EXPLORATION OF THE NATURAL DESIGN STRATEGIES OF NOVICE ENGINEERS

BY

MARK A. SHERMAN
B.S., UNIVERSITY OF MASSACHUSETTS LOWELL (2008)

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF MASSACHUSETTS LOWELL

Author .....
December 1, 2010

Certified by .....
Fred G. Martin
Associate Professor
Thesis Supervisor

Certified by .....
Michelle Scribner-MacLean
Assistant Professor
Thesis Reader

Accepted by .....
Jie Wang
Department Chairman

Exploration of the Natural Design Strategies of Novice Engineers

by

Mark A. Sherman

Abstract of a thesis submitted to the faculty of the
Department of Computer Science
in partial fulfillment of the requirements
for the degree of
Master of Science
University of Massachusetts Lowell
2010

Thesis Supervisor: Fred G. Martin
Title: Associate Professor

Thesis Reader: Michelle Scribner-MacLean
Title: Assistant Professor

Abstract

This project explores how middle school students approach design problems, focusing on testing and iteration behaviors. Students were asked to solve design problems and create generalized processes for solving them. Observations of the students were analyzed using new methods for the characterization of testing and design iteration. These data yielded patterns of testing behavior intrinsic to the specific students, as well as patterns within individual activities. From these patterns, guidelines for engineering activity design were created.

Students participated in five 60-minute activity sessions. Each session presented one activity that was focused on a specific engineering discipline. The disciplines include math problem solving, electrical engineering, mechanical engineering, and computer science. The subject areas that received treatment included parallel and series circuits, gear reduction and LEGO construction, algorithm design, real-time control systems, requirement satisfaction, and working within time constraints.

Students were video and audio recorded. Students were encouraged to talk through their process and were regularly prompted by investigators to verbalize their thoughts explicitly. The data were coded for important behaviors. From these codes, data on student iteration and testing patterns were extracted and analyzed. New methods of characterizing and analyzing testing and iteration were created.

It was concluded that many factors were related to the success of the student design. The time spent exploring the problem before starting testing and iteration was the most significant factor. Other factors included the speed and consistency at which iterations were conducted. Recommendations for the creation of design-based engineering activities include scheduling techniques, introduction methods, and use of simulation tools.

Acknowledgments

The work in this thesis was only possible because of all the helpful and guiding members of the UMass Lowell engineering education community. Many people have helped me not just in research effort, but in introducing me to new thoughts and ideas that became foundational to this project.

I would like to first thank Dr. Michelle Scribner-MacLean for being so inclusive towards me. She involved me in her work, where I learned about many of the foundational works that appear in this document. It was not just about getting sources for the literature review, it was about having an experienced colleague. She taught me that everything is always a learning process, for students, teachers, and researchers alike.

I give my most sincere thanks to Dr. Fred Martin. He was patient and open-minded, yet always loyal to the science and process of development. I would like to thank Dr. Martin for years he has already invested in me, the care he has taken to see me succeed, and the all the opportunities he has afforded me.

I thank Howard Sticklor, who set up the program upon which this research is based. I very much appreciate his support of both my work and that of our student participants.

I extend thanks to Dr. Sarah Kuhn and Michael Penta, with both of whom I have shared many exciting conversations on education methods and research. Your ideas are woven into this thesis just as much as my own.

I thank my parents, Karen and Barry, who have been supportive and encouraging in all my endeavors, no matter how daunting. I believe that their continued reinforcement (and often reality checking) has been critical in all of my successes. They sparked my

interest in learning and teaching early on in my life. Their stories are inspirational to me, and I try every day to be as hard working as they are.

I would lastly like to thank my closest friends who have supported me through every phase of this process. Nick McKinnon, Mary Angeleri, and especially Stacy Kadesch, I thank you.

This material is based upon work supported by the National Science Foundation under Grants No. DRL-0624669 and No. DGE-0841392.*

*Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Contents

List of Figures

List of Tables

Preface

This research is an exploration in engineering education. My story leading to this research starts three years ago with a reading group on engineering and design where, as an undergraduate, I started learning about the current research in the field. Many of the foundational works of this study were first introduced to me in that reading group. I then spent two years working hands-on with middle school students and design activities. In an after-school program, I was a mentor for groups of 6th, 7th, and 8th graders, and provided them with experience in constructing and programming robots and robotic devices. This program covered many different concepts, from basic electricity to functional software design. Recently, I have been a part of a proposal writing process that exposed me to many different models of design, and showed me that there are significant and seemingly incompatible variations among many of them. From here my first concept for this research was conceived: to build a model of the design process as observed in middle-school students. This would be useful in the field of engineering education not just as another model, but as a tool to understanding this specific age of students.

Chapter 1

Introduction

Engineering education lacks many of the well-developed and thoroughly-tested tools seen in more traditional educational subjects. In most areas of education there are volumes of methods for teaching and assessments measuring effectiveness of interventions. These mechanisms of instruction and assessment are generally well-tested in laboratory and field settings. Design and engineering lack such a body of work, as Wilson and Guzdial (2010, p. 35) stated about the field of computing. Schools across the United States are adopting engineering curricula at nearly all levels of education, yet there are few research-supported mechanisms for understanding their effectiveness.

This study intends to address this opportunity. The study was conducted in the Department of Computer Science in association with the Graduate School of Education. The activities presented were technical in nature. Design and analysis of these activities required deep domain knowledge in their respective engineering areas. The researchers possessed experience in these design fields, as well as knowledge in the field of education.

1.1 Research Focus

This study observed middle school students as they executed short engineering design activities. In the state of Massachusetts, middle school may include fifth through

eighth grade. The students were introduced to engineering activities spanning multiple disciplines: electrical engineering, mechanical engineering, math problem solving, and computer science. The students were tasked to solve the given problems and create generalized processes for solving them. The focus of the data analysis was on testing, improvement, and iteration of the designs. New methods of characterizing and analyzing design iteration were created.

1.2 Problem Statement

This study is an exploration, and as such is interested multiple, closely related dimensions. The four questions of interest are:

- Do students exhibit patterns in testing and iteration? What are those patterns?
- What characteristics of a design activity elicit specific iteration patterns?
- What is the correlation between iteration in designing and success of the design?
- What guidelines can be written for the creation of future activities?

1.3 Approach

This research used a laboratory study of a small sample population of middle school students. Middle school students are inherently novice designers, where they are highly unlikely to have received any previous formal engineering training. The students participated in five activities, one per week. Each activity represented a different discipline of engineering. Problem difficultly increased over the five weeks. The students were instructed and coached in thinking aloud to gain access to their tacit, internal processes. The sessions were video and audio recorded. The data was coded for many different design behaviors, informed by (Welch, 1999). From this data information about testing and iteration was extracted, and methods of characterizing testing and iteration were developed.

1.4 Hypothesis and Contributions

Here, each research question is framed in more detail:

Do students exhibit patterns in testing and iteration?

It is expected that individual students will demonstrate personal trends across all activities.

What characteristics of a design activity elicit specific iteration patterns?

The design activities were designed to differ in complexity, speed of construction, and level of abstractness. These types of properties are expected to have a specific effect on iteration patterns in students, resulting in each activity having general trends that cross all students within the specific activity.

What is the correlation between iteration in design the design process and the success of the final design?

Multiple sources in the literature depict iteration as critical to design success. For example, Dow et al. (2009) showed that, in college students, forced iteration makes an inexperienced designer just as good as non-iterating designer who has domain experience. In this study, it was expected that rate and count of iteration would strongly correlate with success. Each activity is analyzed with individual success metrics, so this hypothesis was tested within each activity separately.

What guidelines can be written for the creation of future activities?

Each activity was expected to result in a certain unique pattern of testing and iteration. By comparing these patterns, it would then be possible to generate recommendations on properties of the activities themselves. These guidelines could be generalized for use by educators.

1.5 Rationale

Exploring the tacit processes of novice engineers will further our understanding of human design faculties. With this study, motivations behind why students engage in certain models of behavior was explored. This study chose to focus on iteration, a single component of the engineering process. As is discussed in the Background chapter, iteration is a fundamental characteristic of the design process. This study chose to use design iteration as a lens for analysis of the student design work.

Chapter 2

Background

Education has evolved with culture and technology. Classical wisdom calls only for the "three R's" (reading, writing, and arithmetic) in schools, which today is considered foundational but insufficient for life. Education has been expanded to include history, science, advanced mathematics, art, and technology. The current focus on scientific and mathematical education is called "STEM," referring to Science, Technology, Engineering, and Math. Engineering and technology now aspire to sit with equal prominence as science and math. This study observes the development of design thinking in middle school students, and investigates a few of the skills necessary to get there.

2.1 Brief History of Engineering Education

STEM education, of which engineering is a component, is in need of "evidence-based" tools to measure their effectiveness (Wilson and Guzdial, 2010). With that, much of the literature used is from closely related academic fields, specifically science and math education. Despite the larger depth of work in those fields, they all succumb to a problem inherent in education research: there is a disconnect between locally generated and generalizable knowledge. Techniques generated in classrooms that are locally usable and effective often do not translate well to other classroom settings. While that knowledge may work in one circumstance, it is little beyond anecdotal to

the greater community. Conversely, lab-generated scientific data on education is often too abstract to be directly usable for real teachers (Sandoval and Bell, 2004).

With more emphasis being put on technology in education, the development of computational thinking becomes important (Wing, 2006). Computational thinking is the collection of concepts, skills, and abstractions people need to best leverage computers and technology as an aid in human knowledge development. Thinking computationally is difficult, but with it people are able to solve new problems with the help of computation theory.

Brown (1992) conducted much of her work around the development of a learning community in lieu of the traditional didactic classroom experience. In a traditional environment, students are "passive recipients" of information dispatched by teachers and media. The relationship is largely unidirectional, where the only return of information from the students is from assessments that are based on drill, practice, and memorization. In this environment the students needs only to develop skills at storing rote facts and reproducing them on demand. Such an environment is prime for many pedagogical pitfalls, such as the development of a disconnect between knowledge and belief, where students may understand a concept as it was presented but do not believe it to be true (Chinn and Samarapungavan, 2001).

In Brown's community of learners, students begin to act as researchers and co-teachers of the material. The teachers, rather than function as managers assigning repetitive tasks, become facilitators who present the tools and encourage the curiosity necessary for engaging learning experiences. The teacher also serves as a good role model for learners, who show interest and discovery themselves. A classroom in this mode is no longer a work camp producing documents, but a research lab motivated by coherence and deep understanding. Assessments ideally are wrapped in immersive applications of knowledge such as projects and portfolios which can be subjectively as well objectively analyzed.

One of the fundamental skills outlined by Brown is the ability to self-monitor. Atman et al. (1999) support this, claiming self-monitoring to be a critical component of successful students. When a student engages in self-monitoring, the student can

identify when he or she is both succeeding and stuck, and takes appropriate actions to reach their learning goal. This skill considered part of meta-cognition, which is an advanced but teachable set of thinking skills (Beyer, 1988).

Strategies that are today considered meta-cognitive appear early and often in education literature. Bloom and Broder (1950) describe observed differences between successful and non-successful students as they approach problem solving. Successful students were better at, among other things, understanding problem requirements and maintaining contact with those requirements as they worked towards a solution. This is a fundamental element of self-monitoring.

Meta-cognition is, especially in children, hard to develop. Children empirically do not employ many meta-cognition techniques, if any at all. Most adults do, using strategies to overcome natural limitations of memory retention and recollection (Brown, 1992). The process of design has an intrinsic emphasis on meta-cognitive processes, as it has been shown to help students effectively learn about complex systems (Hmelo et al., 2000). The effectiveness of a design process is greatly enhanced by the inclusion of a feedback system, where the designer is re-analyzing both the problem and what he or she has done thus far to approach it. This concept is seen throughout the literature, often citing the "reflective practitioner" of Schön (1983).

The stages of cognitive development defined by Piaget and Inhelder (1969) help explain the slow development of metacognitive abilities. Children in middle school are only just entering the formal operations phase where they can perform reasoning on abstract representations. The metacognition skills required in engineering require abstract evaluation. Development of these skills is not possible until the child is cognitively ready, which does not generally happen until 15 to 18 years into life.

Building on Schön (1983), Adams et al. (2003) described iteration as the core tenet of design, triggered by certain cognitive activities: self-monitoring, clarification, and examination. Problem-setting in addition to problem-solving is emphasized. Reasoning is done through experimentation. A variety of representations are generated, and fluidly moved between to best serve the thought of the moment. The reflective practitioner is not just a developer making a solution, he or she is an experimental

scientist trying to understand the situation he/she created in solution development. These ideas, executed in tightly iterating loops, are the image of an ideal design strategy as described by Schön.

Many researchers and educators have attempted to model the design process. Most of these models have strong thematic similarities, usually including the following states: learning about the problem, identifying resources and constraints, generating ideas, implementation, testing, and revising. Despite the general similarities, different models can communicate completely different methods for design. For one example, Welch (1999) claims that the states of the design process are essentially unordered, allowing the designer to move laterally to any state at any time. On the other hand, Kimbell and Stables (2007) reported upon numerous models that are linear or have a tightly-prescribed progression through steps. The multitude of contrasting views of the design process is currently a source of conflict in the education community (Scribner-MacLean, 2009), and is an important topic of consideration in this research.

The Boston Museum of Science (2008) created a model designed for use by teachers who may not themselves be trained designers. It is intended to be used as part of a curriculum about design for young students, and may not be based on professional methodology. In this model five states form a ring, where travel is only implied to be possible between adjacent states. The states are Imagine, Plan, Create, Test, and Improve. The literature accompanying this model expresses that the student does not need to be bound by this ring, and may move from any state to any other state at any time. However, the graphic does not represent this and strongly implies the notion of neighbor-only traversal.

Adams et al. (2003) presents a model derived from observations of senior-level undergraduate engineering students. This model is scientifically accurate to an actual process that took place in research subjects. It is both more complex and less beautiful than the Museum of Science model, and is clearly intended for a different audience. Many would argue that the model presented by Adams is more useful, as it represents a functional process, but that depends on the individual's definition of usefulness. To an elementary school teacher, the Museum of Science model, while clearly incomplete,

may be the correct tool for the job at hand.

Most of these models have a heavy emphasis on iteration. Dow et al. (2009) found that iteration was critical to success of a time-constrained design problem. Participants with no prior experience who iterated through their design process performed as well as the participants who had prior experience but did not iterate. The process of repeating design tasks helped the participant explore the problem space and become familiar with the relevant physics. Eckert et al. (2009) found that iteration also takes on additional influence in corporate engineering, where a new design can depend on the proven framework of previous designs.

2.2 Design Activity Studies

Welch (1999) performed an experiment on design activities carried out by seventh grade students. In this study, students worked in dyads to build the tallest paper tower out of a finite set of resources within a specific time period. Welch enumerated a design process as five steps that represented a general consensus of concepts available in literature. The steps are:

1. Understand the problem
2. Generate possible solutions
3. Model a possible solution
4. Build a solution
5. Evaluate the solution

Welch claimed that the actual process undergone by both professionals and amateurs of design would not be linear, and that it would recurse on itself many times. The activity was intended to be representative of a real-world engineering task, characterized by having a goal, constraints, and some criteria to recognize a successful solution. The students worked in pairs, which is also an element of simulating real-world design,

where most development is the result of combined efforts of two or more people working cooperatively.

Design of activities for this research was also informed by the model-eliciting activities of Lesh and Harel (2003). Lesh's work was based in mathematics education, and this paper used fractions and proportions as the subject to be explored by the research participants. A model-eliciting activity (MEA) is designed such that the students' product is not a single answer, but a rule or process that can be applied to solve similar problems. Three MEAs are presented in the cited paper. The simplest example is the "bigfoot" activity, where students are told they are forensic examiners and need to identify the height of a person based on a shoe print. The expected response is for students, working in small groups, to observe themselves, and somehow create a proportion between human shoe size and height. The students are not requested to determine the height of the one example person, but to provide the police with a mechanism with which they can identify the height of any person based on shoe size.

The results of the MEA research, besides the concept of the MEA itself, is that problem solving can be viewed as a process of local concept development. Many researchers in the past have studied the process of learning concepts, specifically in math and science. Lesh concluded that the same process and principles involved in normal concept learning occur in a fast, recursive fashion during problem solving. The concepts being developed during problem solving are not necessarily general, they are specific to the current problem and the situation surrounding it, but the same development process has been observed. This observation allows the connection to be drawn between local problem solving and general, long-term cognitive development.

2.3 This Study

Many studies have examined the design process, providing a variety of insightful models and theories. Some of these studies, such as Schön (1983) and Eckert et al. (2009), model how adult, professional engineers perform design tasks. Others focus on

children, like Welch (1999) and Lesh and Harel (2003). These latter studies provide a look at how students generally go about design tasks, but they do not examine the elements of what makes up a design process. Models such as that presented by The Boston Museum of Science (2008) provide states of thought that students and professionals alike seem to pass through, but do not investigate the component actions that make that process what it is.

