MODEL-BASED ROUTE PLANNING AND DIFFICULTY ESTIMATION OF INDOOR BOULDERING PROBLEMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

MERT TÜREDİOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF COGNITIVE SCIENCE

JANUARY 2023

MODEL-BASED ROUTE PLANNING AND DIFFICULTY ESTIMATION OF INDOOR BOULDERING PROBLEMS

Submitted by Mert Türedioğlu in partial fulfillment of the requirements for the degree of **Master of Science in Cognitive Science Department, Middle East Technical University** by,

Prof. Dr. Banu Günel Kılıç
Dean, **Graduate School of Informatics** _____

Dr. Ceyhan Temürcü
Head of Department, **Cognitive Science** _____

Assoc. Prof. Dr. Barbaros Yet
Supervisor, **Cognitive Science Dept., METU** _____

Examining Committee Members:

Prof. Dr. Cem Bozşahin
Cognitive Science Dept., METU _____

Assoc. Prof. Dr. Barbaros Yet
Cognitive Science Dept., METU _____

Asst. Prof. Dr. Diclehan Tezcaner Öztürk
Industrial Engineering Dept., Hacettepe University _____

Date: 26.01.2023

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Mert Türedioğlu

Signature : _____

ABSTRACT

MODEL-BASED ROUTE PLANNING AND DIFFICULTY ESTIMATION OF INDOOR BOULDERING PROBLEMS

Türedioğlu, Mert

MSc., Department of Cognitive Sciences

Supervisor: Assoc. Prof. Dr. Barbaros Yet

January 2023, 74 pages

Bouldering – a subdiscipline of climbing– is a both mentally and physically demanding sport as it challenges both the problem-solving skills and physical abilities of climbers. In order to be successful, boulder climbers must find fast and accurate solutions to novel climbing problems they encounter. This study focuses on the decision-making processes of climbers when solving boulder problems in an artificial and standardized climbing wall called MoonBoard. MoonBoard is a popular climbing wall that has a large international community. This study aims to build a goal-based AI agent that learns from previous solutions to plan the sequence of actions for novel boulder problems it encounters. We evaluate the agent's cost estimates and climbing solutions by comparing it to the difficulty estimations and solutions provided by expert climbers.

Keywords: Artificial Intelligence, Bayesian Network, Machine Learning, Climbing, Bouldering

ÖZ

KISA KAYA PROBLEMLERİNİN MODEL TABANLI ROTA PLANLAMASI VE ZORLUK TAHMİNİ

Türedioğlu, Mert

Yüksek Lisans, Bilişsel Bilimler Bölümü

Tez Yöneticisi: Doç. Dr. Barbaros Yet

Ocak 2023, 74 sayfa

Tırmanışın bir alt disiplini olan kısa kaya tırmanışı, tırmanışçıların hem problem çözme becerilerini hem de fiziksel yeteneklerini sınayan bir spordur. Başarılı olmak için, kaya tırmanışçıları karşılaştıkları yeni tırmanış problemlerine hızlı ve doğru çözümler bulmalıdır. Bu çalışma, yapay ve standartlaştırılmış bir tırmanma duvarı olan MoonBoard'taki kısa kaya problemlerini çözerken tırmanışçıların karar verme süreçlerine odaklanmaktadır. MoonBoard, geniş bir uluslararası topluluğa sahip popüler bir tırmanma duvarıdır. Bu çalışma, karşılaştığı yeni kısa kaya problemleri için hamle sırasını planlamak üzere önceki çözümlerden öğrenen, hedef tabanlı bir yapay zeka modeli geliştirmeyi amaçlamaktadır. Modelin hamle zorluğu tahminlerini ve tırmanış çözümlerini, uzman tırmanıcılar tarafından sağlanan zorluk tahminleri ve çözümlerle karşılaştırarak değerlendiriyoruz.

Anahtar Sözcükler: Yapay Zeka, Bayes Ağı, Makine Öğrenmesi, Tırmanış, Kısa Kaya Tırmanışı

To My Family and Friends

ACKNOWLEDGMENTS

First of all, I would like to express my gratitude and appreciation for Barbaros Yet, whose guidance, support, and encouragement have been invaluable throughout this thesis. I also wish to thank Kerem Demirtaş for his guidance and support to this thesis.

I want to thank all my instructors in the department. They brought me to the level of doing a research in this new field for me.

I would also like to thank Çağatay Köksoy for planting the seeds of this thesis idea and Ulaş Celep and Deniz Bayramoğlu for their support.

I would also like to thank Yeşim Koçoğlu for her emotional and intellectual support for this thesis.

Finally, I would like to thank my mother Zeynep Berna Aslan and dear family for their endless support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
DAG	Directed Acyclic Graph
GAM	Generalized Additive Model
NLP	Natural Language Processing
PEAS	Performance Measures, Environment, Actuators and Sensors
PSIS-LOO	Pareto Smoothed Importance Sampling - Leave-one-out-cross-validation

CHAPTER 1

INTRODUCTION

Climbing is a sport that tests one's physical and mental abilities. Physical difficulties include climbing up with the help of his hands and feet in a steep and environment. Alongside the physical difficulties of this task, climbing intensely challenges one's mental skills. Hörst (2008) sees climbing as a well-balanced sport between physical, technical, and mental skill requirements (see *Figure 1*). Each climbing route is a decision problem waiting to be solved. A climbing route is the sum of the climbing holds, of which start holds, and end holds are specified. To achieve a successful climb, the climber plans and executes a movement series that includes only the designated holds in the route. In a climbing competition, the athlete sees the route to climb for the first time. The athlete plans the sequence of movements before climbing in the limited observation time. This plan he developed for the novel route consists of deciding which hold to hold with which limb and in which order. This sequence of movements (solution) determined by the athlete is called beta. When the athlete climbs with a false beta, the required physical effort may increase and cause failure. For example, when Olympic winner in climbing Janja Garnbret broke new record by winning an entire season of IFSC Climbing World Cups (Jahns, 2021), she has accurately and efficiently solved 74 of the 78 novel boulder problems she had encountered the first time and successfully climbed them.

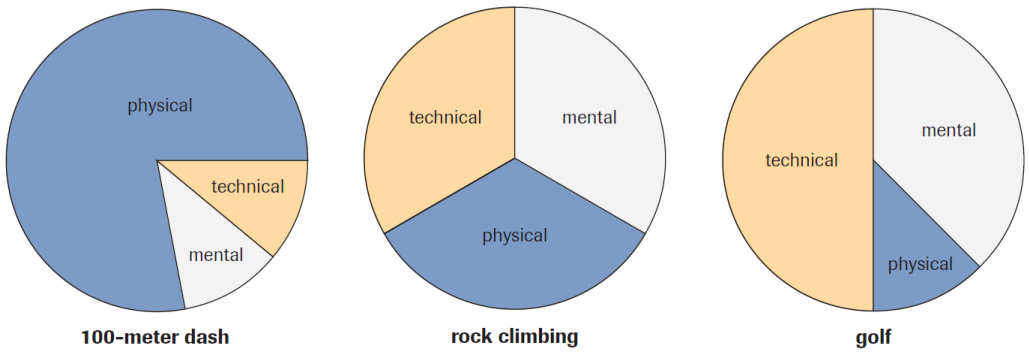

Figure 1: Relative Demands of Various Sports (Adapted from (Hörst, 2008))

Climbing is both an outdoor and indoor sport. It started as an outdoor sport. Later, people started doing this sport on artificial indoor climbing walls to train for outdoor routes. Today, indoor climbing has become a discipline on its own. It gained Olympic sport status with the 2020 Tokyo Olympics (The International Olympic Committee, 2016). Athletes compete in 3 different branches: Speed, Lead, and Bouldering. All these test the climber, physically and mentally, at different levels and ways.

