
DRAWAMT L1 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT LL1 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act1 'Normal[4.2,1.68]';
/* Termination of Act1 */
RELEASEAMT L2 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL2 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
/* Entry of resources into Input */
/* Entry of resources into WaitPile1 */
/* Startup of Act2 */
DRAWAMT L3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT LL3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act2 'Normal[8.4,3.36]';
/* Termination of Act2 */
RELEASEAMT L4 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL4 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
/* Entry of resources into WaitPile2
']!=' Entry of resources into WaitPile2
************************************************************************************
**************
/* Startup of Act3
DRAWMAT L5 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWMAT LL5 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURANTION Act3 'Normal[9.8,3.92]';
************************************************************************************
**************
/* Termination of Act3
RELEASEAMT L6 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL6 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
************************************************************************************
**************
/* Entry of resources into WaitPile3
************************************************************************************
**************
/* Startup of Act4
DRAWMAT L7 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWMAT LL7 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURANTION Act4 'Normal[5.6,2.24]';
************************************************************************************
**************
/* Termination of Act4
RELEASEAMT L8 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL8 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
************************************************************************************
/* Entry of resources into WaitPile4
********************************************************************************
******
/* Startup of Act5
DRAWAMT L9 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
DRAWAMT LL9 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
DURATION Act5 'Normal[7,2.8]';
********************************************************************************
******
/* Termination of Act5
RELEASEAMT L10 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
RELEASEAMT LL10 '1';
//ONFLOW , ONDRAW , ONRELEASE Code Here
********************************************************************************
******
/* Entry of resources into Completed
********************************************************************************
******
/* Entry of resources into LEM1
********************************************************************************
******
/* Entry of resources into LEM2
********************************************************************************
******
/* Entry of resources into LEM3
********************************************************************************
******
/* Entry of resources into LEM4
********************************************************************************
******
/* Entry of resources into LEM5
********************************************************************************
******
/ * Initialization of Queues, Running the Simulation, Presenting Results
/ ***************
/ * Iteration
/ ***************
WHILE CurrentIteration<=NrIterations;
    CLEAR;
    INIT Input 100;
    INIT WaitPile1 0;
    INIT WaitPile2 0;
    INIT WaitPile3 0;
    INIT WaitPile4 0;
    INIT Completed 0;
    INIT LEM1 1;
    INIT LEM2 1;
    INIT LEM3 1;
    INIT LEM4 1;
    INIT LEM5 1;
    SIMULATEUNTIL Completed.CurCount>=100;
    COLLECT cWaitPile1 WaitPile1.AveCount;
    COLLECT cWaitPile2 WaitPile2.AveCount;
    COLLECT cWaitPile3 WaitPile3.AveCount;
    COLLECT cWaitPile4 WaitPile4.AveCount;
    COLLECT cCompleted Completed.CurCount;
    COLLECT cAct1 Act1.TotInst;
    COLLECT cAct2 Act2.TotInst;
    COLLECT cAct3 Act3.TotInst;
    COLLECT cAct4 Act4.TotInst;
    COLLECT cAct5 Act5.TotInst;
    COLLECT cLEM1 LEM1.AveCount;
    COLLECT cLEM2 LEM2.AveCount;
    COLLECT cLEM3 LEM3.AveCount;
    COLLECT cLEM4 LEM4.AveCount;
    COLLECT cLEM5 LEM5.AveCount;
    COLLECT cSimTime SimTime;
    PRINT StdOutput "Iteration %.0f Duration %.4f \n"
        CurrentIteration SimTime;
ASSIGN CurrentIteration CurrentIteration+1;
WEND;
/***************
/* Reporting
/***************
PRINT StdOutput "cWaitPile1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile1.AveVal
cWaitPile1.SDVal
cWaitPile1.MinVal
cWaitPile1.MaxVal;
PRINT StdOutput "cWaitPile2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile2.AveVal
cWaitPile2.SDVal
cWaitPile2.MinVal
cWaitPile2.MaxVal;
PRINT StdOutput "cWaitPile3 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile3.AveVal
cWaitPile3.SDVal
cWaitPile3.MinVal
cWaitPile3.MaxVal;
PRINT StdOutput "cWaitPile4 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile4.AveVal
cWaitPile4.SDVal
cWaitPile4.MinVal
cWaitPile4.MaxVal;
PRINT StdOutput "cCompleted Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cCompleted.AveVal
cCompleted.SDVal
cCompleted.MinVal
cCompleted.MaxVal;
PRINT StdOutput "cAct1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct1.AveVal
cAct1.SDVal
cAct1.MinVal
cAct1.MaxVal;
PRINT StdOutput "cAct2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct2.AveVal
cAct2.SDVal
<table>
<thead>
<tr>
<th></th>
<th>Ave</th>
<th>SD</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Act2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Act3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Act4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Act5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEM1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEM2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEM3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEM4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
PRINT StdOutput "cLEM5 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cLEM5.AveVal
cLEM5.SDVal
cLEM5.MinVal
cLEM5.MaxVal;
PRINT StdOutput "cSimTime Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cSimTime.AveVal
cSimTime.SDVal
cSimTime.MinVal
cSimTime.MaxVal;
Appendix D. Stroboscope Simulation Code – 5 Work Station Pull