One of the most fundamental of these components is the process of iteration, about which study has just begun. Dow et al. (2009) analyzed the increase in efficacy provided by the presence of an iterative process in college students. This study shows that iteration is verifiably critical part of effective design. The author is not aware of any studies that focus on middle school students and the role of iteration in the design process. This study takes this focus, observing and interpreting students' design processes across a variety of problem types.

Chapter 3

Methodology

Students participated in five 90-minute activity sessions, once per week, for five consecutive weeks. Each session presented one activity that was intended to exercise different engineering and design principles. Students were observed and video recorded as they carried out the activities.

3.1 Subject Selection

This study used seven middle-school students as subjects. Students were in grades seven and eight, with average age of approximately 12 years old. The students reported that the grades they received in school were average or better, with the centroid of the distribution being 'B' marks. The selection process was done by an after school enrichment program, where this study was advertised as an activity in which students could elect to participate. This enrichment program served students from a region that is academically under-performing. The students were nominated by a teacher or administrator for having talent for learning, even if the students' grades do not reflect that talent. Five of the six students identified themselves as a minority race. This study was open to all students of the specified grade levels who were active in the after school program. Selection was simply the first students to volunteer to participate.

All students read, understood, and signed a student assent form and their guardians read, understood, and signed a parental consent form (see Appendix C.1). Parental

consent forms were available in English and Spanish. The student assent form was available only in English, as that is the language the research activities were conducted in. Guardians had the option to sign a media release form, which allowed recorded media of their child to be used in publication.

3.2 Session Protocol

In each session, students engaged in an engineering and design activity. Their conversations, problem-solving strategies, and concluding interviews were observed in person by researchers and video recorded for later analysis.

Each of the five sessions followed the same agenda. Students arrived and were brought to a conference room for snack time, where they were casually introduced to the concepts involved in that day's activity. Loud thinking protocol, in the style of TAPPS, was explained and reinforced every week at this time. Loud thinking is a process of talking through one's thoughts as a stream of consciousness (Lochhead and Whimbey, 1987). After about twenty minutes in the conference room, the students moved to the activity room where the apparatus for that day's study was set up. Working in pairs, students worked through the prescribed activity. The entire time in the activity room was audio and video recorded. Throughout the activity students were prompted by researchers to explain their thoughts, ideas, and understanding of concepts.

At the conclusion of the activity, students were led in a group conversation about what they learned and developed. These conversations were also recorded. The questions asked by researchers during this conversation were variations of:

- What did you do to find your solution?
- Did you do anything early on that you did again to help yourself?
- If you had a friend who was coming in to work on this problem tomorrow, what would you recommend they do?
- Are there any other tricks you discovered?

Week	Activity	Related Field
1	"Rush Hour" Game	math problem solving
2	Light Optimization	electrical engineering
3	Gear Reduction	mechanical engineering
4	Word Search	computer science
5	Elevator Control	computer science

Table 3.1: List of design activities, in order that they were conducted by students, with their related field.

3.3 Design of Activities

The activities performed by the students were designed in advance to cover a large sample of design problems. Each activity stressed different areas of engineering space, starting with a simple activity and progressing towards complex algorithm design. In total, the activities are intended to support computational thinking, as discussed in Chapter 2. The activities were performed a week apart from each other, which allowed for changes to be made to the activity plans based on preceding sessions. Every activity represented a real engineering or design problem, had observable iteration behavior, defined metrics of success, and were process-oriented, as discussed in Section 2.2.

Subject areas of activities included:

Parallel and series circuits: the two fundamental arrangements of resistive components in electrical circuit design, their properties, and their tradeoffs.

Gear ratios and reduction: the combination of different sized gears to convert between high-speed and high-torque, used in most motor applications.

Algorithm design: the design of a processes or set of rules that can be executed on a data set to arrive at a particular solution.

String-matching: a category of problems that involve identifying sequences of characters within a greater set.

Active control systems: a category of systems that use real-time feedback, typically to control a moving machine.

In addition to subject areas, the activities stressed different elements of design skills:

Balancing between contradicting requirements: a common problem in engineering, where satisfying two requirements simultaneously is impossible, and a compromise must be considered.

Working within time constraints: the skill of assessing how deeply a problem can be examined in the amount of time provided as not to be unfinished at the conclusion of that time.

Physical construction: a difficult task in general.

Symbolic manipulation of a system: performing design tasks abstractly by use of generalized symbols.

Each activity had two levels of success. The first level is the completion of a successful design that solves the problem. The second level is synthesizing a general process for arriving at a solution. Both levels are defined as success criteria for each activity in the following sections. The five activities are now presented. They are also summarized in Table 3.1.

3.3.1 Week 1: Rush Hour

"Rush Hour" is a sliding tile game where the player must manipulate cars on a small grid. The object of the game is to slide the cars so that the target car, indicated in Figure 3-1 by the arrow, can escape through the opening at the right side of the game board. The game has a small set of simple rules, but a high number of possible states for any given puzzle. Every time a piece is moved, the game is considered to be in a different state. Certain states are only accessible from other specific states. For example, the top-left car in Figure 3-1 can only be moved to the top-right position after the truck in the upper-right has been moved out of the way. All of the states in which that car is in the top-right position are only accessible through a selection of states where the truck has been moved. The total number of states possible for this game is the total number of unique combinations that the cars can be arranged in, which is large space.

Rush Hour Specifications		
Success Criteria	Solution:	Solved puzzle
	Process:	Described Strategies
Iteration Metric	Backtracking or resetting: deviation from current course	

Table 3.2: Rush Hour activity success and iteration criteria.

Purpose

This activity was an ideal introductory experience for the students. It had few rules to learn and was fun to play. Students were quickly engaged; most of them had played the game before. As a design activity, this provided observations of the students' methods in navigating large number of possible states of the game. With every move of a game piece, the possible states accessible on the next move change. Many times, the path the player is on does not have a solution, and the player must backtrack to reach an earlier state where a different decision could yield a solution, or, start over and try again. This property is known as a "deep state space." The backtracking behavior was a primary component of the analysis of this activity.

Welch cited Ericsson and Simon (1984) with the importance of a warm up process, stating that when verbal information exchange is involved, a period of warm-up is necessary for the subject and researcher to communicate most effectively. This activity, with its simple rules and familiarity, provided an opportunity for students to focus on the loud thinking strategy, warming up that skill for the remaining sessions.

Protocol

The students were each given a game unit with a beginner-level puzzle to solve. The students were encouraged to "think out loud" during the entire process, and to think about the process they were employing to arrive at solutions.

Once the students felt that they understand the concept of the game, they were put into pairs and given a more difficult puzzle. During this phase, a camera was positioned above the table to record the game board, the students' hands, and the students' voices. Additional puzzles of increasing difficulty were used as time allowed.

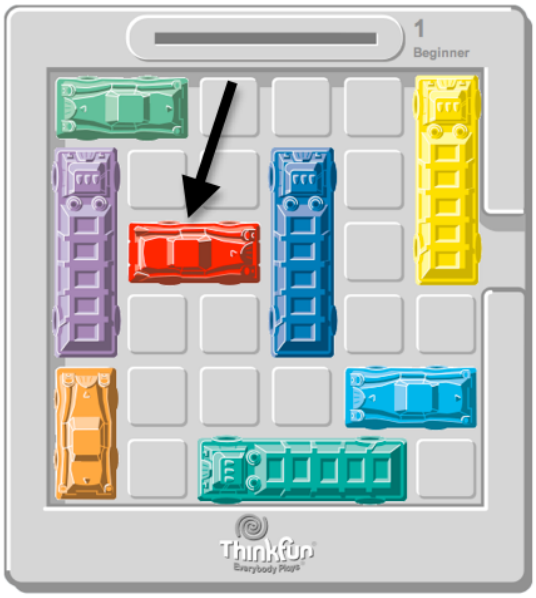


Figure 3-1: Rush Hour game board. The arrow indicates the target car.

During the session students were asked to identify the "key move" of puzzles that they had solved. The "key" is the critical move that unlocks the puzzle, allowing it to be solved very easily.

At the conclusion of the puzzle solving session, the student pairs were asked to explain the process of solving the puzzle. The students were instructed to explain their process of finding a solution, not the steps to carry out a specific solution. They were then instructed to explain the game strategy for a friend who had not seen the game previously.

A semi-structured interview was used to probe students' understanding. The interview was based on these questions:

- Please explain what you did to find the solution to these puzzles.
- Did you do anything in the first puzzle that you did again in the second to help yourself out?
- I want you to explain how to go about solving these kinds of puzzles. Pretend you're explaining it to a friend who has never done this before.
- Are there any tricks you discovered?

Light Optimization Specifications		
Success Criteria	Solution:	Built working circuit (plus formulaic score)
	Process:	Reasoning for design
Iteration Metric	Turning on power supply	

Table 3.3: Light Optimization activity success and iteration criteria. The solution criteria of building a working circuit is evaluated as true or false, but this activity is unique among those in this study in there is additionally a numerical score associated with the solution.

Rationale

The Rush Hour game was characterized by its large state space, as discussed above. Trying to solve it led to a rapid iteration of ideas and high degree of backtracking. This type of problem is common in computer science and mathematics, and can be generalized to include other games (such as chess) as well as real engineering problems. For example, circuit board routing is a design activity that has a large number of states where decisions progressively lead to new areas of the state map, fitting this general problem type.

3.3.2 Week 2: Light Optimization

The Light Optimization problem explored basic electrical principles and required students to balance between contradicting requirements. Students were presented with eight small light bulbs and a power supply. The design task was to connect the lights with alligator clip wires in a way that minimized cost without sacrificing brightness.

In the problem specification, wires and volts had costs associated with them. The students were given a scenario that they work for a power company, and each light bulb represents a house that they service. They needed to find a method to light all of the homes at the lowest cost to the company. Example student work is shown in Figure 3-2.

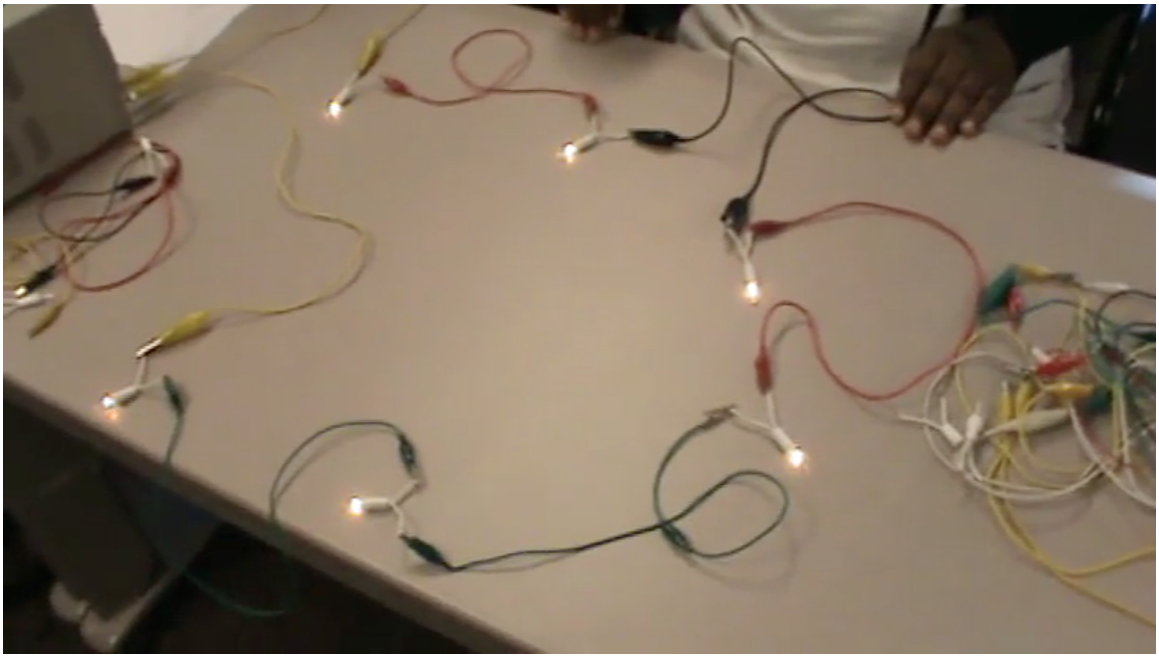


Figure 3-2: Example of solution to the Light Optimization activity. The lights are strung together in series by alligator clips.

Purpose

This activity generated observations of students exploring natural phenomena and balancing requirements for an optimal solution. A formula generated a cost per solution score, but students were allowed to construct arguments for the optimality of their solution. This was intended to further promote loud thinking and give insight to the student's thought processes.

Protocol

During the initial snack time, a researcher led a conversation about what "optimization" means, soliciting suggestions from the students. Once in the classroom, the students were presented with a workstation for each dyad. Each workstation had a variable DC power supply, a number of small, low-power light bulbs (holiday mini-bulbs), and connecting wires with alligator clips. Two workstations were initially set up with example circuits. One example was four lights connected in series, such that the same current flowed through all the lights, with the power supply operating at 12 volts.

The other station had the same number of lights connected in a parallel configuration, such that every light had full voltage from the power supply, which operated at 3 volts. The two systems were approximately the same brightness. The terms "series" and "parallel" were never used with the students.

The students were presented with a set of criteria for assessing their solutions. The goal was stated to achieve "reasonable brightness." Costs were associated for number of wires used and amount of voltage required. The final cost of a solution was

$$\frac{a \cdot \text{wires} + b \cdot \text{voltage}}{\text{number of lights}}$$

where a is the cost of wires and b is the cost per volt. The students were provided with this formula at the beginning of the session.

The students, working in pairs, solved two iterations of this activity. The two iterations had different material costs. The change in cost values was designed to obsolete the first iteration's optimal solution, forcing reconsideration of the problem.

At the conclusion of the second problem iteration, the students, in pairs, were asked to document their solutions and explain how they arrived at them. They were instructed to explain their process as if for a friend who was not present. Interview questions examined the students grasp of the electrical principles, what they considered to be "optimal," and how they went about discovering what they needed to know in order to solve the problem.

Rationale

The electrical concepts in this activity were fundamental to an understanding of electricity and electronics, which creates the physical basis of computing technology. The problem was also multi-dimensional, with number of connections, voltage, and number of lights serving as related but confounded design parameters (Suh, 1998). This activity emulates real-world design problems that require simplification, satisficing, or constraint of solution space.

Gear Reduction Specifications		
Success Criteria	Solution:	Built transmission that lifts mass
	Process:	Non-trivial suggestions for a friend
Iteration Metric	Applying energy to system, either by hand or by motor	

Table 3.4: Gear Reduction activity success and iteration criteria.

3.3.3 Week 3: Gear Reduction

In this activity the students built a transmission from LEGO Technic gears and components between a fixed motor and load.

Purpose

This problem allowed for direct access to the students' spatial and mechanical reasoning skills, a domain used heavily in many different areas of design.

Protocol

The students were introduced to the concepts of gear reduction, specifically the nature of "little-to-big" relationships (Martin, 1995). Each student dyad was supplied with a workstation apparatus, constructed of LEGO Technic with a motor, a support structure on which to build their transmission, and an output pulley. The pulley was connected to an interchangeable mass. The students were presented with two different tasks, and asked attempt either one or both:

1. Lift the greatest amount of weight possible.

2. Lift a specific weight from the floor as quickly as possible.

The students had all the remaining time to design and build their solutions. Paper was provided, and students were encouraged to document their transmissions in the same fashion as was used in the concept tutorial.

At the conclusion of the work time, each dyad's construction was tested individually as the group observed.

Word Search Specifications		
Success Criteria	Solution:	Created steps to solve puzzle
	Process:	Same as Solution
Iteration Metric	Testing modification to algorithm	

Table 3.5: Word Search activity success and iteration criteria. This activity is unique in that the solution and process are the same: the student's desired product is not a solved puzzle but the algorithm to solve the puzzle.

Rationale

This activity was an engineering design task. The requirements were simple, and the constraints needed to be discovered by the students. This activity was more than just a gear design test; it required construction of a stable structure which is alone a significant task. The solution was open-ended, as the gears and other components could be assembled in nearly infinite configurations, many of which may solve the problem. This activity stressed implementation and time management.

3.3.4 Week 4: Word Search

Students designed algorithms to solve word searches, where words were hidden in a two-dimensional grid of otherwise random letters. This activity introduced the concept of algorithm design. The puzzle and worksheets can be seen in Appendix A.

Purpose

The concept of an algorithm was introduced in this activity, which was also used in the following activity. As such, this activity partially functioned as a warm up to the more complex elevator control algorithm activity.

This activity allowed observations of students process in deducing and applying patterns, and observed students' approaches to writing generalized procedures.

Y	E	S	L	F
M	N	H	R	I
P	T	O	U	R
A	C	E	M	E

Figure 3-3: Example word search puzzle provided to students.

Protocol

This protocol was implemented as a worksheet for students, as can be seen in Appendix A. The concept of an algorithm was introduced with the game of tic-tac-toe as example. The students were given the instruction set to never lose tic-tac-toe, and were encouraged to study and try it. Once the concept of using instructions to play a game had been mastered, the students were given a very simple word search puzzle. The example puzzle was designed to be nearly trivial, and is shown in Figure 3-3. The words hidden in this puzzle are *ace*, *fire*, *shoe*, *tour*, and *yes*.