Bouldering routes, also called bouldering problems, are shorter than lead and speed but are the most mentally challenging and hardest to solve. This thesis uses the terms 'routes' and 'problems' interchangeably. Bouldering problems are designed to force athletes make many tries before successfully climbing routes. Efficient solutions are precious in bouldering problems. In International Bouldering World Cups, athletes have four minutes to climb a problem. The number of attempts in this four minute climbing period measures athletes' success on the problem. The athlete must flash the problem to get the highest score. To flash a problem means climbing a problem on the first try. That is why it is essential to find an efficient solution in a few tries. Expert athletes can consistently do so. This shows their improved problem-solving skills as well as physical fitness for this task.

To develop their physical and problem-solving skills, athletes use various tools. One of these tools is ‘training boards’, which has become more popular over the years. The training board is a single-plane artificial climbing wall consisting of many holds. These multiple holds allow for a wide variety of only the designated holds. Some of these training boards are standardized such as MoonBoard. MoonBoard is one of the most widely used climbing boards (see Figure 2). These standardized training boards have large communities. Climbers share their newly created problems and solutions in these online communities.

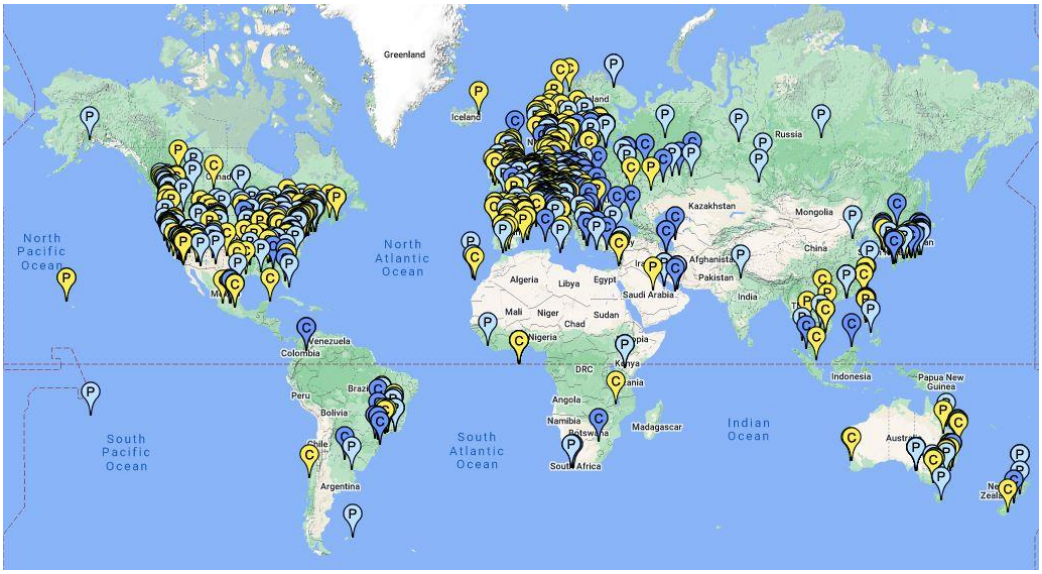

Figure 2: MoonBoards World Map (Adapted from (Moon Climbing, 2022e))

Training boards are well suited for computational research for two reasons. First, climbing boards are standardized. Second, they are in grid structure on which evenly spaced climbing holds bolted. This standardized grid structure enables one to digitize the climbing problems and collect computer interpretable data. This study focuses on climbing problems based on MoonBoard.

MoonBoard climbing board layouts share the same grid structure and dimensions. The layouts of climbing holds have been revised in 2016, 2017, and 2019. We focused on the original MoonBoard layout, which is the 2016 layout. It is a 40-degree negatively angled climbing board (see *Figure 3*). This board comprises two parts: the main board (upper part), which is a 18x11 grid, and the kicker(lower part), which is a 5x2 grid. The kicker part is the same in all layouts. There are ten footholds on the kicker. The main board differs by combinations of MoonBoard hold sets used. The 2016 layout has 140 climbing holds on the main board.

Figure 3: MoonBoard 2016 Layout (Adapted from (Moon Climbing, 2022b))

In a typical MoonBoard problem, the goal is to reach the target hold and grab it with two hands in a controlled manner. *Figure 4 is a* typical MoonBoard problem. Holds marked with colored circles represent the legitimate-to-touch holds in a problem. A MoonBoard problem consists of all these holds. The target hold is red-circled, the start hold is green-circled, and the blue-circled ones are intermediate holds.

Figure 4: A Typical MoonBoard Problem (Adapted from (Moon Climbing, 2022c))

This thesis aims to study decision-making in climbing, especially in bouldering. We tackle this problem by building a goal-based agent that learns movement costs and difficulties from data and plans sequence of actions to minimize these costs. We compare the predictions and plans of the agents with the difficulty classifications and solutions generated by human climbers. We aim to shed light on this decision-making problem by developing an AI agent. To achieve this aim, three research questions are addressed:

> Q.1. Can we design a learning and search-based AI agent making climbing decisions that athletes can implement?
>
> Q.2. Can this agent accurately classify the difficulty of boulder problems?
>
> Q.3. Can this agent find an accurate solution for a novel boulder problem?

The novelty of this study lies in its methodology and subject. Climbing has yet to be studied as a decision-making problem. The other novelty is that the planning task was not

treated as just a distance optimization problem. It is based on data and experience while calculating the costs of all possible moves. Then, it finds an optimized solution to a given bouldering problem. The statistical model was developed in the Bayesian framework to calculate those costs. Data of this statistical model are composed of previous climbs of several climbers of several problems at different grades. The solutions produced were validated with data.

This thesis is structured as follows: *Chapter 2* is a literature review in which related terms of climbing are explained and the studies in the literature reviewed. *Chapter 3* presents the methodology of the study. The AI agent is defined in this chapter, and the learning and planning models are explained. *Chapter 4* reports the results and validations of the models. *Chapter 5* concludes this thesis. The study is summarized, the results are discussed, and suggestions for future studies are shared.