/************************************************************************************
**************
/* General section for problem parameters
VARIABLE NrIterations 1000;
SAVEVALUE CurrentIteration* 1;
/************************************************************************************
**************
/* Definition of resource types
GENTYPE gen; /GEN
GENTYPE labor; /LA
/************************************************************************************
************
/* Definition of network nodes
COMBI Act1;
QUEUE Input gen;
QUEUE WaitPile1 gen;
COMBI Act2;
QUEUE WaitPile2 gen;
COMBI Act3;
QUEUE WaitPile3 gen;
COMBI Act4;
QUEUE WaitPile4 gen;
COMBI Act5;
QUEUE Completed gen;
QUEUE LEM1 labor;
QUEUE LEM2 labor;
QUEUE LEM3 labor;
QUEUE LEM4 labor;
QUEUE LEM5 labor;
/************************************************************************************
************
/* Definition of network Links
LINK L1 Input Act1;
LINK L2 Act1 WaitPile1;
LINK L3 WaitPile1 Act2;
LINK L4 Act2 WaitPile2;
LINK L5 WaitPile2 Act3;
LINK L6 Act3 WaitPile3;
LINK L7 WaitPile3 Act4;
LINK L8 Act4 WaitPile4;
LINK L9 WaitPile4 Act5;
LINK L10 Act5 Completed;
LINK LL1 LEM1 Act1;
LINK LL2 Act1 LEM1;
LINK LL3 LEM2 Act2;
LINK LL4 Act2 LEM2;
LINK LL5 LEM3 Act3;
LINK LL6 Act3 LEM3;
LINK LL7 LEM4 Act4;
LINK LL8 Act4 LEM4;
LINK LL9 LEM5 Act5;
LINK LL10 Act5 LEM5;

/************************************************************************************
*************
/* Definition of global variables and programming objects
COLLECTOR cWaitPile1*;
COLLECTOR cWaitPile2*;
COLLECTOR cWaitPile3*;
COLLECTOR cWaitPile4*;
COLLECTOR cCompleted*;
COLLECTOR cAct1*;
COLLECTOR cAct2*;
COLLECTOR cAct3*;
COLLECTOR cAct4*;
COLLECTOR cAct5*;
COLLECTOR cSimTime*;
COLLECTOR cLEM1*;
COLLECTOR cLEM2*;
COLLECTOR cLEM3*;
COLLECTOR cLEM4*;
COLLECTOR cLEM5*;
/************************************************************************************
**************
/* Startup of Act1
SEMAPHORE Act1 'WaitPile1.CurCount==0';
DRAWMNT L1 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWMNT LL1 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act1 'Normal[4.2,1.68]';
/*****************************************************************************/
/****************************
**************
/* Termination of Act1
RELEASEAMT L2 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL2 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
/************************************************************************************
/****************************
**************
/* Entry of resources into Input
/****************************
/****************************
**************
/* Entry of resources into WaitPile1
/****************************
/****************************
/* Startup of Act2
SEMAPHORE Act2 'WaitPile2.CurCount==0';
DRAWMNT L3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWMNT LL3 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act2 'Normal[8.4,3.36]';
/****************************
/* Termination of Act2
RELEASEAMT L4 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL4 '1';
/* Entry of resources into WaitPile2 */

//ONFLOW, ONDRAW, ONRELEASE Code Here

******************************************************************************

**************

/* Startup of Act3 */

SEMAPHORE Act3 'WaitPile3.CurCount==0';
DRAWAMT L5 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT LL5 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act3 'Normal[9.8,3.92]';