The students were prompted at this point with questions, which were discussed briefly as a group. The questions were:

- Did you find all the words?
- What did you do to find the words?
- Did the words just "jump out at you?"
  - What if they didn't?
  - How can you know for sure you found *all* the words?

The students were then prompted to write down any rules or strategies they employed in the example puzzle.

The researcher explained that computers are devoid of the pattern recognition ability that makes words "jump out" for people. However, computers are very fast at simple computation, so they are advantageous to use for extremely large data sets. This provided impetus for the students to develop an algorithm rather than simply solving the puzzle one time.

Elevator Control Specifications		
Success Criteria	Solution:	Wrote control program that operates successfully
	Process:	Useful recommendations for a friend
Iteration Metric	Running program	

Table 3.6: Elevator Control activity success and iteration criteria.

The students were then given the difficult puzzle, which can be seen in Appendix A. The students were encouraged to continue talking through their thought process, and reminded that the process to solve the problem was desired, not the solution itself. If the students completed the puzzle and were confident in the rules they had written, they were given an additional puzzle with the instructions to solve it only by following their rules, not using their human abilities.

Rationale

This activity stressed computational thinking, which caused students to think algorithmically about a normally straightforward task. This activity also had opportunity for cleverness and creativity that arose from noticing patterns in data. These skills are fundamental to computer science.

3.3.5 Week 5: Elevator Control

Elevators are robotic devices that need to act based on multiple inputs and system states, and require well-thought-out control algorithms to function effectively. In this activity, students designed a set of rules to govern the motion of a simulated elevator. The simulation acted as a microworld, allowing students to experiment with a controlled subset of problem physics (Levin and Waugh, 1987). The simulated elevator was implemented in Scratch using the BYOB extension, allowing for custom drag-and-drop code blocks for high-order control of the graphical elevator system (Maloney et al., 2004; Harvey and Mönig, 2010).

The workspace for the microworld is shown in Figure 3-4. The representation of the building is the white panel on the right. The red rectangle (d) is the elevator

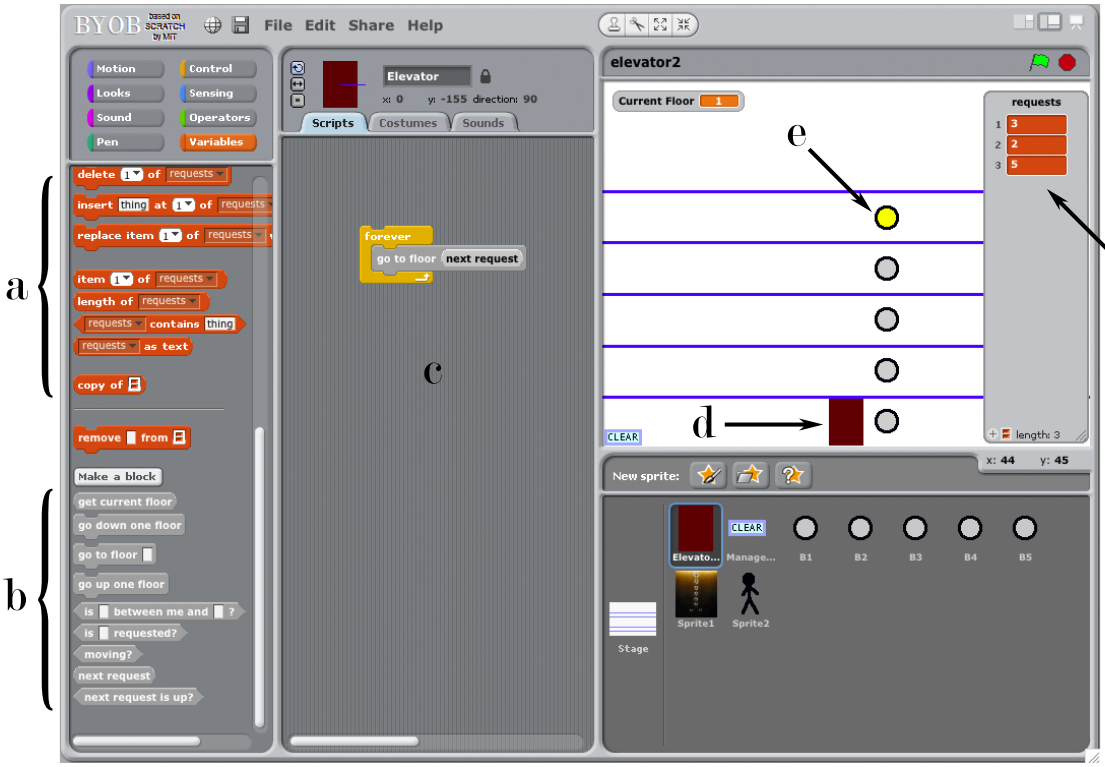


Figure 3-4: Workspace for the Elevator Control microworld. The components of the environment are: (a) Standard programming primitives, including logic; (b) Custom procedures for elevator simulation; (c) Code workspace where the procedures are assembled into a program; (d) Graphical representation of the elevator; (e) Call buttons that add requests to the queue; (f) The request queue, showing three elements.

car, shown at rest on the first floor. The floors are delineated by light-colored lines, and each has a gray, circular call button (e). At the far right of the white panel is a gray inset panel that contains a queue of floor requests as the simulator runs (f). The simulator automatically places a request on the queue when a floor button is clicked by the user.

Code is written by dragging primitives and control structures from the left panel to the center panel. In the center panel, the primitives can be assembled to form command, query, and repetition sequences. A set of procedures were made for the purpose of elevator control and were added to the palette. The custom procedures are shown in in Figure 3-5.

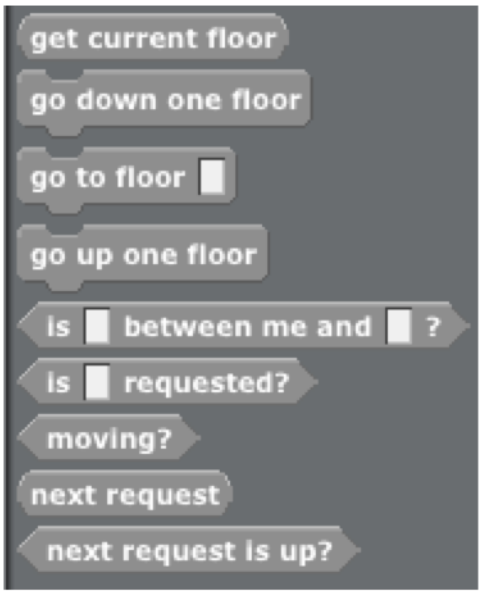


Figure 3-5: Custom procedures for elevator control that were built into the simulator.

Purpose

This activity was intended to explore the students reactions to a difficult optimization problem. When the activity was conducted, students participated in "problem-framing;" that is, making sense of the problem situation and forming their task goals. In this activity, the general goal was to create an improved elevator control program, but what needed to be "improved" was not clear to the students until they conducted sufficient "problem-framing." Unique among the other sessions, this activity encouraged students to think and work purely symbolically, where the tools and materials they manipulated were all abstract representations of real-world items.

Protocol

The students were given the goal of developing an algorithm to control an elevator. The concept of simulation was introduced, as was the Scratch programming language subset used by the simulator.

The students were provided with a simple but working starter solution, shown in Figure 3-6. The first challenge for the students was to understand why the given solution worked and why it was sub-optimal. There was no formal mechanism to

Figure 3-6: Starter solution of the Elevator Control problem. The "forever" structure will infinitely loop whatever code block sequence is nested within it. In this case, the only action is to "go to floor" with an argument of "next request." The elevator will continually go to floor whose request is at the top of the queue.

prevent students from working on a solution before they fully understand the problem, but they were prompted often by researchers to establish their current understanding of the situation.

The provided solution called the elevator to travel to the next floor to which it was called, in the order that the calls were made. The major flaw of this solution was that people who were waiting for service were passed by the elevator as it traveled blindly to the called floor. This inefficiency would result in unacceptable wait times (and therefore unhappy people), as well as unnecessary energy expenditure by the building owner.

Rationale

This activity is abstract, and requires students to manipulate a system by means of symbolic representation. It also requires understanding of a queue structure, positional queries, and boolean logic. These latter elements are part of the everyday problems posed by both researching and practicing computer scientists and engineers. Control elements, such as determining location and direction of travel, are specifically common to robotics.

3.4 Coding and Analysis

Videos from student sessions were analyzed using coding techniques informed by Welch (1999) and Scribner-MacLean (2009), where specific codes were defined to describe important behaviors. Additional codes were created in the style of grounded theory

(Strauss and Corbin, 1997): as behaviors and patterns that were not predicted were observed.

The body of codes covered a wide range of design behaviors, as seen in Appendix B.1, but only the TEST code was used in analysis, as it was used to characterize design iteration.

Analysis was done initially with all of the codes using NVIVO software. Codes were applied to specific time spans of the video where the corresponding behavior or activity was observed. A second phase of analysis was conducted that only included codes that represented testing. This phase generated greater detail and did not use NVIVO.

Data were analyzed to reveal trends in two directions. The first was to show trends across all students within each activity. The second was to show trends within each student across all activities.

Chapter 4

Analysis

The session videos were first analyzed using the coding scheme discussed in Section 3.4. Based on that first pass of video analysis, student iteration emerged as a critical factor to be evaluated. Iteration is defined in Section 4.2. The selection of participants for the final analysis is described in Section 4.1. The following Sections 4.4 through 4.8 describe the analysis of the individual activities.

In three of the five activities, a numerical analysis of observed student behaviors was developed. The three selected had complete data for each student, allowing analysis per student for the whole duration of the session. The three activities also highlighted the three critical engineering fields: electrical engineering, mechanical engineering, and computer science. From these criteria, the Lights Optimization, Gear Reduction, and Elevator Control activities were selected for this analysis.

The numerical analysis created quantitative data that represent the iteration behavior of the students. The metrics included were total number of iteration cycles, average time for each iteration cycle, and non-iteration time before testing. The non-iterating time expresses how long the student spent exploring the problem and creating an initial prototype before beginning to test. The analysis also includes the students' success ratings, as described in Section 3.3. Students that achieved the first level, a working solution, scored a single rating point. Those who achieved the second level, a generalized process, obtained an additional point, totaling two.

4.1 Subjects

The trial was conducted with seven participants, six of whom had sufficient attendance to be included in the analysis. Of the six students in the analysis, five of them were present for all the activities. The sixth student was absent once and missed the Elevator Activity. Code names A through F were assigned to these students to anonymize their data. Students worked in pairs, which was encouraged in most of the activities to facilitate talking through their thought processes. This method of eliciting communication is suggested by Welch (1999). When paired, the students always chose the same teams. This was beneficial to analysis, as the students could be considered part of a consistent group of two. The dyads are A/B, C/D, and E/F.

4.2 Working Definitions

A student was considered to be performing a "test" when a solution, or component of a solution, was applied to the problem in order to ascertain how well that solution or component solves the problem. A critical measure was that a test must be made against the world of the problem, whether it is real physics or a provided simulation, as specified by the activity. An attempted solution was not considered a test if it was being checked against an idea or vision internal to the student, as no new information is gained by the student. For example, one dyad in the Gear Reduction activity turned a series of connected gears by hand, and discovered that the were binding. Another example is from the Elevator Control activity, where a student ran a small part of code and observed the the elevator to ensure it responded as expected. Both of these are instances of tests. In a third example, from the Gear Reduction activity, a student assembled a component, held it up to inspect it, then immediately took it apart and rebuilt it differently. This is not a test, as the component was never validated against anything the real world, it was only checked against the student's judgement.

Iteration cycles are delineated by the student performing a test, with the first iteration starting with the first test of the session. The final iteration ends with the

last test performed in the session. The rationale for this bound is that the final test is the one to show "completeness," at which point the problem is deemed solved by the student (or the student gives up).

The time spent before the first test is conducted is considered to be "non-iterating time," during which the student is being introduced to and exploring the problem.

These definitions were generated from observation of the students, and are consistent across all data.

4.3 Characteristics of Iteration Behavior

This section defines three characteristics of iteration behavior: the percentage of time spent in preparation, the count of iteration cycles performed, and the average time per iteration.

4.3.1 Iteration Count

A primary characteristic of an iterative design process is the number of times that student was observed performing an iteration (or simply, "a test"). An iteration was defined by the student testing a design or a component of a design against the world, as discussed in Section 4.2. Every test was marked with how many minutes into the session it occurred. An iteration is considered to end with a test, so the number of test instances is equivalent to the number of iterations. The most iterations observed by one dyad in a single activity was nine, and the least was zero. Excluding the zero, which is a peculiar case, the lowest amount of iteration was three cycles.

4.3.2 Iteration Time

The time between tests is an indication of how long it took the student to assess the newly discovered information from the last test and integrate it into the solution. Knowing how long a cycle typically takes can be helpful in designing activity schedules. This metric is time between test events. The time from the start of the activity to

the first test was observed to be introductory and exploratory for the student, which is addressed in the next section. The time ends at the student's final test where the solution is declared to work or not, and development ended. Across the three selected activities, iteration times varied from one minute to over a half hour. Within a single dyad and one activity, the standard deviation of average iteration times did not exceed five except in one outlier case, which was nearly eighteen.

4.3.3 Preparation Time

The time before the first test was considered preparation time, during which the student explored the problem and assembled an initial prototype. The shortest preparation time was three minutes, observed in one group during the elevator activity. This task was a software activity, and made it very easy to "dive right in" and start trying things nearly immediately. The other two dyads in that activity waited ten and over twenty minutes before their first test. The preparation was usually around ten minutes, but the actual value of that absolute number was relative to the total time the student spent working on the problem. The preparation time is expressed as a percentage of the total time spent working on the problem, so that the metric can be easily compared across activities and students.

4.4 Rush Hour Activity

This activity provided data about student metacognition, as students were observed trying to understand their own problem solving patterns. Students of this age generally have not yet reached the formal operations stage of cognitive development (Piaget and Inhelder, 1969), and thus have not fully developed metacognitive skills (Beyer, 1988). These students demonstrated rudimentary ability to analyze their own thinking processes, but were often incapable of explaining how they came to a process or conclusion. Also, as supported by Beyer (1988) and other research, metacognitive fluency requires significant practice, so proficiency was not expected.

4.4.1 Expected Outcomes

The Rush Hour game is essentially a symbolic representation, where manipulation of the symbols is inexpensive (in terms of time) and easy to accomplish. It was expected students would demonstrate a high level of backtracking, iteration, and restarting. It was expected that students would run in to many dead end paths, and would have to backtrack through their motions or start over very often. The expectation of rapid backtracking was constructed from this ease of manipulation, lack of rules to process, and testing by the researchers.

This expectation was borne out to be correct. The period between backtracks and start-overs was often mere seconds, where the student had barely executed anything before coming up with another idea to try instead.

4.4.2 Observed Behaviors

In this section a number of interesting behaviors observed in students are described. These behaviors were seen in more than one student and potentially offer insight to the cognitive states of the students during the activity.

Unable to repeat

At the beginning of the session, it was observed that students were unable to explain how they came to a solution. They often claimed that they were unable to repeat it. This behavior was only observed in the first ten minutes of the activity.

Working backwards

Some students elected to work backwards from the goal towards the start state. This strategy was observed at all stages of the session. When using this technique, students never worked their way all the way to the start state. Once sufficient experience with this technique was acquired, the student reverted to a forward-direction method, where they were they were then more easily able to find a solution. Working backwards was

first observed only in one specific dyad. The other groups used it later in the session, possibly after hearing the idea from other students.

Accounting for impossibilities

Two instances were observed where students told researchers that their strategy included accounting for possible future scenarios. Both of these instances, one during the session and other during the wrap up interview, involved preparing for scenarios that were mathematically impossible. Student E described why he moved the red car away from goal was "in case a car had to move through," indicating the empty column he created with the move. There were no cars on that column aligned in that direction. There was no case where the possibility he described could happen.

Assessing limitations

Students often described that their strategy included assessment of limitations. Limitations were simple such as, "I can't move the green car," so the student would consider how to go about either freeing the green car, or disregard it as immovable. This behavior was very important in finally solving the puzzle, as it constrained the problem. This also seems to contradict the behavior noted above where students accounted for impossibilities, but both were observed in the same students during the same puzzles.

Identifying key move

Researchers prompted students in the middle and late portions of the session to identify the "key move" that unlocked the particular puzzle. When first prompted students had difficulty providing an answer, often stammering and indecisive to what the "key" was. Later in the session they became more confident in their answers. The answers were not necessarily right or wrong, but provided insight to what the student thought was important. One example showed a student identifying a single car moving to the far left as the "key." That student was confident in that identification citing that car as in the way of the remainder of the moves necessary to achieve the solution.

Almost any car could be argued to be critical on similar grounds. The student may have picked that particular one because it was the available move a critical in part of that student's process in solving the puzzle.