CHAPTER 2

LITERATURE REVIEW

2.1. Related Terms of Climbing

2.1.1. Route

A climbing route is the sum of the climbing holds, of which start holds, and end holds are specified. The climber carries himself up using these holds. To achieve a successful climb, the climber plans and executes a movement series that includes only the designated holds in the route. There are different types of climbing routes, both indoor and outdoor. The length and environment of the climbing routes provide this diversity: sport climbing, bouldering, ice climbing, alpine climbing, big wall climbing, deep water soloing, etc. The focus of this thesis is indoor bouldering routes. Indoor bouldering routes are short and difficult to solve climbing problems. A typical boulder problem consists of designated start and end holds and intermediate holds. The climber plans a sequence of movements involving these holds. When he follows this plan and reaches the end hold, he successfully climbs the indoor boulder route.

2.1.2. Beta

Beta is the name given to one's sequence of moves while climbing a route (Flanagan, 2013). The term beta may be used for a solution only for a part of the route or the whole route. It could be very specific or a rough description. It is expected to be the optimal solution to the climbing problem. Yet, the optimal solution may differ depending on one's anthropometric measurements. There may be more than one beta for a route, depending on one's problem-solving skills. There may be more than one beta depending on one's problem-solving skills.

2.1.3. Hold

Holds are fundamental elements of climbing routes. They are in different shapes and sizes. A climbing hold can be both natural (granite, sandstone, limestone, etc.) or manufactured (polyurethane, wood, fiberglass, granite, etc.). A climbing route consists of holds.

2.1.4. Difference Between Boulder Problem and Climbing Route

Boulder is a large rock fragment shaped by water or weather erosion *(Oxford, 2022)*. There are indoor (artificial, see *Figure 4*) and outdoor (natural, see *Figure 5*) boulder problems. Compared to routes, boulder problems are much shorter. The height of a boulder problem may go up to ten meters. There is no safety gear involved except crash pads. Crash pads are portable cushions that are laid under a boulder.

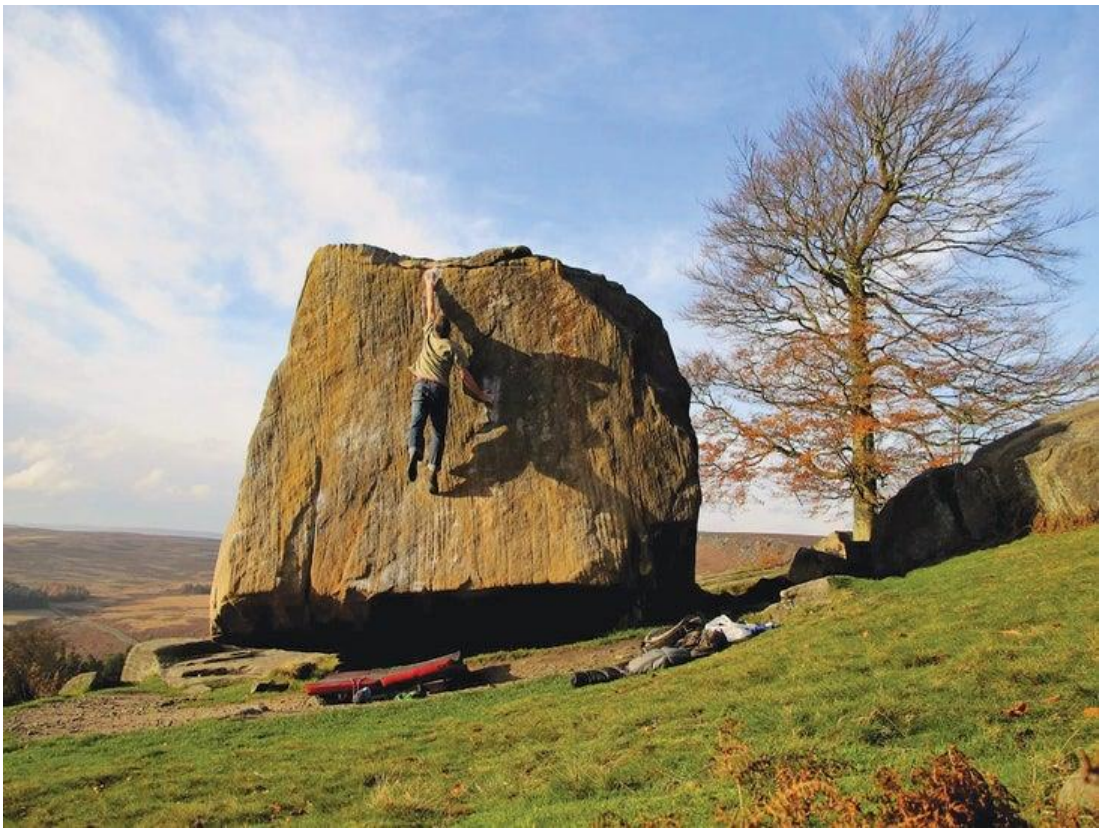

Figure 5: An outdoor boulder problem (Adapted from (Climbing Magazine, 2022))

An indoor boulder problem consists of three groups of holds such as starting holds; intermediary holds; top holds. To complete a boulder problem, one must start climbing only touching starting holds and complete the climb by matching the top hold with both hands in a controlled position. In contrast, an outdoor boulder problem has no specific starting and top holds. A typical boulder problem starts from a specific part of the boulder and ends by climbing onto it.

A climbing route is a specific path one follows to reach the top. The top of a climbing route may be a mountain peak, an anchor of a sports climbing route, or a peak of a frozen waterfall. The route length may vary from five meters to more than two thousand meters. There are indoor and outdoor climbing routes. One may need safety equipment like a climbing rope, gear, and harness to climb a route.

2.1.5. Grade

Grade is the term used to indicate the difficulty level of route or boulder problems in climbing. Grade has no formal definition or formula to calculate it. Routes or problems are graded in comparison to each other. Multiple factors affect the grade: the size of the holds in the route or problems, their direction, their distance from each other, their friction, their grip difficulty, the length of the route or problem, angle of the route or problem. There are multiple grade scales that can be converted to each other. There are separate scales for climbing routes and boulder problems. These scales have emerged regionally, and some are used worldwide. There are two commonly used scales for Boulder problems: The Fontainebleau grading system (font grade for short), The Hueco Scale (or V-Scale). Commonly used scales for climbing routes are YDS (The Yosemite Decimal System), The French Numerical System, The British Grading System, and The UIAA Grading System. For more scales and their analysis, refer to this resource. The scope of this study is boulder problems. The table below shows the two mentioned bouldering grade scales and their conversion to each other.