******************************************************************************

**************

/* Termination of Act3 */

RELEASEAMT L6 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL6 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here

******************************************************************************

**************

/* Entry of resources into WaitPile3 */

******************************************************************************

**************

/* Startup of Act4 */

SEMAPHORE Act4 'WaitPile4.CurCount==0';
DRAWAMT L7 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWAMT LL7 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act4 'Normal[5.6,2.24]';

******************************************************************************

**************

/* Termination of Act4 */

RELEASEAMT L8 '1';

//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL8 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
iscriminalism************************************************************************************
**************
/* Entry of resources into WaitPile4
iscriminalism************************************************************************************
**************
/* Startup of Act5
DRAWMAT L9 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DRAWMAT LL9 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
DURATION Act5 'Normal[7,2.8]';
iscriminalism************************************************************************************
**************
/* Termination of Act5
RELEASEAMT L10 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
RELEASEAMT LL10 '1';
//ONFLOW, ONDRAW, ONRELEASE Code Here
iscriminalism************************************************************************************
**************
/* Entry of resources into Completed
iscriminalism************************************************************************************
**************
/* Entry of resources into LEM1
iscriminalism************************************************************************************
**************
/* Entry of resources into LEM2
iscriminalism************************************************************************************
**************
/* Entry of resources into LEM3
iscriminalism************************************************************************************
**************
/* Entry of resources into LEM4
iscriminalism************************************************************************************
**************
/* Entry of resources into LEM5
/************************************************************************************
**************
/* Initialization of Queues, Running the Simulation, Presenting Results
/********************
/*/ Iteration
/**************
WHILE CurrentIteration<=NrIterations;
CLEAR;
INIT Input 100;
INIT WaitPile1 0;
INIT WaitPile2 0;
INIT WaitPile3 0;
INIT WaitPile4 0;
INIT Completed 0;
INIT LEM1 1;
INIT LEM2 1;
INIT LEM3 1;
INIT LEM4 1;
INIT LEM5 1;
SIMULATEUNTIL Completed.CurCount>=100;
COLLECT cWaitPile1 WaitPile1.AveCount;
COLLECT cWaitPile2 WaitPile2.AveCount;
COLLECT cWaitPile3 WaitPile3.AveCount;
COLLECT cWaitPile4 WaitPile4.AveCount;
COLLECT cCompleted Completed.CurCount;
COLLECT cAct1 Act1.TotInst;
COLLECT cAct2 Act2.TotInst;
COLLECT cAct3 Act3.TotInst;
COLLECT cAct4 Act4.TotInst;
COLLECT cAct5 Act5.TotInst;
COLLECT cLEM1 LEM1.AveCount;
COLLECT cLEM2 LEM2.AveCount;
COLLECT cLEM3 LEM3.AveCount;
COLLECT cLEM4 LEM4.AveCount;
COLLECT cLEM5 LEM5.AveCount;
COLLECT cSimTime SimTime;
PRINT StdOutput "Iteration %.0f Duration %.4f \n"
CurrentIteration SimTime;
ASSIGN CurrentIteration CurrentIteration+1;
WEND;
/***************
/* Reporting
/***************
PRINT StdOutput "cWaitPile1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile1.AveVal
cWaitPile1.SDVal
cWaitPile1.MinVal
cWaitPile1.MaxVal;
PRINT StdOutput "cWaitPile2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile2.AveVal
cWaitPile2.SDVal
cWaitPile2.MinVal
cWaitPile2.MaxVal;
PRINT StdOutput "cWaitPile3 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile3.AveVal
cWaitPile3.SDVal
cWaitPile3.MinVal
cWaitPile3.MaxVal;
PRINT StdOutput "cWaitPile4 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cWaitPile4.AveVal
cWaitPile4.SDVal
cWaitPile4.MinVal
cWaitPile4.MaxVal;
PRINT StdOutput "cCompleted Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cCompleted.AveVal
cCompleted.SDVal
cCompleted.MinVal
cCompleted.MaxVal;
PRINT StdOutput "cAct1 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct1.AveVal
cAct1.SDVal
cAct1.MinVal
cAct1.MaxVal;
PRINT StdOutput "cAct2 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cAct2.AveVal
Act2.SDVal
cAct2.MinVal
cAct2.MaxVal;
PRINT StdOutput "%Act3 Ave %8.4f %t SD %8.4f %t Min %8.4f %t Max %8.4f \n"
cAct3.AveVal
cAct3.SDVal
cAct3.MinVal
cAct3.MaxVal;
PRINT StdOutput "%Act4 Ave %8.4f %t SD %8.4f %t Min %8.4f %t Max %8.4f \n"
cAct4.AveVal
cAct4.SDVal
cAct4.MinVal
cAct4.MaxVal;
PRINT StdOutput "%Act5 Ave %8.4f %t SD %8.4f %t Min %8.4f %t Max %8.4f \n"
cAct5.AveVal
cAct5.SDVal
cAct5.MinVal
cAct5.MaxVal;
PRINT StdOutput "%LEM1 Ave %8.4f %t SD %8.4f %t Min %8.4f %t Max %8.4f \n"
LEM1.AveVal
LEM1.SDVal
LEM1.MinVal
LEM1.MaxVal;
PRINT StdOutput "%LEM2 Ave %8.4f %t SD %8.4f %t Min %8.4f %t Max %8.4f \n"
LEM2.AveVal
LEM2.SDVal
LEM2.MinVal
LEM2.MaxVal;
PRINT StdOutput "%LEM3 Ave %8.4f %t SD %8.4f %t Min %8.4f %t Max %8.4f \n"
LEM3.AveVal
LEM3.SDVal
LEM3.MinVal
LEM3.MaxVal;
PRINT StdOutput "%LEM4 Ave %8.4f %t SD %8.4f %t Min %8.4f %t Max %8.4f \n"
LEM4.AveVal
LEM4.SDVal
LEM4.MinVal
cLEM4.MaxVal;
PRINT StdOutput "cLEM5 Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cLEM5.AveVal
cLEM5.SDVal
cLEM5.MinVal
cLEM5.MaxVal;
PRINT StdOutput "cSimTime Ave \t %8.4f \t SD \t %8.4f \t Min \t %8.4f \t Max \t %8.4f \n"
cSimTime.AveVal
cSimTime.SDVal
cSimTime.MinVal
cSimTime.MaxVal;
Appendix E. MATLAB Code – Oops Game “Plan” Strategy