Theory development and testing

Students developed theories and ideas very rapidly, but did not generally test them fully. When testing a theory, the student would proceed a few moves, then interject with a new idea and pursue that. This is similar to when the students worked backwards, where they never worked through an idea to completion, but moved on when they gathered enough experience to think of something else.

4.4.3 Wrap Up Discussion

In the wrap up discussion, students made many suggestions on how to go about solving the problem, but these suggestions often lacked a substantive basis. Students provided summaries of their tactics that were inconsistent with the observations during the activity. Many students changed their responses while explaining them. Those students had a different answer at the end of their explanation than they had at the beginning.

There were some points that were made by the students that were supported by earlier observations. One of the first points to emerge from the discussion was that thinking out loud actually helped them solve the problem. The process of verbalizing the thought process had observably improved results in solving the problem. This observation is supported by Lochhead and Whimbey (1987) among others.

Students generated many strategies that they claimed worked for them; they are outlined below. These strategies contradict each other, but each one was defended strongly by the student who presented it. Every strategy presented that was specific to certain pieces was disprovable by counterexample. The first three strategies in the list below are in this category. The general strategies, however, could be useful if presented to other students, which was the question posed by the researchers.

- Move trucks down
- Move little cars first
- Move towards exit to get out
- Look for the key piece
- Go with the flow
- Look ahead at least two moves
- Take it slow

Students were asked how they knew when to backtrack or start over. The responses were as expected: when totally stuck or stuck in a repetitive loop. One student claimed he never restarted, only backtracked. Observation of that student supported this, which was an anomaly. Every other student did full restarts many times over the course of the session.

The student who never reset was also observed solving the puzzle but not realizing it. He set an intermediate goal and was focused on solving that when he created an opening to solve the whole puzzle, but he did not notice that opportunity.

4.4.4 Rush Hour Results

Students were observed iterating extremely quickly, often with only seconds between iterations, as seen in Figure 4-1. Students were unsure about their own processes, and often contradicted themselves when explaining how they developed strategies. Very little process description provided by the students was supported by the observations during the session.

This game was a good introductory activity because its easiness to learn.

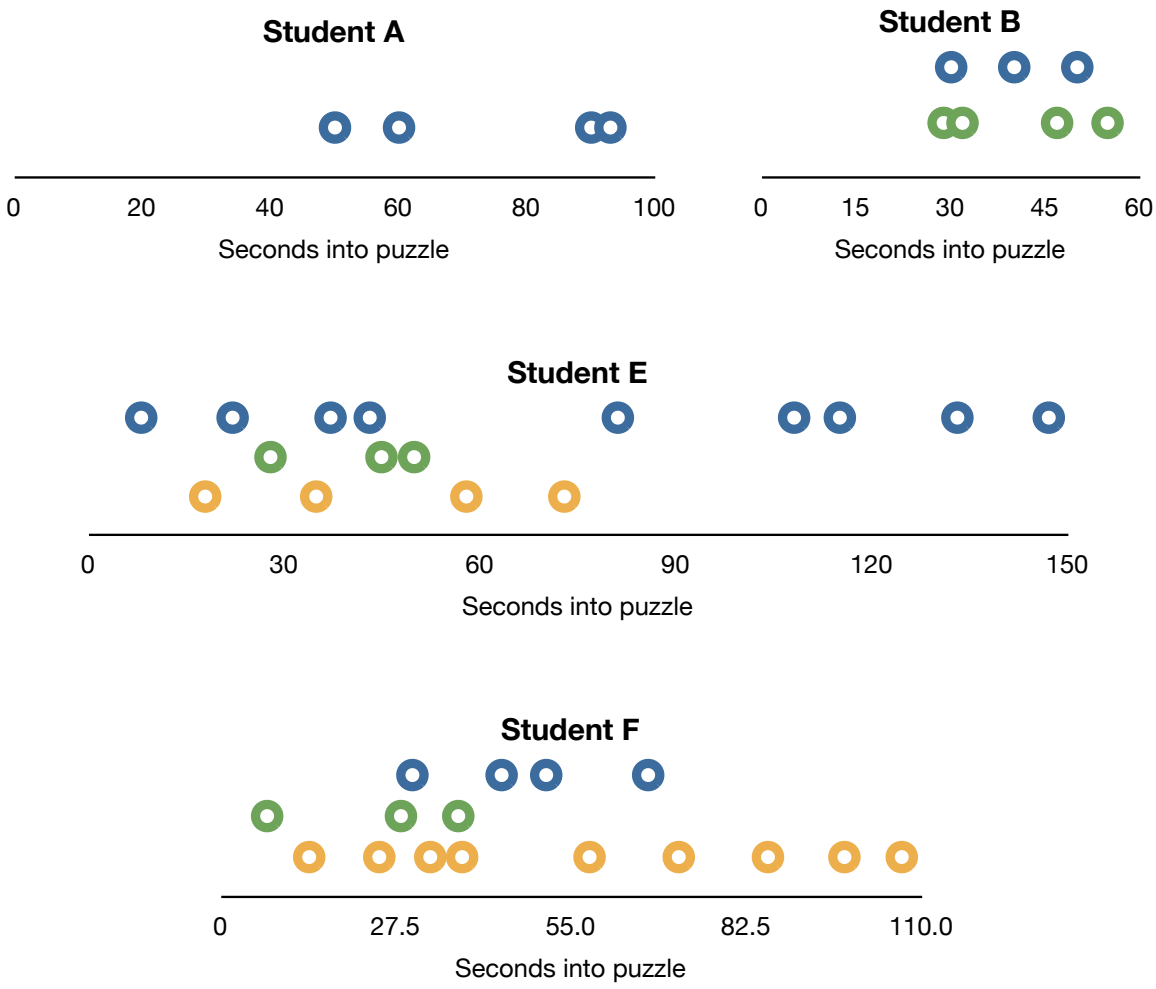


Figure 4-1: Rush Hour Activity iteration timelines. Each graph represents one student. Each mark represents a test occurring at the specified time. Each row is a different puzzle that the student completed. The horizontal axes are time, in seconds.

4.5 Light Optimization Analysis

This activity had a well-defined schedule and a structured metric of design success, which was made explicit to the students in the form of a mathematical calculation. The session was broken into two phases, with each phase containing a complete problem solving cycle. The second phase presented the same problem as the first, but with component price values modified, changing the location of the optimal point in the solution space. Testing was also very deliberate, as it required the use of the power supply. Rules were in place that students could only turn on the power supply when their system was safe and nobody was touching the circuit. Unlike other activities, test feedback could not be generated without the power supply being on, making the test cycles relatively formal.

For students, the metric of design success was based on a formula, and students turned to arithmetic throughout the activity to help them plan and check their designs.

4.5.1 Expected and Observed Outcomes

With two deliberate design phases, it was expected that students would be more successful in the second phase. The first phase was expected be mostly exploration, and the second one was expected to have more frequent iteration and testing cycles. This was shown to be at least superficially accurate, as more iteration cycles occurred across all the students in the later portion of the session. When looking at the entire session as a whole, however, the concentration of iterations was congruent with that of other activities that did not have deliberate phases. It is possible that the increase in testing cycles indicates a greater pattern of developing comfort with the problem over time, and the forced phases may have had little effect.

4.5.2 The 1-Wire Misconception

A misconception was observed that was completely unexpected and provided a serious problem for the students experiencing it: thinking power can flow over a single wire. Students with this misconception were unable to build a working circuit until they

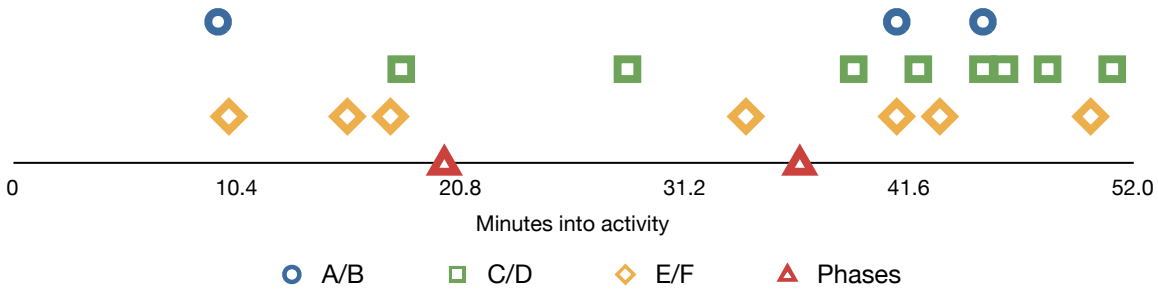


Figure 4-2: Light Optimization Activity iteration timeline. Each mark represents a test occurring for the specified student dyad. Each row represents a different student group. The triangular marks indicate the end of the first design phase and the beginning of the second. The horizontal axis is time, in minutes.

received additional corrective instruction from the researchers. The students C and D created their first design to look like a lollipop: one wire came from the power supply and connected to one point in a series loop of lights. The other terminal of the power supply went to a similar loop. Clearly this circuit did not function, as there was no circuit created between the two terminals of the power supply.

4.5.3 Voluntary Use of Math

The most unexpected and significant observation made during this session was the students voluntarily using written algebra to help their designs. Without any prompting from researchers, students took it upon themselves to manipulate the scoring formula to find how many lights and wires they wanted to strive for to make a certain score. They then revisited that formula after testing to validate their design. While the subjects were above-average students, using mathematics in both planning and validation is usually only observed in designers with engineering training.

4.5.4 Light Optimization Results

Figure 4-2 shows a timeline of all iterations performed by students during this activity. The first design phase ended at 20 minutes, and is marked by the first triangle on the timeline. The second phase, with modified design criteria, began at 36 minutes, and is indicated by the second triangle. In the time between the two phases, students

	A/B	C/D	E/F
Iteration Count	3	8	7
Time per iteration	18 min	5 min	7 min
St.Dev. of time per iteration	14.8 min	3.7 min	4.8 min
Non-iterating time	21%	35%	20%
Success (0,1,2)	2	1	1

Table 4.1: Results from the Light Optimization activity. The best performing team had the fewest iterations, which is believed to be based on that team utilizing prior knowledge.

analyzed their first design and prepared for their second. Two groups performed tests during this period, as can be seen in the figure. Additionally, this figure shows that the second design phase contained many more iterations than that first for all groups.

This activity was included for numerical analysis. Table 4.1 shows that the most successful student group (A/B) had the fewest number of iterations. This dyad utilized prior knowledge to deliver a working design on the first try and were satisfied with that design for the remainder of the session. That student pair recognized the holiday lights and immediately recalled how a string of Christmas tree lights work. They proceeded to build a series string, like that of the Christmas tree, and had a working solution very quickly. The second phase of the activity forced a redesign by changing the parts costs, but group A/B determined their solution also satisfied the revised requirements, and they left their design essentially unchanged.

Putting aside the above dyad, the other two groups (C/D and E/F) were fairly similar in iteration count, average iteration time, and non-iterating introductory time. The biggest difference between them was that group C/D spent more time in introduction before beginning testing. Group C/D had the 1-wire misconception of electricity which gave them a disadvantage, as discussed in Section 4.5.2. Both of these groups met the first level of success.

The most significant observation was the students' self-motivation to use mathematics to improve their design, discussed in Section 4.5.3.

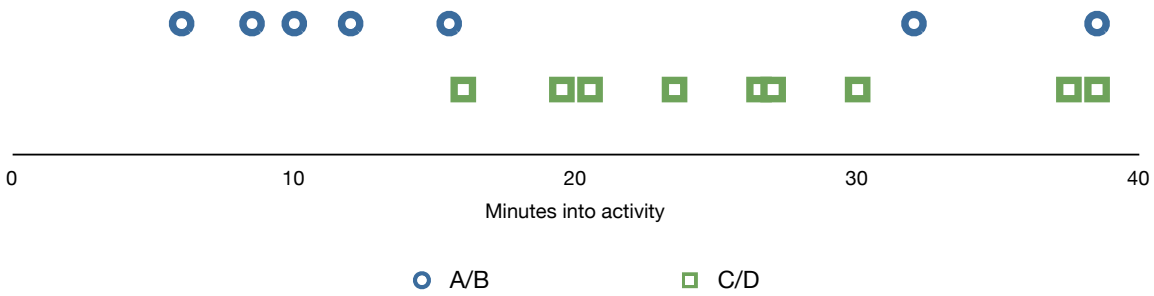


Figure 4-3: Gear Reduction activity iteration timeline. Each mark represents a test occurring for the specified student dyad. Each row represents a different student group. The horizontal axis is time, in minutes.

4.6 Gear Reduction Activity

The Gear Reduction activity had the students build LEGO structures that enable a small motor to lift a mass via gear reduction. This activity is complex, and required not just an understanding of gear reduction, but of LEGO construction as well. This proved to be a harder problem than anticipated by the researchers.

4.6.1 Expected and Observed Outcomes

This activity was intended to be an exercise in gear spacing, meshing, and reduction. Students were expected to choose testing patterns somewhere on the continuum between large numbers of small, local tests and small number of large, entire-system tests. The best solutions were expected to come from students who iterated more often. This hypothesis was demonstrated to be generally correct, but was complicated by the unexpected degree of difficulty of the activity.

4.6.2 Difficulty in Construction

In the other activities presented in this project, the students were designing against a highly constrained physical world (the Light Optimization activity), a simulated, abstract environment (the Elevator Control activity), or used a symbolic world (the "Rush Hour" and Word Search activities). In the Gear Reduction activity, the student design was tested directly against nature, which provided many unanticipated

	A/B	C/D	E/F
Iteration Count	7	8	0
Time per iteration	5 min	3 min	
St.Dev. of time per iteration	4.8 min	2.1 min	
Non-iterating time	15%	42%	100%
Success (0,1,2)	1	2	0

Table 4.2: Results from the Gear Reduction activity. The best performing team had the longest non-iterating time, with the exception of group E/F, who never advanced to testing.

complications. The challenge was not focused on gear ratios as expected, but on the more general and difficult task of building. Students were plagued with an under constrained design space including part selection, structure design, gear choice, and component connections. Despite the unexpected difficulty, most students (two of the three dyads) did succeed in solving the design problem. However, the process-oriented model that was desirable was lost as students focused entirely on finding a single working solution.

4.6.3 Gear Reduction Results

The iterations performed by the students in this activity are shown on a single timeline in Figure 4-3. The two groups shown had very different iteration patterns. Group A/B is bimodal, with many iterations in the beginning and end segments of the activity. Group C/D started iterating later in the session, and was more consistent with iteration spacing. Group E/F did not perform any iterations. All of these observations are analyzed in detail below.

This activity was included for numerical analysis. The results are shown in Table 4.2. Dyad E/F never advanced to performing a test, and as such is listed with zero iterations and a non-iterating time of 100%. This group did not successfully complete the activity. Excluding that pair, the most successful student group, C/D, had the longest introductory time at 42%. Group A/B had nearly the same number of iterations as C/D, but they were more spread out over the session time. C/D

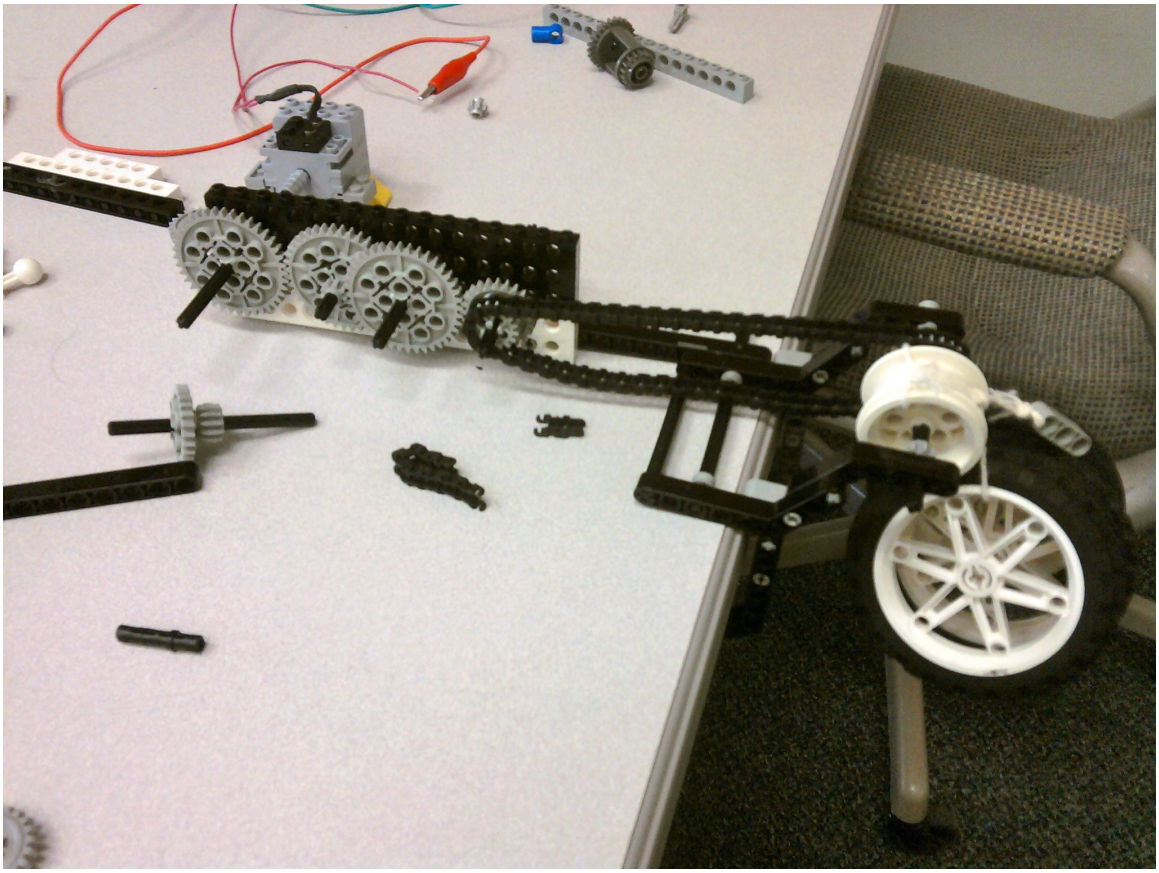


Figure 4-4: Gear Reduction solution made by C/D dyad. This design successfully completed the task. This image includes the mass that was lifted, which are the large wheels to the right.

iterated faster, with more consistent time per iteration. Both A/B and C/D achieved successful solutions, but only C/D achieved a generalized process.