Table 1: Bouldering Grade Scales Conversion Chart

Hueco	Fontainebleau	Hueco	Fontainebleau
VB	3	V7	7A+
V0-	4-	V8	7B
V0	4		7B+
V0+	4+	V9	7C
V1	5	V10	7C+
V2	5+	V11	8A
V3	6A	V12	8A+
	6A+	V13	8B
V4	6B	V14	8B+
	6B+	V15	8C
V5	6C	V16	8C+
	6C+	V17	9A
V6	7A		

2.1.6. Merged Grade

Merged grade is the name we give to combining multiple grades that are close to each other. There is no such term as merged grade in the literature. However, as mentioned when explaining the grade term, the grades of climbing routes or boulder problems are determined compared to each other. Therefore, from time to time, there are changes of one or two degrees up or down in grades. There are common terms for naming these

changes: upgrade and downgrade. In order to cover such cases, we obtained 'merged grades' by combining the grades of the boulder problems in the study. These types of merged grades are used to indicate the skill level of climbers. A climber's skill level is directly related to the grade of the most difficult problems they climb. Some grade ranges define skill level categories for climbers. There are studies in which participants are grouped according to their skill levels: (Nick et al., 2011) and (Vereide et al., 2022). The participants' skill levels are determined by their hardest route achievement. In these studies, routes are not bouldering routes, but sport climbing routes. There is no research in the literature that groups participants according to their boulder grade ranges. Yet, Hörst grouped climbers according to boulder grade ranges in his book (2008). The merged grade ranges and Fontainebleau grades of the problems in the dataset are specified in Chapter 3.2.1.1.

2.1.7. MoonBoard

A Training board is an artificial climbing wall that consists of manufactured holds. MoonBoard (see *Figure 3*) is one of the standardized training boards. MoonBoard is the pioneer of standardized training boards. What makes it a pioneer and suitable to study is that it is standardized. It is standardized in two ways. First, the board has a specific angle, namely 40 degrees overhanging board. Second, holds are standard, bolted on fixed spots that are the same on every MoonBoard. These features enable us to collect data from different climbers from different locations.

There is more than one layout. In this study, we focused on MoonBoard 2016 layout. It has 140 holds on its 18x11 grid structure. At the time of this study, there were 451 benchmark routes of this layout.

2.1.8. Move

In the context of climbing, move is the name of the transitioning of the hand or feet from one hold to another hold. A move may involve multiple limb relocations. If we call the list of climber's limb positions a state at any given moment, it is possible to conceive the move as transitioning from one state to another. For a figure-supported explanation and computer-readable representation of a move, see *Chapter 3.2.1.4*.

2.1.9. The Distinction of Benchmark and Non-benchmark Problems

There are two types of MoonBoard problems: 'benchmark' problems and 'non-benchmark' problems. Benchmark problems are those whose quality has been validated by the MoonClimbing moderators. Multiple factors determine this quality. As mentioned above, the problems are graded according to their difficulty. Benchmark problems are those whose grade has been validated. Another factor affecting the quality of the problem is that there are neither too many nor too many holds in the problem. Benchmark problems also satisfy this criterion. Such a distinction was needed because there are more than 50

thousand problems in the MoonBoard problem database. Not every problem that community designs and uploads to the dataset meet these criteria.

Two problems are visualized side by side in Figure 6. The problem on the left is a benchmark problem, and the problem on the right is a non-benchmark problem. Holds marked with colored circles represent the legitimate-to-touch holds in the problem. The climber may only use circle-marked holds while climbing the problem. One problem consists of all these holds. The 'non-benchmark problem' on the right has too few holds to be climbable, while the benchmark problem on the left has enough holds.

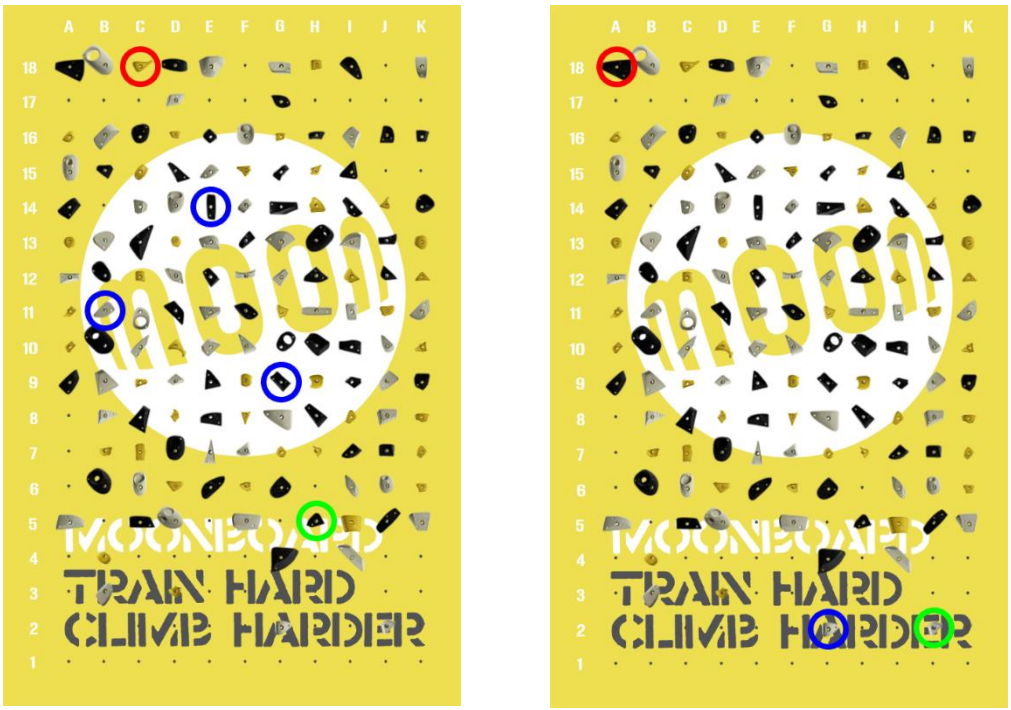

Figure 6: Benchmark (Left Adapted from (Moon Climbing, 2022c)), and Non-benchmark (Right Adapted from (Moon Climbing, 2022d))

2.2. Previous Studies in Climbing

Previous quantitative studies on boulder climbing mainly focused on problem grade estimation and new route generation in training boards such as MoonBoard. These tasks were mainly studied with machine learning algorithms. This section reviews the previous work on MoonBoard problems.

Dobles et al. (2017) used machine learning techniques for boulder problem grade estimation in their research. Their preferred technique is the Convolutional Neural Network technique, which is widely used in pattern recognition. This technique is generally used to detect patterns in images and to recognize the object in the image. The pattern recognition capability of this technique is promising in identifying similar patterns

in route difficulties. They represented the MoonBoard routes as multi-hot encoded as in the above research. As mentioned above, this representation only contains information about which holds are in the route and which are not. Their study experimented with three different classifiers: Naive Bayes classifier, Softmax Regression Classifier, and Convolutional Neural Network classifier. They also used the Support Vector Machine but did not report the result due to its poor performance in the validation set. They compared the accuracy scores of 3 different models with people's predictions. They achieved an accuracy score of 36.5% from all three models. People's accuracy score was reported as 93.4%. They see one of the reasons behind this as people guessing routes not just from pictures but from their climbs.