clc;
clear all;
% this program is for oops game when played under planning strategy
%n=number of cards (4 to 400)
%yard = array of drawn cards (size 1*n)
%card=Card at hand at each step of the game (totaly n steps)
%count=current step of the game ( 1 to n)
%project=matrix of project (n^o.5*n^0.5 : 2*2,3*3,4*4)
%planningy=array of planning yard
%p= # of plannings
%b= # of builds
pnum=zeros(1000,1);
for iter=1:1000
n = 81;
p=0;
b=0;
planningy = zeros(1,n);
yard = randperm(n);
project=zeros(n^0.5);
count = 1;
%countp counts for planningy
countp=0;
% checks if all the remaining cards have the
%possibility to be built with no oops happening (like if we have gotten
%2,5 and 8 already) then all the remaining cards are just built. no
%planning any more
while count<=n;
    card=yard(count);
    i=floor((card+(n^0.5-1))/n^0.5);
    j=mod(card,n^0.5);
    if (j==0);
        j=n^0.5;
    end
    l=0;
    if (count==1);
        project(i,j)=1;
        b=b+1;
    else
        if (i-1>0)
            if (project(i-1,j)==1)
                project(i,j)=1;
                l=1;
            end
        end
        if (i+1<n^0.5+1)
            if (project(i+1,j)==1)
                project(i,j)=1;
                l=1;
            end
        end
    end
    if (l==0)
        project(i,j)=1;
        b=b+1;
        countp=0;
    end
    count=count+1;
end
p=countp;
b=