Building with LEGO is difficult. This activity was more difficult than expected and did not carry the intended focus of pure gear mechanics. The groups who managed to perform tests and iterations did overcome these problems, and the group that did not test did not achieve any levels of success.

The solution built by group C/D scored the highest success rating, and is shown in Figure 4-4. The diagram they drew is seen in Figure 4-5. This group had one stage that actually performed a reduction of 8:40. The design included an unnecessary 1:1 stage and a counter-productive 40:24 gear increase. The overall reduction was sufficient to complete the task, and the construction was solid.

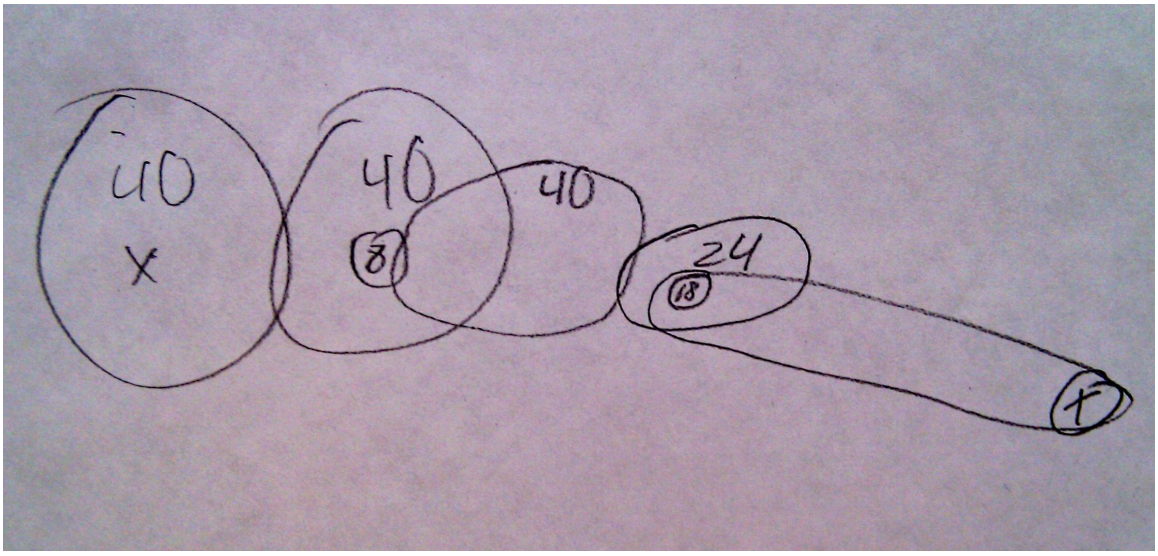


Figure 4-5: Gear Reduction Solution diagram by C/D dyad.

The solution built by group A/B is shown in Figures 4-6 and 4-7, with their diagram shown in Figure 4-8. The design was theoretically sound, employing three stages of 24:40 reduction. The construction was weak, but sufficient to survive the one-day activity.

The group that did not complete, group E/F, provided a diagram that is shown in Figure 4-9. This diagram lacks detail of the gear interactions. It does not specify any gear sizes, nor does it indicate how the gears actually connect together. The diagram is an abstract representation of a the general shape of a solution, but does not specify any actual implementation.

4.7 Word Search Activity

This activity was designed to be easy for the students to engage with, as most of them were very familiar with word searches. It also provided a good introduction to algorithms, where students could easily see the application of a mechanical process on the word search grid.

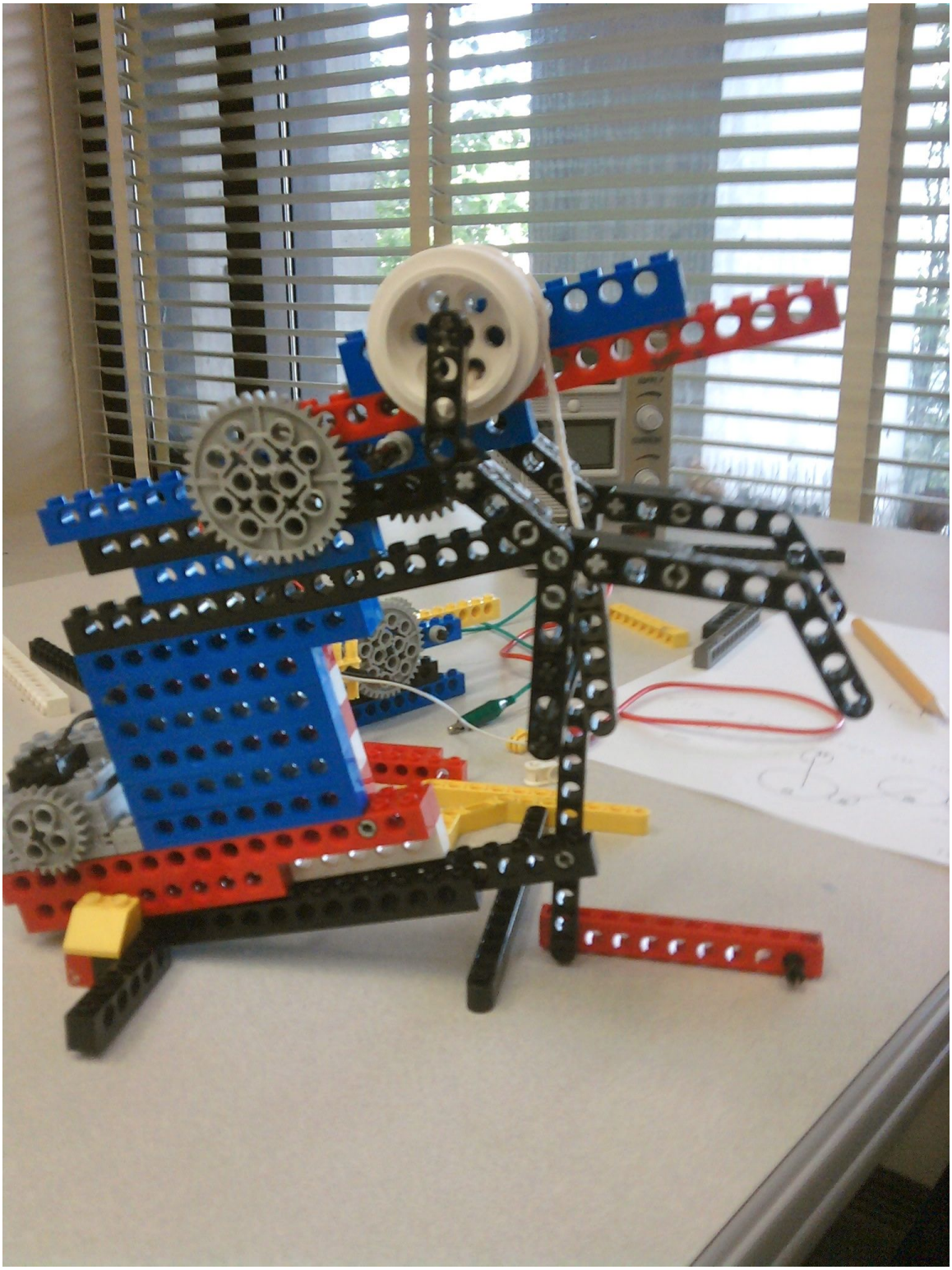


Figure 4-6: Gear Reduction solution made by A/B dyad, front view. This design was successful, but not as solid as the one made by C/D. A length of chain is called for to connect the motor at the far left to the large gear in the center (not shown).

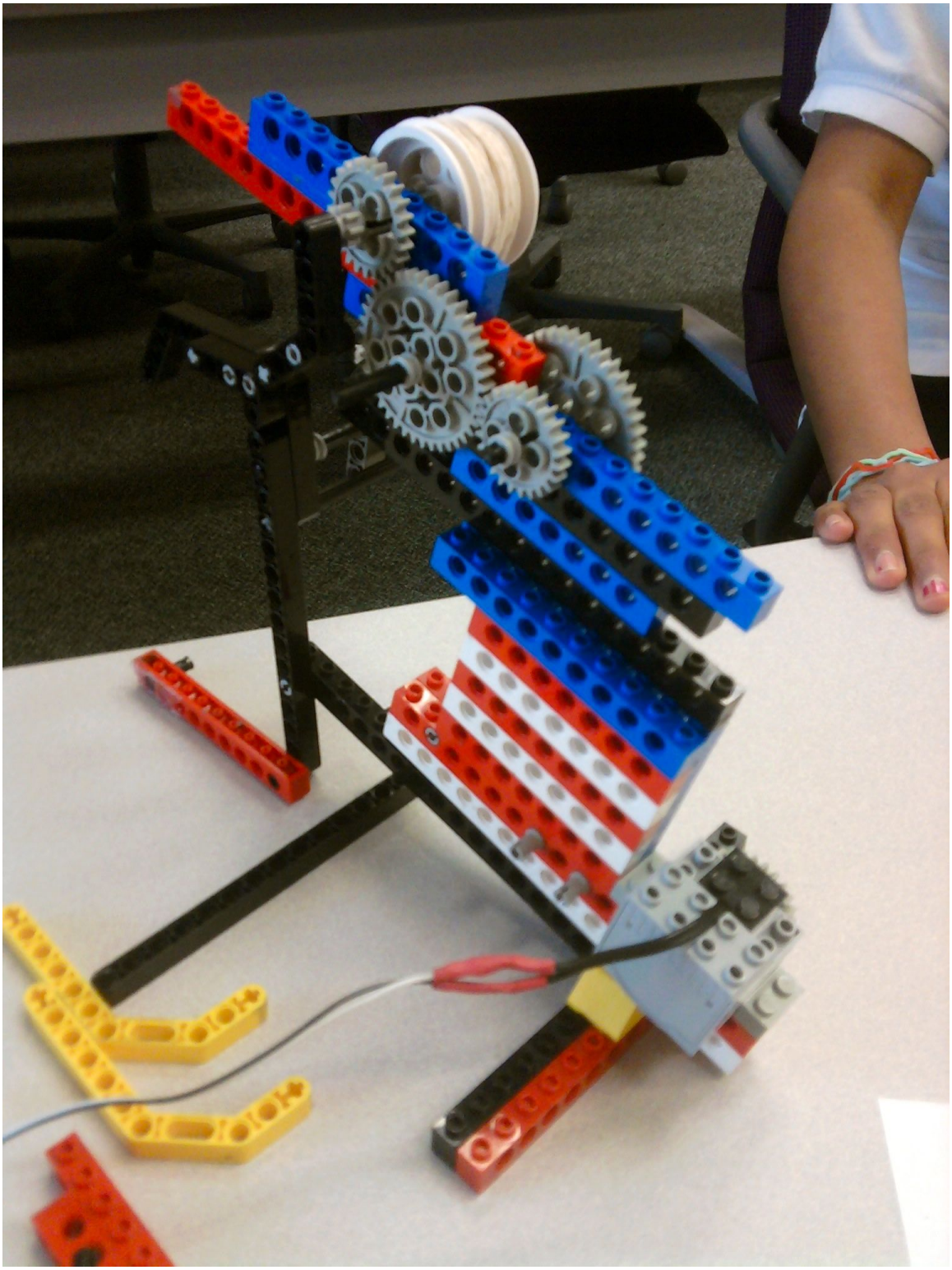


Figure 4-7: Gear Reduction solution made by A/B dyad, rear view.

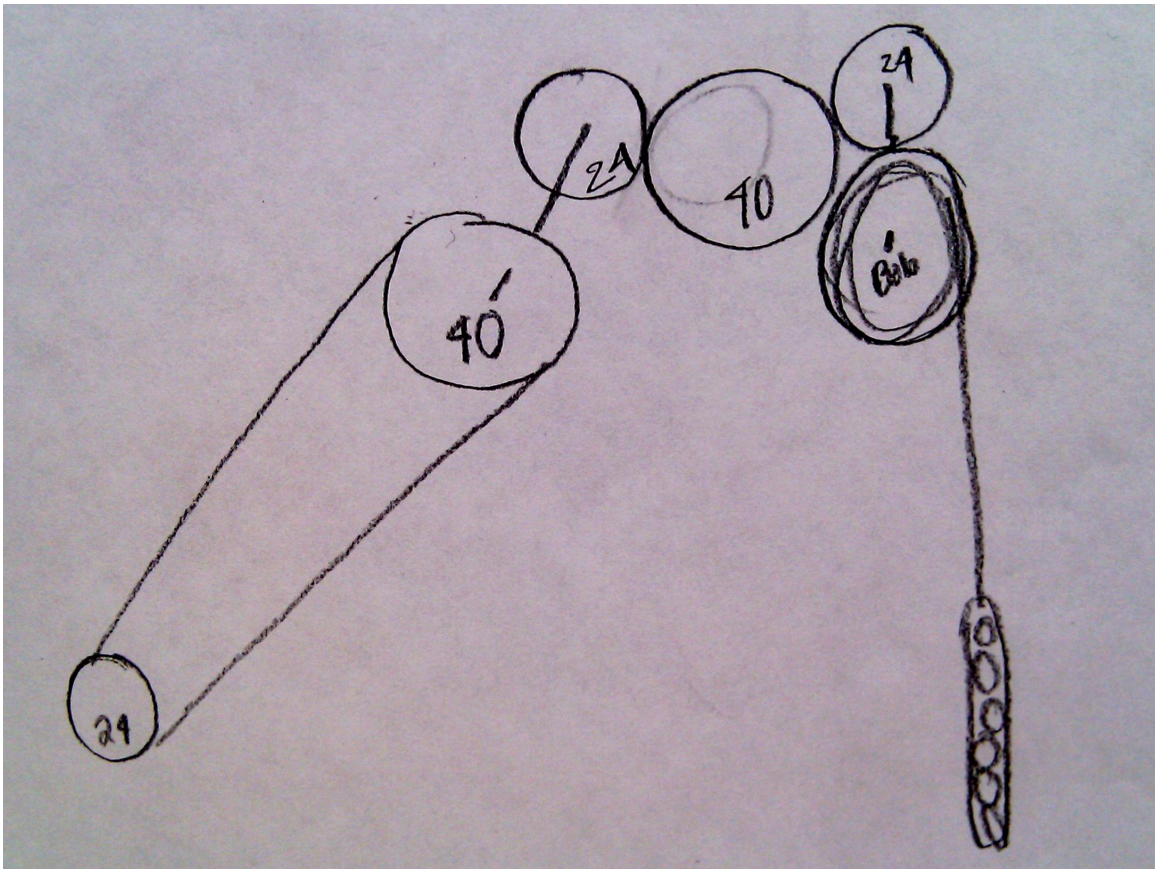


Figure 4-8: Gear Reduction solution diagram by A/B dyad.