Tai et al. (2020) approached the boulder problem grade estimation using neural networks in their research. They were inspired by the NLP (Natural Language Processing) domain in which Graph Convolutional Network architectures have proven successful in text classification. They have preprocessed their problems into multi-hot encoded representations. To briefly explain the character of this representation, the route representations consist only of which of the 140 holds of the MoonBoard is on the route. They trained their models with these multi-hot encoded route representations and did not do any feature engineering. This representation does not contain any other information about the characteristics of the routes. They compared the accuracy scores of 17 different machine-learning models. The most successful model in this comparison was the Graph Convolutional Model, inspired by the NLP domain. They compared the accuracy scores of 17 different machine-learning models. In this comparison, the most successful model was the Graph Convolutional Model, inspired by the NLP domain, with an accuracy score of ~73%. In this study, unlike the research in this report, the number of routes used in the model is more than seventeen thousand. In addition to the routes whose difficulty levels and quality were approved by the MoonBoard team (Benchmark Routes), other MoonBoard routes added by the climbers were also used in their research. The 451 Benchmark routes are insufficient sample sizes for developing machine learning models. They used the mentioned community routes (non-benchmark problems) to increase this number while developing and testing their models.

In both studies, one of the most important factors reducing the accuracy score is that the difficulty levels of the route in the sample are not evenly distributed. There are far more routes from easy grades and fewer routes from hard grades. These imbalances have a detrimental effect on models' accuracy performance scores.

Duh & Chang (2021) approached the route difficulty assessment and generation using neural networks in their research. They have established three separate neural network models: BetaMove, GradeNet, and DeepRouteSet. BetaMove is developing a solution to the given route problem. GradeNet estimates climbing route difficulty taking into account the beta provided by BetaMove. DeepRoute set produces climbing routes. They preferred the Recurrent Neural Network algorithm because of the sequential nature of the climbing routes. GradeNet achieved an accuracy rate of 64.3% in the training set and 46.7% in the test set. Climbing routes are labeled as discrete. However, the routes in the same difficulty

label differ in some difficulty. Therefore, Duh & Chang also reported an accuracy rate of +-1 in their research. GradeNet's +-1 accuracy rate was reported as 91.3% on the training set and 84.8% on the test set. The +-1 accuracy rate is a good test indicator in terms of capturing the slight differences in the difficulty of the climbing routes according to the differences in the heights of the climbers. The quality evaluation of the routes generated by DeepRouteSet was made through a survey. The survey evaluated problems under four headings: unnecessary holds, strange moves/fluency of moves, reasonable route, and route quality. Routes produced by DeepRouteSet are presented for evaluation together with MoonBoard routes. DeepRouteSet was found to be more successful than MoonBoard routes in 4 headings. Researchers believe that the success of GradeNet and DeepRouteSet lies in the success of BetaMove. It has successfully produced and evaluated routes suitable for climbing with betas offered by BetaMove to routes.

Stapel (2020) used a heuristic approach for generating new climbing routes. To achieve this purpose, climbing holds, and climbing moves were classified, and the effects of these classifications on the difficulty of climbing routes were evaluated. These classifications are also expected to assist future machine learning research. As in this research, the scope of the research has been reduced to MoonBoard problems. It is assumed that there is a relationship between the climbing hold difficulty and the route grade difficulty. Climbing holds were graded with the help of a questionnaire. The climbers evaluated all the holds on the MoonBoard and suggested a number. With the help of these difficulty suggestions and the climbing hold type, the following possible holds to the climb route hold sequence are selected by a Greedy algorithm. In addition, the distance and rotational angle of the following hold are presented to the model for this Greedy algorithm to consider. The climbers evaluated the climbing routes produced by this algorithm with a questionnaire. People who participated in the survey evaluated climbing routes based on grade, route flow, and enjoyment of the climber. The flow of the generated routes was found to be worse than the flow of the MoonBoard routes. A main limitation of their algorithm is that it does not consider the climber's last position when adding a new hold to the hold sequence. So, it disrupts the flow of the route. Another limitation is that the footholds in the route are not considered.

Seal and Seal (2022) have included graph theory in their work on climbing and routing. Inspired by MoonBoard and similar training boards, they simulated an artificial climbing wall with randomly positioned holds. They then determined a start and end hold on the board with these random holds. With the help of Dijkstra's algorithm, they found the shortest hold path between the starting and ending holds. The paths found are suitable for climbers with specific physical characteristics. Different routes have been found for people with different physical characteristics. Adjacent nodes and edges are determined based on these physical properties. All the holds were filtered according to the defined physical properties, and graphs were created from the appropriate nodes. They found that the model found different routes for people of different heights. They compared the difficulty levels of the routes found over their node numbers and total distances. Routes that are longer than the shortest route or that contain more nodes are more difficult. In this

paper, it is recommended to include the properties of the holds in the route to estimate the difficulty level.

Çelik (2022) studied the boulder problem difficulty. The research question of his thesis is how much model complexity affects difficulty estimation accuracy. Climber's best practices are included in the model as a set of rules. The model is designed so that violations of these rules increase the route difficulty. There are two elements to increase model complexity: hold properties and climber properties. Hold properties are the type of holds, distance to hold surface, and hold direction. Climber properties are limbs. The base model does not include the climber property. He stated that including Climber properties did not significantly increase accuracy compared to the base model. However, including hold properties in the model significantly improved the model's prediction accuracy.

No study in the literature produces solution sequences that include both hand and foot movements for MoonBoard problems. In order to produce such a movement sequence, movement costs must be calculated. There is no study that uses a machine learning method to learn costs. This study includes both novelties. The next chapter explains the climbing agent includes these novelties. In the next chapter, the climbing agent, which includes these innovations, will be explained.

CHAPTER 3

METHODOLOGY

3.1. Agent

3.1.1 Climber as a goal-based agent

This study aims to design a goal-based agent in the climbing domain. This agent is expected to accomplish two climbing-related tasks: accurately guessing the grade(difficulty of a climb) of a boulder problem and finding accurate, lowest-cost solutions to novel boulder problems. Since the proposed agent is a goal-based agent solution sequences must end with goal-states. The goal state is defined as any position in which both hands are in contact with end nodes of given problem.

This agent learns from experience. There are two datasets for this goal-based agent to gain experience. The first will give the goal-based agent experience in estimating the route difficulty, and the other will help it calculate the costs of the moves to find the lowest-cost solution to the novel boulder problems. The datasets of this study are novel. It was collected from the web for this study. The first dataset was obtained from MoonBoard's website with a bot we developed in Python, and the second dataset was obtained from MoonBoard's mobile application.

Following Norvig & Russell (2020), we define the task environment and structure of the agent in this section..

3.1.1.1. Task Environment

The task environment is defined in terms of Performance Measures, Environment, Actuators and Sensors (PEAS) (Norvig & Russell, 2020). An overview of the performance measures of the agent is listed as follows:

- Finding lowest-cost solutions to boulder problems.
- Giving accurate grades to boulder problems.

- Climbing efficiently in the sense that there is no redundant or extremely hard movement.

- Making climbing decisions that athletes can implement.