98
end
end
if (j-1>0)
    if (project(i,j-1)==1)
        project(i,j)=1;
        l=1;
    end
end
if (j+1<n^0.5+1)
    if (project(i,j+1)==1)
        project(i,j)=1;
        l=1;
    end
end
p=p+1;
if (l==0)
    planningy(countp+1)=card;
    countp=countp+1;
end
end
for kk=1:n
    for k=1:countp
        cardk=planningy(k);
        iii=floor((cardk+(n^0.5-1))/n^0.5);
        jjj=mod(cardk,n^0.5);
        if (jjj==0);
            jjj=n^0.5;
        end
        if (iii-1>0)
            if (project(iii-1,jjj)==1)
                project(iii,jjj)=1;
            end
        end
        if (iii+1<n^0.5+1)
            if (project(iii+1,jjj)==1)
                project(iii,jjj)=1;
            end
        end
        if (jjj-1>0)
            if (project(iii,jjj-1)==1)
                project(iii,jjj)=1;
            end
        end
        if (jjj+1<n^0.5+1)
            if (project(iii,jjj+1)==1)
                project(iii,jjj)=1;
            end
        end
    end
end
count=count+1;
a=zeros(n^0.5);
for ii=1:n^0.5
    for jj=1:n^0.5
        if (ii-l>0)
            if (project(ii-l,jj)==1)
                a(ii,jj)=1;
            end
        end
        if (ii+1<n^0.5+1)
            if (project(ii+1,jj)==1)
                a(ii,jj)=1;
            end
        end
        if (jj-1>0)
            if (project(ii,jj-1)==1)
                a(ii,jj)=1;
            end
        end
        if (jj+1<n^0.5+1)
            if (project(ii,jj+1)==1)
                a(ii,jj)=1;
            end
        end
    end
    if (a==1)
        break
    end
end
pnum(iter,1)=p;
end
Appendix F. MATLAB Code – Oops Game “Build” Strategy

clc;
clear all;
% this program is for oops game when played under planning strategy
%n=number of cards (4 to 400)
%yard = array of drawn cards (size 1*n)
%card=Card at hand at each step of the game (totaly n steps)
%count=current step of the game ( 1 to n)
%project=matrix of project (n^0.5*n^0.5 : 2*2,3*3,4*4)
%oopsy=array of oops yard
%o= # of oops
%b= # of builds
onum=zeros(1000,1);
for iter=1:1000
n =81;
o=0;
b=0;
oopsy = zeros(1,n);
yard =randperm (n);
project=zeros(n^0.5);
count = 1;
%counto counts for oopsy
counto=0;
%checks if all the remaining cards have the
%possibility to be built with no oops happening (like if we have gotten
%2,5 and 8 already) then all the remaining cards are just built. no
%planning any more
while count<=n;
    card=yard(count);
i=floor((card+(n^0.5-1))/n^0.5);
j=mod(card,n^0.5);
    if (j==0);
j=n^0.5;
    end
l=0;
if (count==1);
    project(i,j)=1;
b=b+1;
else
    if (i>0)
        if (project(i-1,j)==1)
            project(i,j)=1;
l=1;
        end
    end
    if (i<n^0.5+1)
        if (project(i+1,j)==1)
            project(i,j)=1;
l=1;
        end
    end
end
end
101
end

if (j-1>0)
    if (project(i,j-1)==1)
        project(i,j)=1;
        l=1;
    end
end

if (j+1<n^0.5+1)
    if (project(i,j+1)==1)
        project(i,j)=1;
        l=1;
    end
end

if (l==0)
    oopsy(counto+1)=card;
    counto=counto+1;
    o=o+1;
end

if (l==1)
    b=b+1;
end

for kk=1:n
    for k=1:counto
        cardo=oopsy(k);
        iii=floor((cardo+(n^0.5-1))/n^0.5);
        jjj=mod(cardo,n^0.5);
        if (jjj==0);
            jjj=n^0.5;
        end
        if (iii-1>0)
            if (project(iii-1,jjj)==1)
                project(iii,jjj)=1;
            end
        end
        if (iii+1<n^0.5+1)
            if (project(iii+1,jjj)==1)
                project(iii,jjj)=1;
            end
        end
        if (jjj-1>0)
            if (project(iii,jjj-1)==1)
                project(iii,jjj)=1;
            end
        end
        if (jjj+1<n^0.5+1)
            if (project(iii,jjj+1)==1)
                project(iii,jjj)=1;
            end
        end
    end
end
count=count+1;

a=zeros(n^0.5);

for ii=1:n^0.5
    for jj=1:n^0.5
        if (ii-1>0)
            if (project(ii-1,jj)==1)
                a(ii,jj)=1;
            end
        end
        if (ii+1<n^0.5+1)
            if (project(ii+1,jj)==1)
                a(ii,jj)=1;
            end
        end
        if (jj-1>0)
            if (project(ii,jj-1)==1)
                a(ii,jj)=1;
            end
        end
        if (jj+1<n^0.5+1)
            if (project(ii,jj+1)==1)
                a(ii,jj)=1;
            end
        end
    end
end

if (a==1)
    break
end

onum(iter,1)=o;
end