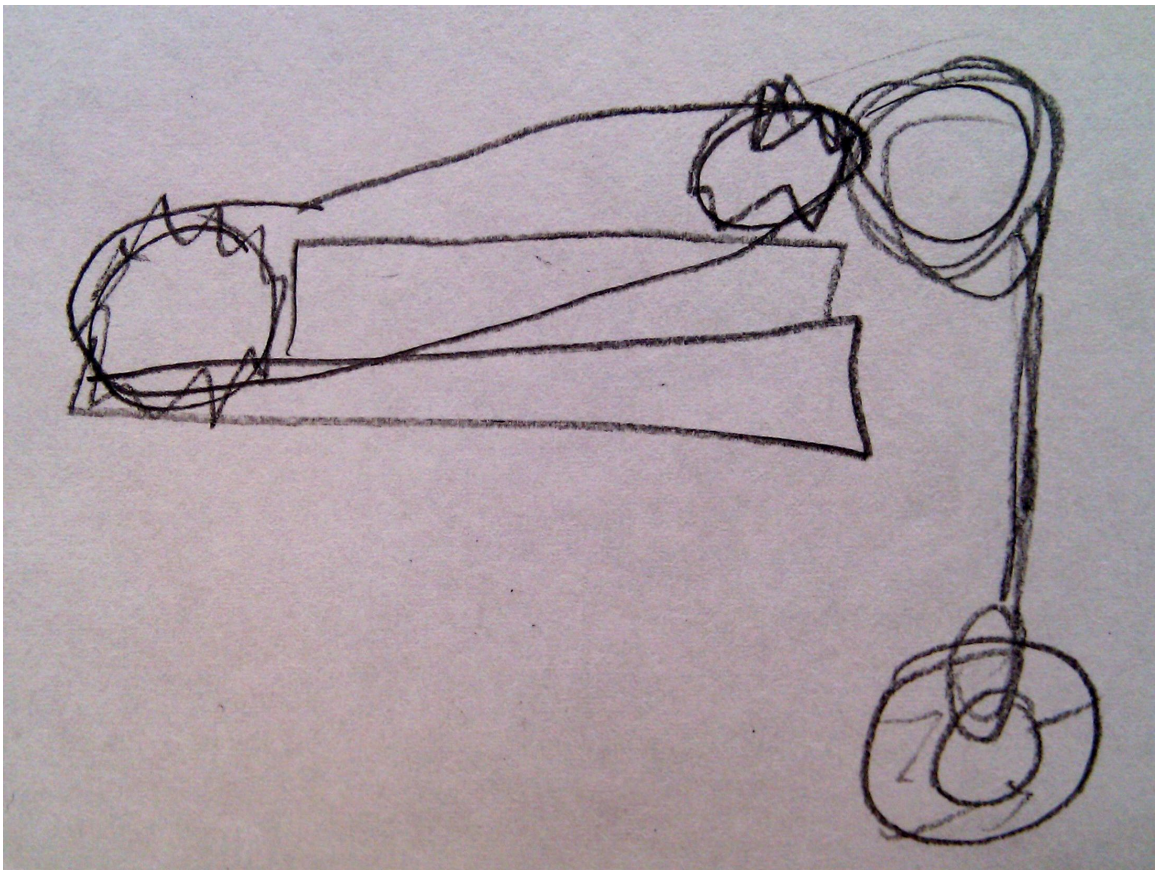


Figure 4-9: Gear Reduction solution diagram by E/F dyad. This group never tested their design, nor came to a working solution. The diagram lacks detail of the system.

4.7.1 Expected and Observed Outcomes

The word search was expected to be easy for the students to gain traction with, enabling them to focus on the algorithm design component. The students were expected to have a few test iterations as they worked through how algorithms function. The activity itself was relatively simple; it was so simple that many students lost interest before the end of the session. In practice though, this session was successful. Students maintained engagement for the majority of the period and held to the rules of the activity to facilitate the desired learning experience. Half of the students actually created legitimate algorithms; the other half created short lists of human-targeted instructions, and did not break them out to component operations.

The algorithms the students wrote were classified into one of two categories based on the style of instructions that were used: high-level and low-level. High-level instructions required intelligent interpretation, such as "look for any letters in the word." Low-level instructions were more basic, and could feasibly be implemented in a machine language, such as "scan this row for letter x." The students were not instructed in this difference; the categories were only for purposes of analysis.

4.7.2 Word Search Results

Students became aware of the concept of an algorithm and understood that they were designing them. This is indicative that students of this age possess the cognitive development to think abstractly at the level that is required for algorithm design.

Despite the students' understanding of the problem and high level of engagement, the iteration count was low. Most students only performed two significant changes on their algorithms, and these occurred only as artifacts of the session instructions. Most students made these iterations at about 20 and 30 minutes into the activity. One student made one additional iteration at about 40 minutes.

Analysis of this activity is based on the type of instructions the students wrote. High-order instructions were clearly intended to be interpreted by humans, as they required complex reasoning to be executed, such as "wait for words to pop out."

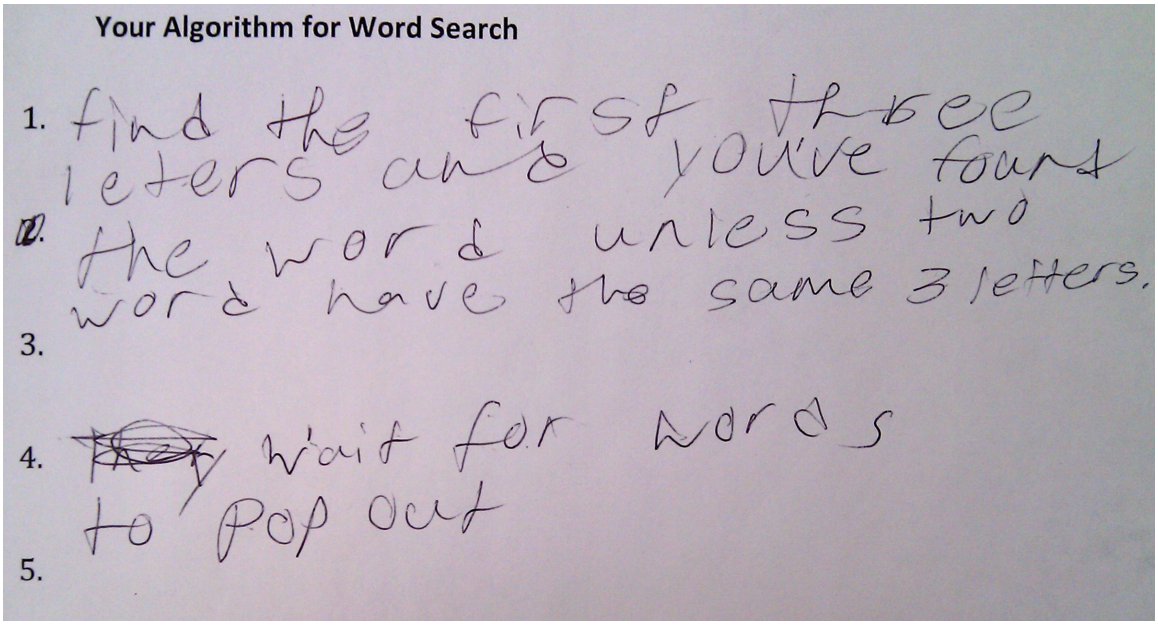


Figure 4-10: Example of a student's algorithm for the Word Search activity. This algorithm uses high-order instructions intended for a human.

The minority of students used simpler instructions that could be argued as machine-compatible. Simpler instructions included "look for the first letter" and "see if second letter is touching it." These simple instructions could be combined to create the behaviors that the majority of students represented in a single statement.

An example of high-level instructions can be seen in Figure 4-10. That student wrote an instruction to find the first three letters of the word at once. The statement was very compact, including a built-in exception: "unless two words have the same 3 letters." This phrasing requires a complex interpretation in order to execute.

An example of simpler instructions can be seen in Figure 4-11. This student started off in step one with a strategy of "If you can't find it the first time, try again." This first instruction could be argued to be specifying a loop, which, according to the student's explanation, was the basic intention. Step two is precise, instructing the executor to find the first letter of a word, and then see if the second letter is adjacent. If the second letter is adjacent, check to see if the whole world is there. This step could be broken into to four individual instructions.

The students who wrote low-level instructions averaged one additional iteration

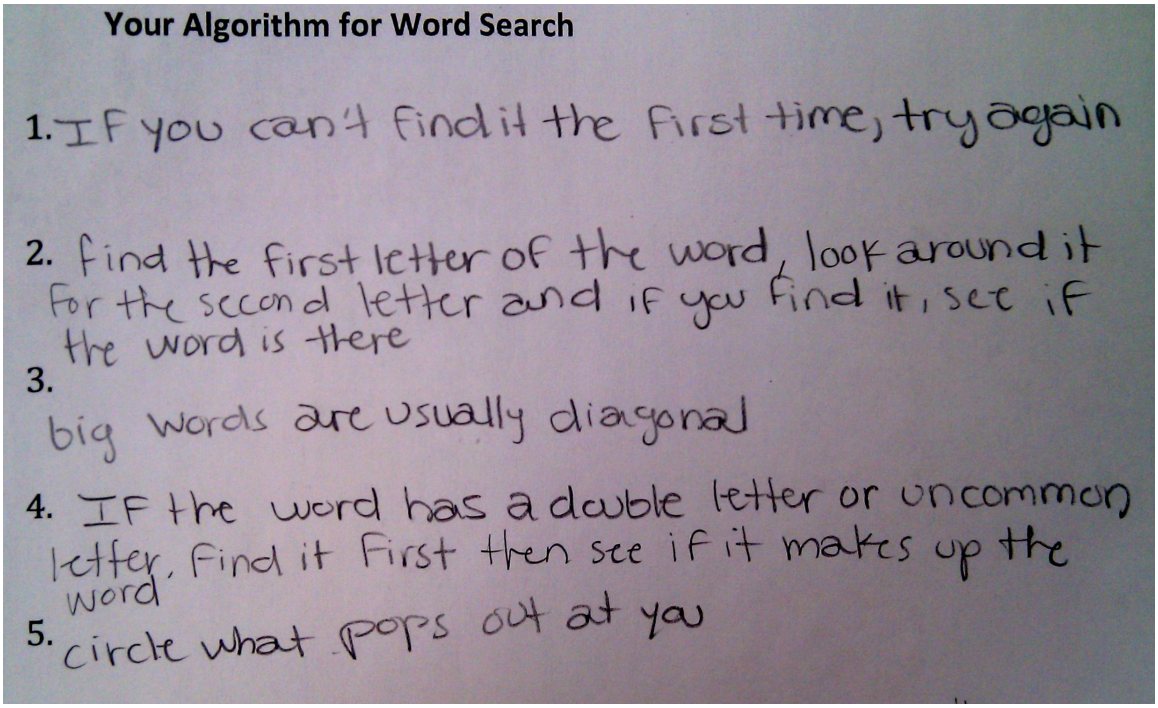


Figure 4-11: Example of a student's algorithm for the Word Search activity. This algorithm uses simple, generalizable instructions.

than those who wrote high-level instructions. This could be indicative to additional complexity being required to write a longer series of simpler instructions.

4.8 Elevator Control Activity

Building on the algorithm writing experience from the word search activity, the elevator control problem is the culmination of the activity sequence. Students built and tested in a simulated environment where the researchers had control of all system mechanics, allowing for only the pertinent interactions to be available. This still allowed for a complex interaction to take place, as the students were were asked to write code to solve an open-ended problem.

4.8.1 Expected and Observed Outcomes

This was the most difficult activity, so it was expected that students would have a hard time gaining traction with the problem. Before a student could be productive,

the student would need to understand the constructs of the programming language, begin to comprehend the programming concepts available to them, and form a good understanding of the problem. The most difficult part of this was the latter. The first half of this activity ended up being focused primarily on problem framing. Once students began to comprehend what they were actually trying to do, their productivity improved greatly. Student competence with the programming tools seemed to increase once the problem was understood, indicating that understanding the tool is in part related to having a clear need for it.

4.8.2 Problem Framing

The greatest difficulty in this activity was understanding the task at hand. The setup provided to the students included example code that worked as a fully functional system. The students first questioned what their task was if a working solution was already present. The task was to create an better optimized solution, but what made the example sub-optimal was not apparent.

The example code would go to every floor that was requested in the order that the requests were made. This solution works fine for simple examples where the request queue has a small number of requests, or floors are requested in a convenient order. There are many common examples, however, that illustrate the sub-optimal nature of this system. One such example is if the elevator is on the first floor, and the buttons five, three, and four are quickly pressed, in that order. At the first button press, the elevator will immediately start moving towards the fifth floor. Assume the button presses were complete by the time the elevator reaches the second floor. It now has requests to service the third, fourth, and fifth floors, and is already moving upwards. The example code will skip the people waiting on the third and fourth floors and go straight to the fifth floor. At that point it will skip the fourth floor again as it moves to the third floor, as that was the order that the buttons were pressed. If a person entering on the fifth floor then made a request, that request would not even be considered until all the previously queued pickups had completed.

The above example and ones like it were used by the researchers to help illustrate

Figure 4-12: Elevator Control activity iteration timeline. Each mark represents a test occurring for the specified students. Each row represents a different student group. The horizontal axis is time, in minutes. All students considered themselves done and stopped working after their final test.

what was wrong with the example code. The researchers did not explicitly state what was wrong, but guided the students in running such examples and led them to reach their conclusions about inefficiency. Once this inefficiency was understood the students were observed becoming more effective in creating a solution.

4.8.3 Elevator Control Results

Student iteration data is shown in Figure 4-12. The three groups exhibited different amounts of introductory time in this activity, ranging from a short three minutes (group A/B) to nearly half the session time at 24 minutes (student D). Each student stopped working after their group's final test, concluding that they had completed the exercise. Student C reached this point first, followed by student D and dyad A/B respectively.

This activity was included for numerical analysis. The results are shown in Table 4.3. In this session, success was correlated both with tight iteration and longer introductory time. The two groups who succeeded in both a working solution and generalized process spent at least a third of their total time exploring the problem, and then iterated quickly at regular intervals. The most pronounced correlation is the short introductory time of dyad A/B, who were the only group not to successfully complete the activity. Only one of the students had any previous experience with the Scratch system. Interestingly, that one student was in the group that did not succeed in this activity.

	A/B	C	D
Iteration Count	9	6	5
Time per iteration	5.3 min	3.6 min	3.4 min
St.Dev. of time per iteration	4.4 min	2.9 min	1.4 min
Non-iterating time	7%	36%	64%
Success (0,1,2)	0	2	2

Table 4.3: Results from Elevator Control activity. The students of dyad C/D chose to work separately in this session. There are no data for dyad E/F.

The use of a simulator in Scratch was largely successful. All but one of the students had no prior experience yet were able to piece together functional control programs. Beyond that, students were able, to varying degrees, create programs that did as they intended. In one case a student's goal was modified as the student learned about the capabilities and limitations of the system.

Chapter 5

Discussion

One behavior that emerged as a critical component to design is iteration. The literature supports that iterative design cycles are critical for design tasks across academics, children, and industry. Atman et al. (1999) showed that senior college-level design students iterated far more than their underclassmen counterparts. Eckert et al. (2009) reasoned that iteration is critical in industry, and should be practiced more than it is. Dow et al. (2009) provides the most compelling conclusion, stating that iteration helps designers achieve as well as domain knowledge could afford them without iteration.

In this study, the iterative nature of the different students was accessible for investigation. By analyzing the schedule by which students test and modify their solutions, patterns were identified with design success and characteristics of the activities themselves.

Chapter 4 illustrated the iteration patterns for each individual activity. Where each session presented a different type of design task, comparing activities by iteration count is not valid. The percentage of time spent in introduction is scaled based on the total time spent on the activity, making this metric portable between activities. The introduction time, or non-iterating time, characterizes what percent of the student's total time during the session was spent before they performed their first test. It is posited that this time was used for learning about the problem and performing initial construction. The average times for the three numerically analyzed activities are given in Table 5.1.

	Light Opt.	Gear Reduction	Elevator Control
Average Non-Iterating time	25.5%	28.6%	35.8%
Standard Deviation	7.0	13.0	13.5

Table 5.1: Average non-iterating times for each activity. These data do not include instances where students did not complete an activity.

Information can be gained about the overall performance of the students. Table 5.2 shows the three student dyads with their overall success rating, non-iteration time, and time per iteration. The overall success rating comes from the combined success rating of the three activities that were numerically analyzed: Light Optimization, Gear Reduction, and Elevator Control. As previously discussed in Section 3.3, each activity had two levels of success. Achieving the first level represents the completion of a successful design that solves the problem. Achieving the second level represents synthesizing a general process for arriving at a solution. Students that obtained the first level scored a single rating point for the activity. Those who achieved the second level obtained an additional point, totaling two for the activity. The maximum rating for three activities was six points, shown as the denominator for dyads A/B and C/D. The third pair only participated in two of the activities, so that group's maximum rating was four.

The average non-iterating time for each group is presented in Table 5.2, where the percentage shown is the amount of time that elapsed before the first tests. The standard deviation of this metric is included as well. Based on that information, dyad C/D not only spent more time on average in pre-iteration, but did so with greater consistency than group A/B. The number present for group E/F only includes the Light Optimization activity, as it is the only one they completed. Average time per iteration includes all iteration cycles made across all three activities. The most successful dyad shows not only the shortest time per iteration, but the most consistency in those times.

	A/B	C/D	E/F
Overall Success Rating	3/6	5/6	1/4
Average Non-Iterating time	14.4%	37.9%	incomplete/4.9%
StdDev Non-Iterating time	*6.0*	*2.7*	
Average time per iteration	7.8 min	3.8 min	incomplete/6 min
StdDev Av time per iteration	*7.8*	*3.1*	

Table 5.2: Success correlations with non-iterating time and average time per iteration based on three activities. Group E/F participated in, but did not complete, the Gear Reduction activity and did not participate in the Elevator Control activity. That group's listings for non-iterating time and time per iteration considers only the one activity they completed: Light Optimization.