Yet, these measures need to be more specific and well-defined. To follow the formal description of task environments Norvig and Russell proposed (2020), the following dimensions of our task will be specified. The dimension options are as follows: Fully Observable or Partially Observable; Deterministic or Stochastic; Single Agent or Multi-Agent; Episodic or Sequential; Static or Dynamic; Discrete or Continuous; Known or Unknown. The properties of our task can be specified as follows: Fully Observable, Single Agent, Deterministic, Sequential, Static, Discrete, and Known. In other words, our agent has information about the whole environment at any time. There is only one agent in the environment; there are no rival or ally agents. The environment is deterministic. It contains no randomness. It is sequential because each move affects the next move while solving a boulder problem. The environment does not change; it is static. It doesn't have any dynamic elements in it. Although solutions to boulder problems are sequential, the task environment is discrete. Solutions consist of discrete states. It is a known environment since the agent knows the resulting states of its actions beforehand.

Below is the specification of all letters in the PEAS acronym for the AI agent:

- **P**erformance measures: Accurate solutions to suggested solutions, Distance, Path Cost, Number of Moves

- **E**nvironment: Directed Graph with the properties: Fully Observable, Single Agent, Deterministic, Sequential, Static, Discrete, and Known.

- **A**ctuators: Character output on screen

- **S**ensors: Keyboard input

3.1.2 Bouldering as an AI problem

The figure below is an example of raw input of a MoonBoard problem. The goal of the AI agent is to reach a goal state of to the problem. The green circled holds are the starting holds of the problem. The red circled hold is the goal hold of the problem. Basically, the agent must find a solution which is a sequence of moves. The sequence must start with green circled holds and ends with red circled goal hold. The goal in this problem is reaching a state in which the agent's both hands are on the goal hold.

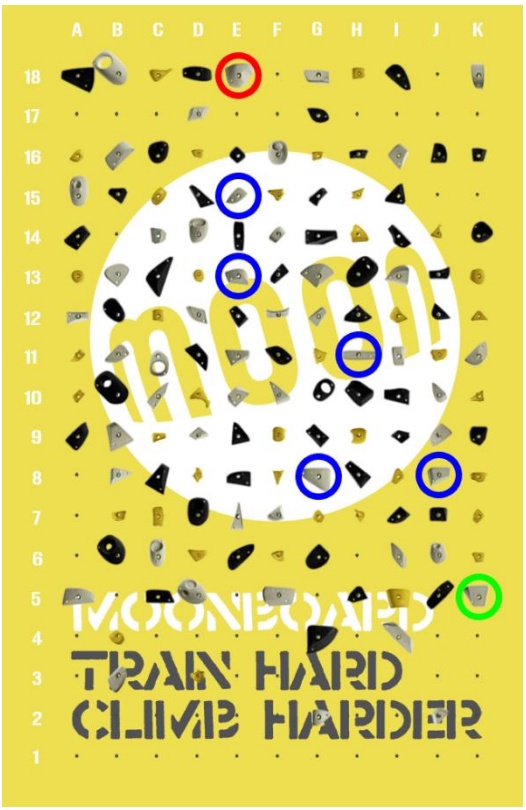

Figure 7: Problem named Bitter (Adapted from (Moon Climbing, 2022g))

To define this problem more systematically, we will define it through five components suggested by Norvig and Russell (2020). These are: initial state, possible actions, transition model, goal test, path cost.

To specify these five components, what state means in our problem should be explained briefly. The details of the representation of states of a climber are given in *Chapter 3.3*. States are lists of the positions of limbs of a climber on the MoonBoard. *Figure 14* shows a climber on a MoonBoard problem. The state of the climber is an ordered list of positions of his limbs which is ['G8', 'H11', 'K5', None].

- Initial State:
    - Hands: They are on designated start hold or holds. If there is only one start hold, then both hands must be on the start hold. If there are two starting holds, one hand must be started with one start hold and the other hand on the other start hold.
    - Feet: The feet must be either on the kicker holds or in such a way that they do not touch any other hold.

- Possible Actions: Take one or more limbs to another hold or cut contact from the hold.

- Transition Model: Updating the state with the new limb positions.

- Goal Test: Checks whether any goal state is achieved. Any goal state must satisfy the following rules:

    - Hands: If there is only one end hold, then both hands must be matched on the end hold. If there are two end holds, then one hand must be on one end hold and the other hand on the other end hold.

    - Feet: The feet must be either on designated holds of the problem or in such a way that they do not touch any other hold.

- Path Cost: Path cost is the sum of all move costs.

3.2. Learning

3.2.1. Data

3.2.1.1. Problem Dataset: The Explanation of the Dataset and An Example MoonBoard Problem

The boulder problem dataset of this study consists of MoonBoard boulder problems. MoonBoard has multiple layouts. We focused on the problems from the MoonBoard 2016 layout. Although there are more than fifty thousand problems in this layout, we only used benchmark problems (for benchmark/non-benchmark distinction check *Chapter 2.1.)*. Climbers set boulder problems and share with other climbers via MoonBoard mobile and web applications. Thanks to the standardized design of the MoonBoard, any climber can access any same boulder problem and climb those problems. We acquired the problems from the MoonBoard website. With the help of a bot built in Python, we collected the hold information and difficulty level of those boulder problems. We developed that bot using Python's Selenium package to acquire problem dataset.

There are 451 benchmark boulder problems in the problem dataset. The problem dataset is slightly skewed towards easier grades. The below histogram shows the original grade (Fontainebleau) distribution of boulder problems:

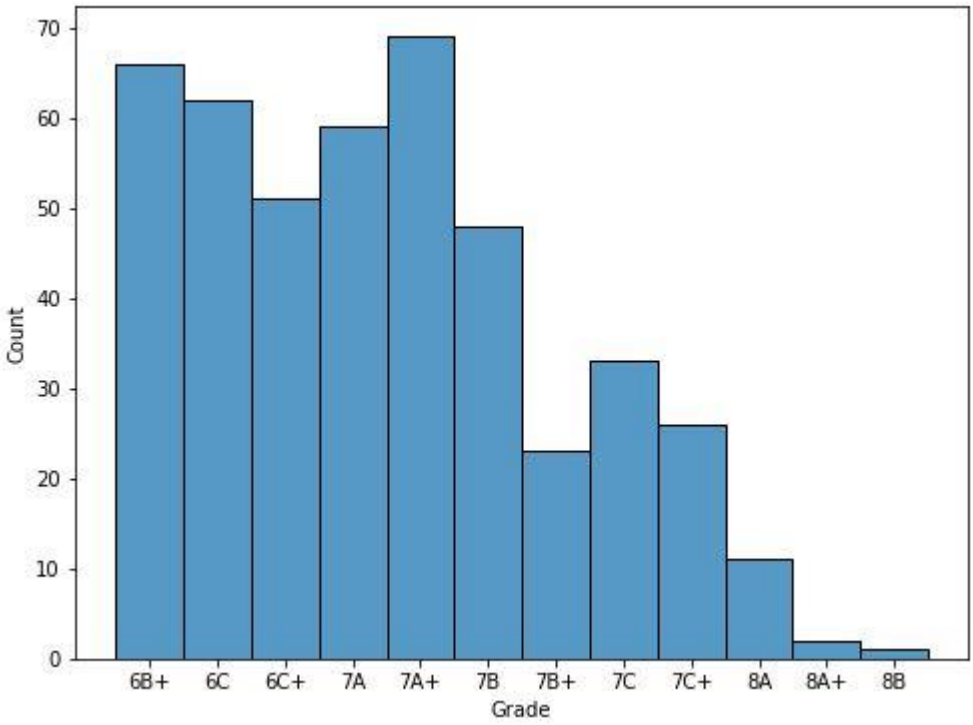

Figure 8: Distribution of Problems by Grade

We merged Fontainebleau grade categories into 3 categories: Intermediate, Advanced, Elite. For the detailed explanation of term merged grade and the reason for this need, see *Chapter 2.1*.

- Intermediate: from 6B+ to 7A. 238 problems
- Advanced: from 7A+ to 7C. 173 problems.
- Elite: 7C+ to 8B. 40 problems.

The below histogram shows the merged grade distribution of boulder problems.

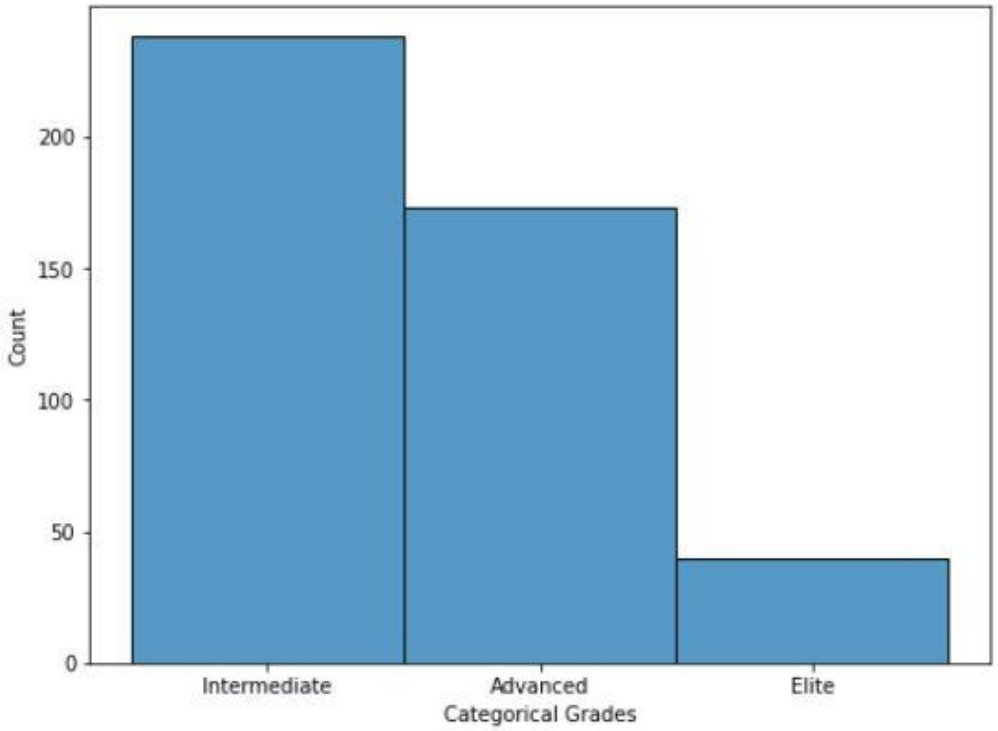

Figure 9: Distribution of Problems by Merged Categorical Grade

3.2.1.1.1. *An Example MoonBoard Problem*

The problem dataset consists of MoonBoard problems. A MoonBoard problem has two essential features: the holds that make up the route and the grade of the route. A MoonBoard problem consists of holds specified on the MoonBoard. There are three types of holds in any MoonBoard problem: starting holds, intermediary holds, and ending holds. End holds are called top holds. A MoonBoard problem consists of 1 or 2 starting holds, 1 or more intermediary holds, and 1 or 2 top holds.

For a graphic of a MoonBoard problem, see *Figure 10* below:

- The starting holds are green-circled.
- The intermediary holds are blue-circled.
- The top holds are red-circled.

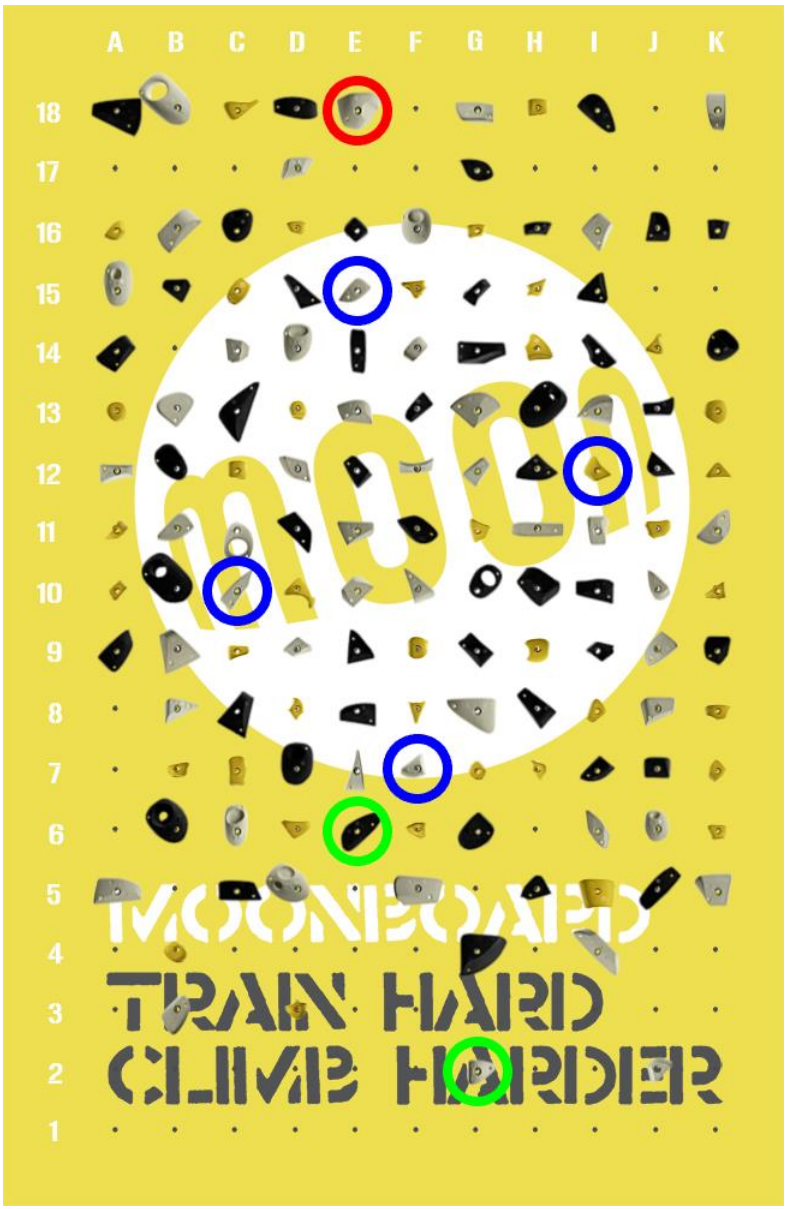

Figure 10: A MoonBoard Problem (Adapted From (Moon Climbing, 2022f))

Figure 10 shows the raw data point of a problem.