5.1 Conclusions

The data presented in Table 5.2 shows that the most successful students were consistently slower to begin testing across all activities, indicating that they spent more time in each activity trying to understand the problem before attempting a solution. The speed of iteration also correlates with success, but not as strongly as the introductory time spent.

Across the activities, there was also an overall trend in student performance data. Non-iterating time, or introductory time, was shown above to be the strongest indicator of student success in this study. Analyzing that metric for the activities across all students showed additional trends. Table 5.1 indicates a monotonic, positive correlation between difficulty of the activity and the exploratory time spent by all the students. The more difficult an activity was, the longer it took students to begin testing their solutions.

The strong trends of exploratory time for both individual students and specific activities indicates that this incubation period is a critical component of design and has an effect on how well the students performs their design iterations later in the session.

Following, the research questions that formed the foundation of this study are revisited and directly addressed.

Do students exhibit patterns in testing and iteration? What are those patterns?

Students showed a number of patterns in their testing habits during design. The patterns appeared to be based on a few variables. The first variable, as shown in the Light Optimization activity, was relevant prior knowledge. If a student already had a strong content knowledge from a previous experience, then that student will not require an extensive exploration phase, and will not find a need to iterate deeply. The second variable was the complexity of the problem. The more difficult the problem was, the more time students took in exploring it before they began testing cycles. This is indicated in Table 5.1. The more complex activities also had a larger standard deviation of preparation time, indicating that the different students took largely different amounts of time in preparation.

What characteristics of a design activity elicit specific iteration patterns?

As discussed above, complexity of the design activity had an observable effect on student iteration. In addition to that, the cognitive overhead involved in constructing a theory or prototype and testing it had a significant impact. The Rush Hour game required very little overhead to play, both cognitively and physically, and students generated ideas and executed tests at rates far beyond those of the other activities. The degree of abstraction, from the purely tactile Gear Reduction activity to the purely symbolic Elevator Control activity, had no observable effect on iteration rates. Total problem difficulty appeared to be more significant than the use of symbolism towards the amount of preparation time students used.

In classrooms, activities are often designed to include these explicit design phases to guarantee some level of iteration, but the data of this study are inconclusive towards this practice. Students in this study created tests and made incremental improvements on their solutions in all activities, regardless of whether an iteration schedule was provided to them. The activities that included explicit design phases showed no difference in session-long iteration patterns than the activities with no set schedule.

What is the correlation between iteration in designing and success of the design?

The more successful students correlated with greater introductory times and smaller deviations of iteration times. The model of the most successful student had a long period (nearly 60% of total time spent) exploring the problem, and then proceeded to perform six to eight quick iterations in the remaining time.

In the Gear Reduction activity students were faced with a difficult construction task that was confounded by many physical factors. The students who conducted tests and iterations managed to overcome these difficulties, while the students who did not test never managed to understand the problem sufficiently to create a working solution. This lack of understanding is visible in the students' diagram of their solution, shown in Figure 4-9. The diagram does not demonstrate any conceptual understanding of the meshing of gears or the construction of a supportive structure. The abstraction used in this diagram is inconsistent, showing basics of gears, chains, structure, and the load. In contrast, Figure 4-8 shows a diagram that indicates a high level of understanding of the problem, and has abstracted it efficiently to only represent the operation of the gears.

More can be learned from the Gear Reduction activity about testing complex systems. The most successful group had a long exploration period and quick iterations, but also managed to construct a solution faster than the other groups. One significant difference between that group and others was the use of component testing. The successful group tested individual parts of the solution one at a time, and built upon them as they were shown to work. The presence of component tests also correlated with successful designs in the Elevator Control activity. Complex systems required component-level testing for efficient development, but students may not intuitively do this.

What guidelines can be written for the creation of future activities?

This research question as addressed in its own section, which follows immediately.

5.2 Recommendations

Recommendations that bear on the challenge of creating design-based engineering activities can be made from this study.

Give sufficient introduction time to the problem

The greatest indicator of student success in both creating a working solution and a generalizable process was the time the student spent before starting testing and iteration. This time is presumably being used to explore the problem. By designing the structure of the activity to encourage students to explore, the students are more likely to gain the necessary traction to accurately manipulate the problem later in the activity.

One iteration is better than none, but more is even better

The literature strongly supports that the presence of an iterative process, even if it is a single revision, results in better solutions being generated. This study supports that finding. All students were observed achieving success in three to six iterations, but the number of revisions for a successful design depended on the complexity of the activity and the designing style of the student.

Put a desired skill in the critical path

The Light Optimization activity showed students using written algebra of their own volition in order to help their design process. The activity used a calculated score to rate the success of student designs, and that formula was given to the students at the beginning of the session. The students quickly realized that they could use that formula to help guide them through their design. In this case, the students were never prompted to perform any written algebra, but did so as it was understood to be a critical tool to achieving their goals. By designing the activity such that a desired skill (such as algebra) is in the critical path from the student to their goal, the students may have self-provided desire to use that skill.

Clearly frame the problem

The importance of the student's understanding of the task to be completed should not be underestimated. When the students clearly understood the task, such as in the Light Optimization activity, they worked efficiently and effectively. When the goal was unclear, such as in the Elevator Control activity, they became sidetracked and confused. Once the true task was understood, the students demonstrated greater competence in using resources to solve it.

Use a microworld

Design activities should not expose the student to degrees of complexity beyond their immediate learning needs. A microworld is valuable in providing a local context in which the problem can be constrained. The term is generally used for software simulations, where the only relationships involved are explicitly put there by the simulation's creator, but it can also be applied to physical activities. The complex building skills necessary for the Gear Reduction activity could be reduced by providing a "gear wall" where gears can be placed onto pre-made pegs and holes. This approach would abstract away the construction element, and provide pre-calculated axle distances to ensure proper gear meshing. The student would not have to worry about those factors in the design.

5.3 Future Work

New methods for characterizing and analyzing design behaviors, specifically testing and iteration, were created in this project. These methods can be used in future work to characterize additional design behaviors. The analytical techniques presented here can be used to further study the incubation or exploration period of problem solving, student self-assessment of design, and general cognitive strategies for engineering. There is not yet a definitive list of behaviors or strategies that make up design. Additional elements of design methodology may be defined in the future, and the methods developed here can be used to analyze them.

This work examined how students naturally employ iteration, one of the critical elements of the design process. This study did not investigate why the student chose to conduct a test, only when. To gain further insight into how novices solve design problems, the driving factors behind those iterations need to be identified and explored. One likely factor is a self-assessment mechanism employed by the students to know when and how to conduct a test. Student assessment of their solution and their process during the activity can be tested using methods similar to those of this study, and is yet unexplored.

Exploring the tacit motivations behind design iteration and process will yield a deeper understanding of why design processes fit the models previously mentioned. Future work will result in a more complete set of behavior patterns that contribute to a good design process. Teaching these skills explicitly to students will aid them in assessing their own design tendencies, and help them become better designers. Also, from these patterns, a tool could be created that may help in the design of student design activities, providing a guide for creating activities that elicit specific iteration behaviors from students.

References

R. S. Adams, J. Turns, and C. J. Atman. "Educating effective engineering designers: the role of reflective practice." *Design Studies*, 24:275–294, 2003.

C. J. Atman, J. R. Chimka, K. M. Bursic, and H. L. Nachtmann. "A comparison of freshman and senior engineering design processes." *Design Studies*, 20:131–152, 1999.

B. K. Beyer. *Developing a Thinking Skills Program*, chapter Defining Thinking and Thinking Skills. Allyn and Bacon, Inc, 1988.

B. S. Bloom and L. J. Broder. *Problem-solving processes of college students: an exploratory investigation.* University of Chicago Press, 1950.

A. L. Brown. "Design Experiments: Theoretical and Methodological Challenges in Creating Complex Interventions in Classroom Settings." *The Journal of the Learning Sciences*, 2(2):141–178, 1992.

C. A. Chinn and A. Samarapungavan. "Distinguishing Between Understanding and Belief." *Theory Into Practice*, 40(4), Autumn 2001.

S. P. Dow, K. Heddleston, and S. R. Klemmer. "The Efficacy of Prototyping Under Time Constraints." In *C&C'09*, Berkeley, California, USA, October 2009. ACM.

C. Eckert, D. Wyatt, and P. J. Clarkson. "The Elusive Act of Synthesis: Creativity in the Conceptual Design of Complex Engineering Products." In *C&C'09*, Berkeley, California, USA, October 2009. ACM.

K. A. Ericsson and H. A. Simon. *Protocol Analysis. Verbal reports as data*. MIT Press, Cambridge, MA, 1984.

B. Harvey and J. Mönig. "Bringing "No Ceiling" to Scratch: Can One Language Serve Kids and Computer Scientists?." In *Constructionism*, Paris, France, August 2010. American University of Paris.

C. E. Hmelo, D. L. Holton, and J. L. Kolodner. "Designing to Learn About Complex Systems." *The Journal of the Learning Sciences*, 9(3):247–298, 2000.

R. A. Kimbell and K. Stables. *Researching Design Learning*, volume 34 of *Science, Technology, Education Library*. Springer, Heidelberg Germany, 2007.

R. Lesh and G. Harel. "Problem Solving, Modeling, and Local Conceptual Development." *Mathematical Thinking and Learning*, 5(2-3):157–189, 2003.

J. Levin and M. Waugh. "Educational simulations, tools, games, and microworlds: Computer-based environments for learning.." *International Journal of Educational Research*, 12(1):71–79, 1987.

J. Lochhead and A. Whimbey. *Teaching Analytical Reasoning Through Thinking Aloud Pair Problem Solving*, chapter 5. Jossey-Bass, 1987.

J. Maloney, L. Burd, Y. Kafai, N. Rusk, B. Silverman, and M. Resnick. "Scratch: A Sneak Preview." In *C5 '04: Proceedings of the Second International Conference on Creating, Connecting and Collaborating through Computing*, pages 104–109, Washington, DC, USA, 2004. IEEE Computer Society.

F. G. Martin. "The Art of LEGO Design." *The Journal for Robot Builders*, 1(2), March 1995.

J. Piaget and B. Inhelder. *The Psychology of the Child*. Basic Books, 1969.

W. A. Sandoval and P. Bell. "Design-Based Research Methods for Studying Learning in Context." *Educational Psychologist*, 49(4):199–201, 2004.

D. Schön. *The Reflective Practicioner: How Professionals Think in Action.* Basic Books, New York, 1983.

M. Scribner-MacLean. "Exploration of Design Thinking Patterns in Informal and Formal Settings." NSF REESE Proposal, December 2009.

A. Strauss and J. Corbin, editors. *Grounded Theory in Practice.* Sage, 1997.

N. P. Suh. "Axiomatic Design Theory for Systems." *Research in Engineering Design*, 10(4):189–209, December 1998.

The Boston Museum of Science. "Engineering Design Cycle." part of Design Challenges exhibit, 2008.

M. Welch. "Analyzing the Tacit Strategies of Novice Designers." *Research in Science & Technological Education*, 17(1):19–34, 1999.

C. Wilson and M. Guzdial. "How to Make Progress in Computing Education." *Communications of the ACM*, 53(3):35–37, May 2010.

J. M. Wing. "Computational Thinking." *Communications of the ACM*, 49(3):33–35, March 2006.

Appendix A

Student Handouts

Paper handouts were used in three of the activities to provide instructions and work space for the students. They are included here.

Figures A-1 and A-2 show the handout that was provided to students when doing the Gear Reduction activity. At the conclusion of the session the students were asked to draw their design in the style of the example in Figure A-2.

Figures A-3 through A-5 show the handout that was provided to students when during the Word Search activity.

Figure A-6 is one of the two puzzles given to the students. Solutions have been highlighted.

Figure A-7 shows the instruction sheet that was given to students at the beginning of the Elevator Control Activity.

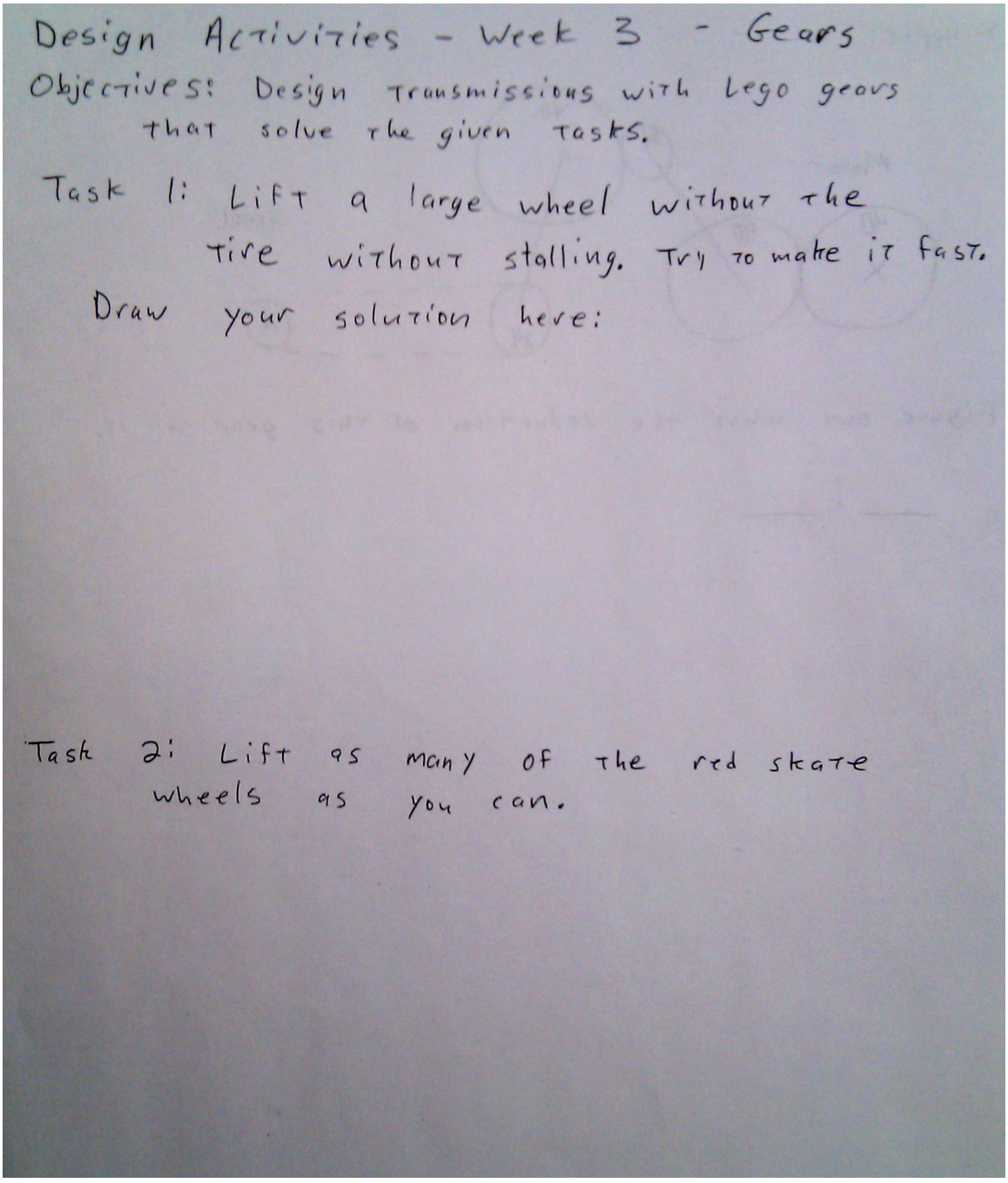


Figure A-1: First page of the worksheet given to students during the Gear Reduction activity.

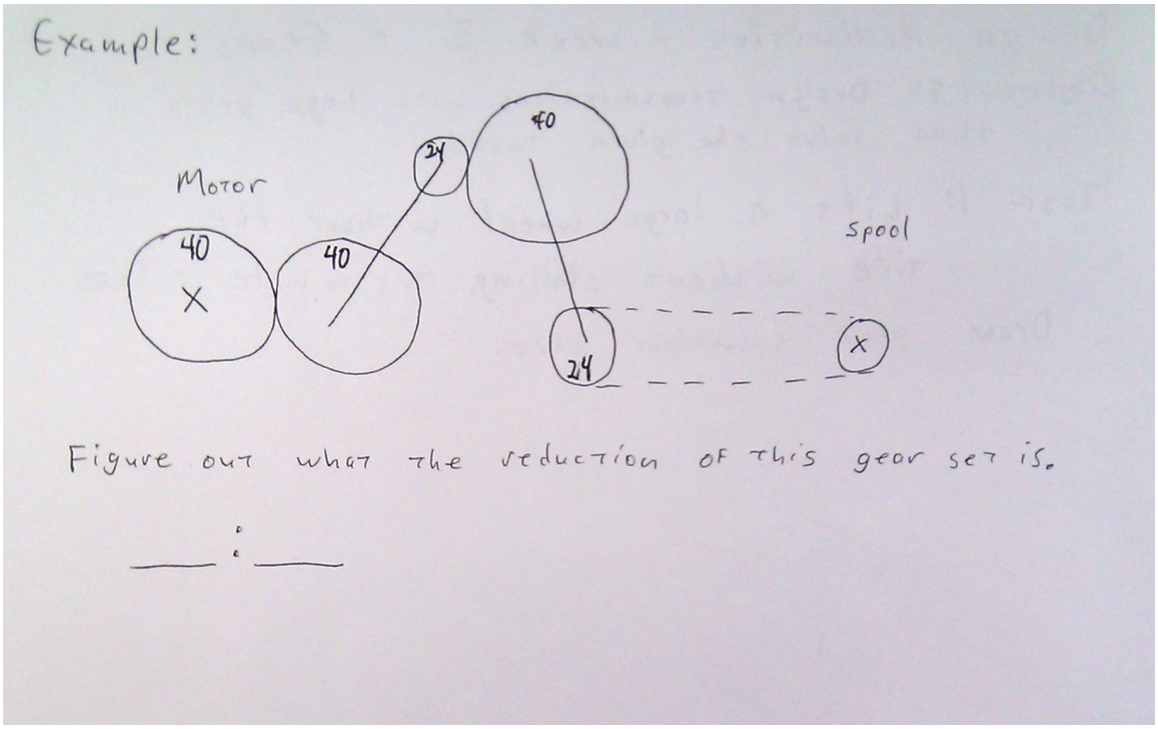


Figure A-2: Second page of the worksheet given to students during the Gear Reduction activity.

Design Strategies Week 4: Word Searches
Presented by Prof. Fred Martin and Mark Sherman

Your task:

Develop a process to find all the straight-line words that are hidden in a grid.

Example: Tic Tac Toe

Solving tic-tac-toe is pretty easy. There is a simple series of rules, which if you follow, will guarantee you won't lose. You can think of these rules as a *strategy*, but when you follow them all strictly is can be considered an *algorithm.* An algorithm is just a set of instructions that make up a process.

Instruction Set for Tic Tac Toe

1. If you have two in a row, and the third is empty, take the empty to make 3.
2. If the opponent has two in a row, and the third is empty, take the empty to block.
3. If a fork can be created, do it. *(In the figure to the right X has created a fork, where O needs to block in two places at once.)*

4. If the opponent is about to make a fork, block the fork.
5. If the center is open, take it.
6. If a corner is open, take it.
7. If a side is open, take it.

Algorithm

A process or set of rules to be followed in calculations or other problem-solving operations.

Figure A-3: First page of the worksheet given to the students during the Word Search activity. This worksheet explains algorithms through the explanation of tic-tac-toe.

Design Strategies Week 4: Word Searches
Presented by Prof. Fred Martin and Mark Sherman

The Word Search

Y	E	S	L	F
M	N	H	R	I
P	T	O	U	R
A	C	E	M	E

Word list:

- Ace
- Fire
- Shoe
- Tour
- Yes

Did you find all the words?

What did you do to find them all?

Did the words just "jump out at you?" .... What if they don't? How can you know for sure that you found *all* the words?

Your Algorithm

Write down any rules or strategies you used.

1.

2.

3.

4.

5.

Figure A-4: Second page of the worksheet given to the students during the Word Search activity. This sheet guided the student through the creation of an example algorithm.

Design Strategies Week 4: Word Searches
Presented by Prof. Fred Martin and Mark Sherman

Your Algorithm for Word Search

1.

2.

3.

4.

5.

6.

7.

8.

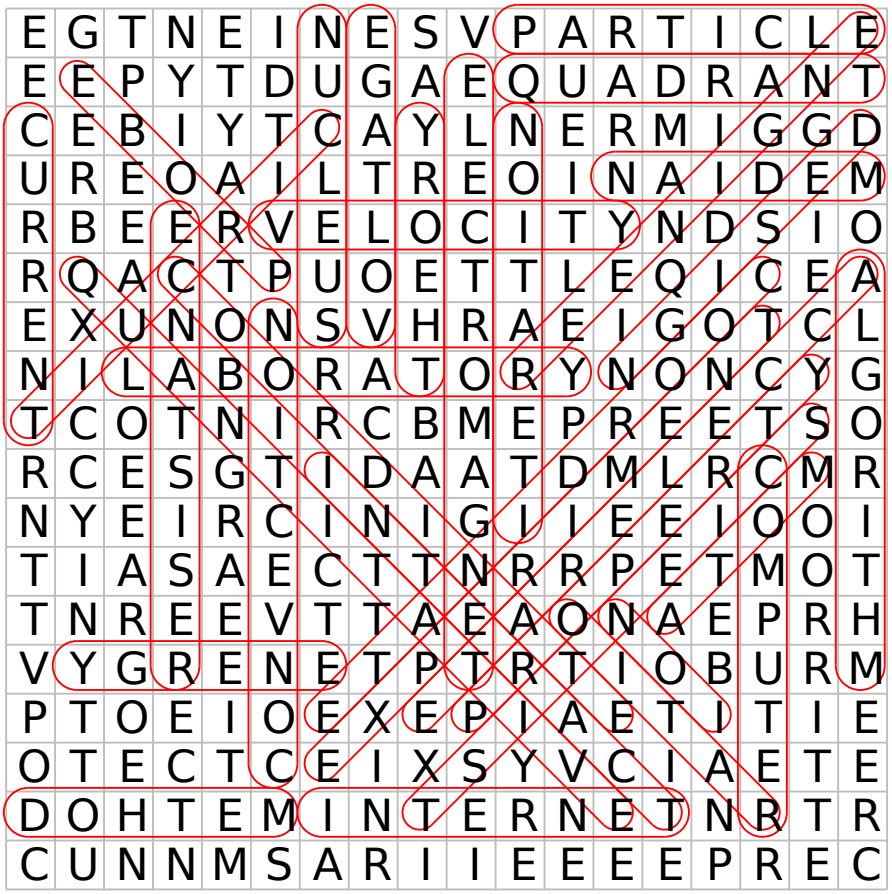
9.

10.

If you need more rules, grab some more paper and add your own numbers. You can have as many as you want.

Figure A-5: Third page of the worksheet given to the students during the Word Search activity. This sheet provided space for students to design their algorithm.

Harder Puzzle 1
Mark Sherman



accelerate
algorithm
atom
circuit
computer
convection
coordinate
coordinate
current
design
electromagnet
energy
engineer
experiment
interact
internet
ion
iteration
laboratory
median
method
nucleus
particle
probe
property
quadrant
quantitative
ratio
resistance
scientist
Theory
Velocity
Voltage

Figure A-6: Word Search puzzle provided to the students. The hidden words are shown.

Design Strategies Week 5: Elevator Control Algorithm
Presented by Prof. Fred Martin and Mark Sherman

Your task:

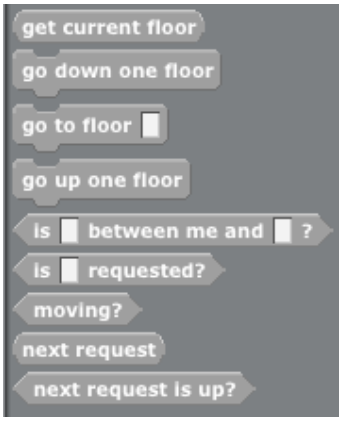
Develop an algorithm to control an elevator.

Your tools

Instead of trying to use real elevators, or doing everything with paper and pencil, we are going to use a simulator. Drag and drop command blocks into the center panel to add them to the program. Connect command blocks together to create sequences of events.

Command blocks are organized into categories. The ones you will be using are **Control, Operators, and Variables.**

Elevator Control Blocks



To get to the Variables panel, click the red Variables button. Scroll all the way to the bottom and you will see gray blocks. These gray blocks control the elevator.

The other categories also hold important blocks.

- Control
  - If a condition, do something
  - Wait until something happens
- Operators
  - Compare numbers: >, =, <
  - NOT operation: reverse yes and no

Figure A-7: Elevator Control instruction sheet.

Appendix B

Analysis Codes

Design Step	Code	Definition
Understand the problem ("ASK")	RB	Read design brief as given to the subjects by the researcher
	DPER	Discussing/referring to performance criteria
	DCON	Discussing/referring to constraints
	PK	Accessing prior knowledge
	CAR	Check available resources
Generate possible solutions ("IMAGINE")	DIS	Discussing possible solutions
	SBS	Selecting best solution
	MAN	Manipulation of materials to explore properties
Modeling a possible solution ("PLAN")	PP	Planning a prototype
	DRAW	Sketching or drawing possible solutions
	MP	Making a prototype
	TEST	Testing one element as the making continues
	AB	Abandon current solution; begin new solution
Building a solution ("CREATE")	IP	Identify a problem with the prototype
	MDC	Making a design change to the prototype
	REF	Refining construction of prototype
Evaluation ("IMPROVE")	EO	Evaluate as subjects observe prototype
	ET	Evaluate as subjects talk about prototype
	ED	Evaluate as subjects draw possible solution
	EDB	Evaluating in terms of the design brief

Table B.1: Starting codes designed by Scribner-MacLean (2009) and informed by Welch (1999). Each code indicates a specific design behavior, and are categorized by the steps of the Boston Museum of Science Design Cycle.

Appendix C

IRB Compliance Documents

C.1 Parent Consent Form

Figures C-1 and C-2 are the Parental Consent Form completed by the legal guardians of all participants in the study. This form was also available in Spanish.

C.2 Student Assent Form

Figures C-3 and C-4 are the Student Assent Form completed by all of the participants in the study.

IRB INFORMED CONSENT or AGREEMENT TO PARTICIPATE FORM
IRB No.:10-046-MAR-XPD Rev. No./Date:2/4-21-10

Consent Form Title: Parent Informed Consent Form

Project Title: Exploration of Natural Design Strategies of Novice Engineers
Principal Investigator: Dr. Fred Martin Associate Professor
Contact Information: UMass Lowell Computer Science, 1 University Avenue, Lowell MA 01854, fredm@cs.uml.edu, 978/934–1964
Co-PI(s):
Student Investigator(s): Mark Sherman
Date Submitted: April 21, 2010

This form has been approved for use by the UML IRB and is valid for up to one year from the approval date.
(PIs -Give a copy of this form to the study participant after they sign it. Originals are to be retained by the PI.)

Authorized IRB Approval Signature: Stephen Dore **Approval Date**: April 21, 2010

The following are essential elements of Informed Consent (these section titles may be edited to suite your needs but the information for each element must be included):

1. Study Purpose: Mark Sherman is a student at the University of Massachusetts Lowell who is exploring how students solve engineering and design problems. This research study will document the design strategies of subjects with little formal design training. This data will help us create better engineering courses and teaching tools.

2. Procedure and Duration: Your child will be asked to participate in a research program called "Engineering & Modeling Activities." This program will involve meeting with your child for 90 minutes on six Thursday afternoons where he/she will be presented with simple engineering activities. Your child will be asked to solve the activity as well as develop a procedure for solving problems like it. All activities will be based on critical thinking and problem solving with technology. Your child will not require any special training to be eligible to participate.

Your child will also be asked to complete a questionnaire to describe their educational interests, including questions about their heritage and your educational background. This questionnaire will not have any identifying information on it and will only be used by the researchers to evaluate the program.

Your child will be selected to participate on a first-come, first-serve basis through the Youth Development Organization. Your child must be in grades 6-8. The YDO will also provide transportation to and from the program.

We ask for your permission to use parts of your child's work (such as writing, diagrams, and explanation) in articles and electronic publications. We also ask your permission to use video recording in the project classroom. The recordings will be used to construct an understanding of student thinking as they carry out and discuss work. No identifying information will be associated with this material. Videos and images of your child will only be used for research analysis and will not be released in any publication without additional written permission from you and your child.

This class will meet in Olsen 302 on UMass Lowell's North Campus.

3. Potential Risks and Discomfort: There is no risk involved in your child's participation in this study.

Figure C-1: Parental consent form, page 1.

4. Incentives/Compensation (if any): There is no payment or financial reward that is provided as compensation for participation in this study.

5. Anticipated Benefits to the Subject or to Non-subjects: We wish to work with your child because we believe this study will lead to improved engineering curriculum for the educational community. Your child may personally gain skills in abstract problem solving, and/or increased understanding of and interest in design principles.

6. Right to Refusal or Withdrawal of Participation: Participation in this study is completely voluntary. This program is only for parents and students who agree to participate in this research study. You may decide not to participate at any time without any penalty. This decision will not affect other services provided to you by the Youth Development Organization or the University of Massachusetts Lowell.

7. Assurances of Privacy and Confidentiality: Only the researchers will have access to recorded materials. All research data will be strictly confidential. Your child's name and all other identification will be removed from everything that is collected, including images, audio recordings, and video recordings. Every precaution will be taken to protect your child's privacy and confidentiality in the data collected. Recorded videos will be destroyed no later than three years after the completion of this research project. Publications based on this research will not include any participant identifiable information.

8. Additional Information (Include contact information for researchers): This form is also available in Spanish. Please contact the researchers to obtain this version. If you do not understand any portion of this form we will be happy to provide a complete explanation. Questions relating to this research project are welcome at any time. Please contact Dr. Fred Martin, Associate Professor, UMass Lowell Computer Science, 1 University Avenue, Lowell MA 01854, fredm@cs.uml.edu 978 -934-1964 Thank you.

PRINCIPAL INVESTIGATOR SIGNATURE(S) -See definition of PI for who is authorized to sign here.
1. Printed Name: **FRED MARTIN** Signature: Date: **APRIL 21, 2010**

PERSON OBTAINING CONSENT
Printed Name: **HOWARD STICKLOR** Date:

Signature:

PARENT/GUARDIAN SIGNATURE
I understand the risks, requirements, and protocols that have been described in this document. I have read this entire document, have had the opportunity to fully discuss my concerns and questions, and fully understand the nature and character of my child's involvement in this research program as a participant and the possible risks and consequences.

Parent, Guardian, or Legal Representative (if applicable)**:**
Printed Name: Date:

Signature:

Child's Name:

Figure C-2: Parental consent form, page 2.

IRB INFORMED CONSENT or AGREEMENT TO PARTICIPATE FORM

IRB No.:10-046-MAR-XPD Rev. No./Date:

Consent Form Title: Student Assent Form

Project Title: Exploration of Natural Design Strategies of Novice Engineers
Principal Investigator: Dr. Fred Martin Associate Professor
Contact Information: UMass Lowell Computer Science, 1 University Avenue, Lowell MA 01854, fredm@cs.uml.edu, 978/934–1964
Co-PI(s):
Student Investigator(s): Mark Sherman
Date Submitted: April 21, 2010

This form has been approved for use by the UML IRB and is valid for up to one year from the approval date.
(PIs -Give a copy of this form to the study participant after they sign it. Originals are to be retained by the PI.)

Authorized IRB Approval Signature: Stephen Dow **Approval Date**: April 21, 2010

You are being asked to enroll in a program titled "Engineering & Modeling Activities." This program is a collection of interesting activities that involve thinking like an engineer. Part of the each activity will be to design instructions that could be used to get another person to solve the activity like you did. This program will meet for 6 sessions, each session is 90 minutes long. This class will meet in Olsen 302 on UMass Lowell's North Campus.

We are interested in how you think about design problems. We are asking for your permission to use parts of your work for publications about this program. Things we may use include your writing, drawings, and explanations. Your name, grade, town, and any other information that can be used to identify you will be removed from everything we use right away. We won't keep your name or identifying information at all.

We are asking for your permission to video and audio record you during the activities. These videos are for us to better understand how you are thinking during the activities. Your name and other information that can be used to identify you will not be attached to video or audio. These videos will only be seen by the teachers conducting the activities, and will not be published without additional written permission from you and your parent/guardian.

You will be asked to fill out a questionnaire about your favorite subjects, your heritage, and your parents' education. Your name will not be on this.

There are no risks involved in being a participant in this study. There is no payment or financial reward that is provided as compensation for participation in this study.

If you change your mind, you may leave the program at any time without any consequences to you from the Youth Development Organization or the University of Massachusetts Lowell.

All information collected will be confidential. Your name or any other identification will never be disclosed. We will protect your privacy and confidentiality. Recorded tapes will be destroyed no later than three years after the completion of this research project.

If you do not understand any portion of this form we will be happy to provide a complete explanation. Questions relating to this research project are welcome at any time. Please contact Dr. Fred Martin, Associate Professor, UMass Lowell Computer Science, 1 University Avenue, Lowell MA 01854, fredm@cs.uml.edu 978 -934-1964. Thank you.

Figure C-3: Student assent form, page 1.

PRINCIPAL INVESTIGATOR SIGNATURE(S) -See definition of PI for who is authorized to sign here.
1. Printed Name: **FRED MARTIN** Signature:  Date: **APRIL 21, 2010**

PERSON OBTAINING CONSENT
Printed Name: **HOWARD STICKLOR** Date:

Signature:

PARTICIPANT SIGNATURE
I understand the foreseeable risks and/or discomfort that have been described in this document. I have read the statements contained herein, have had the opportunity to fully discuss my concerns and questions, and fully understand the nature and character of my involvement in this research program as a participant and the attendant risks and consequences.

Research Participant:
Printed Name: Date:

Signature:

Figure C-4: Student assent form, page 2.