Table 2 shows the processed version of that data. The explanation of columns is below the table.

Table 2: An Example of Processed MoonBoard Problem Data

Id	Grade	Merged Grade	Holds	# of Holds	Total Distance	Mean Distance	# of Yellow Holds	Yellow Holds Percent	Total Incut	Mean Incut	Categorical Grades
26620	6C	0	['G2', 'E6', 'I12', 'E15', 'F7', 'C10', 'E18']	7	491.72	81.95	1	0.14	12	1.71	Intermediate

- The 'Id' column contains the route id.
- The 'Grade' column indicates the Fontainebleau grade of the route.
- The 'Merged Grade' column shows the route's grade reduced to 3 grade categories.
- 'Holds' is the list of holds that make up the route.
- The '# of Holds' column contains the total number of holds in the route.
- The 'Total Distance' column is the total distance of the holds in the route from each other.
- 'Mean Distance' is the average of the total distance between the holds in the route.
- '# of Yellow Holds' is the total number of yellow holds on the route.
- 'Yellow Holds Percent' is the percentage of yellow holds on the route.
- 'Total Incut' is the total 'incut size' of the holds in the route.
- The 'Mean Incut' column is the route's average 'incut size'.
- 'Categorical Grades' is the Fontainebleau grade of the route reduced to 3 climber levels (Categories: Intermediate, Advanced, Elite).

3.2.1.3. MoonBoard Holds

There are 140 holds on the 2016 layout. In addition, ten foot holds are fixed to the kicker part (bottom part) of the training board, which is common to all layouts. Each has a

specific position. The difference between the layouts is the variety and positions of the hold sets.

These 140 holds consist of 50 white, 50 black, and 40 yellow holds. All 140 holds on a MoonBoard are unique. None of them are the same. Besides, all MoonBoards are the same. They all consist of the same holds. Color differences of Holds do not indicate a feature difference, except for yellows. Yellow holds are consistently small compared to other color sets and difficult to grab.

The grid structure of MoonBoard makes it convenient to name the locations of the holds. Starting from the bottom left, the x-axis is lettered from A to K, and the Y-axis is numbered from 1 to 18.

Table 11 (see *Appendix B*) shows the classification of the holds. *Table 11* consists of 4 columns:

- The column 'Name' consists of the location information of the holds. Hold's name consists of letters and numbers. The letter part indicates the column; the number part indicates the row.
- The column 'Color' gives the color information of the hold.
- The column ' Incut' specifies the incut size of the hold in 3 levels. Categories are 0, 1, and 2. From 0 to 2, the incut size of the hold increases.
- The column ‘Match’ specifies whether a hold can be grabbed the by two hands.

3.2.1.4. Beta / Solution Dataset of the Bayesian Model for Move Difficulty

The beta / solution dataset of this study consists of solutions to MoonBoard boulder problems. We focused on the solutions from the MoonBoard 2016 layout.

The boulder problem dataset of this study consists of MoonBoard boulder problems. MoonBoard has multiple layouts. Although there are more than fifty thousand problems in this layout, we only used solutions for subset of benchmark problems(for benchmark/non-benchmark distinction check *Chapter 2.1.)*. We collected the solutions from the MoonBoard mobile application.

There are 201 beta / solution data for 183 benchmark MoonBoard problems in the beta / solution dataset. The below histogram shows the original grade (Fontainebleau) distribution of problems in beta / solution dataset:

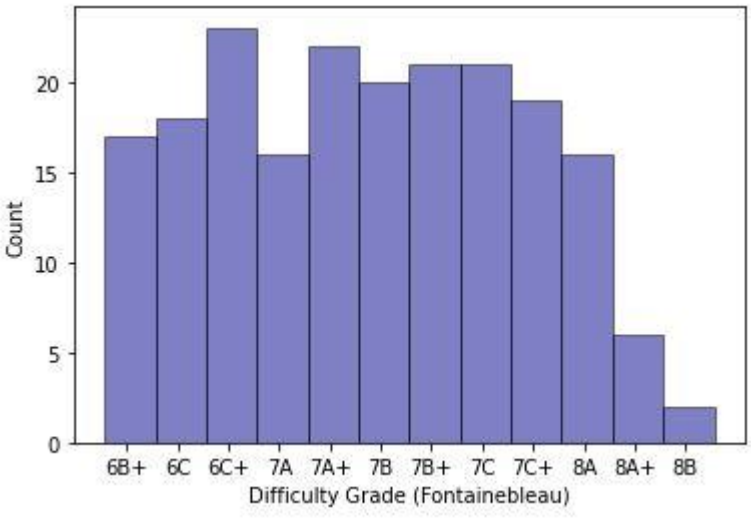

Figure 11: Fontainebleau Grade Distribution of Problems in Beta/Solution Dataset

In each beta data, there are many individual moves. The beta/solution dataset consists of the sum of these unique moves. There are 2456 moves in beta/solution dataset. The below histogram shows the original grade (Fontainebleau) distribution of all moves in beta / solution dataset:

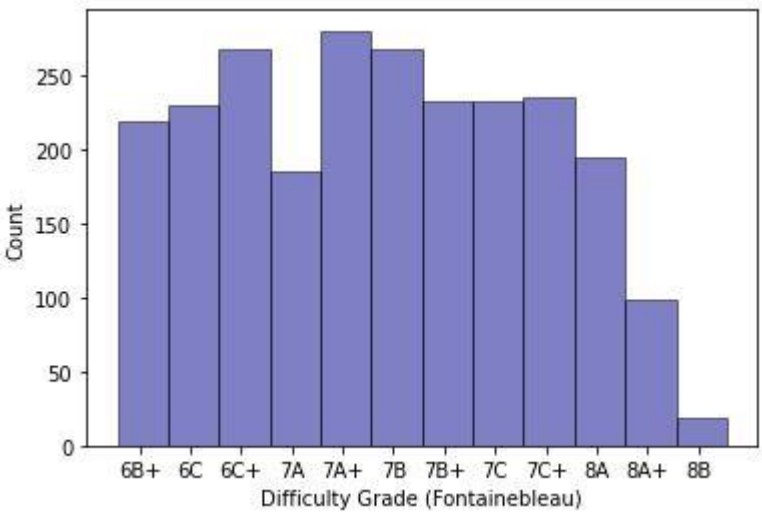

Figure 12: Fontainebleau Grade Distribution of Moves in Beta/Solution Dataset

The Fontainebleau grade categories merged into four categories. The below histogram shows the merged grade distribution of moves in the beta/solution dataset.

- from 6B+ to 6C+, 715 unique moves

- from 7A to 7B, 732 unique moves.
- from 7B+ to 7C, 699 unique moves.
- from 7C+ to 8B, 310 unique moves

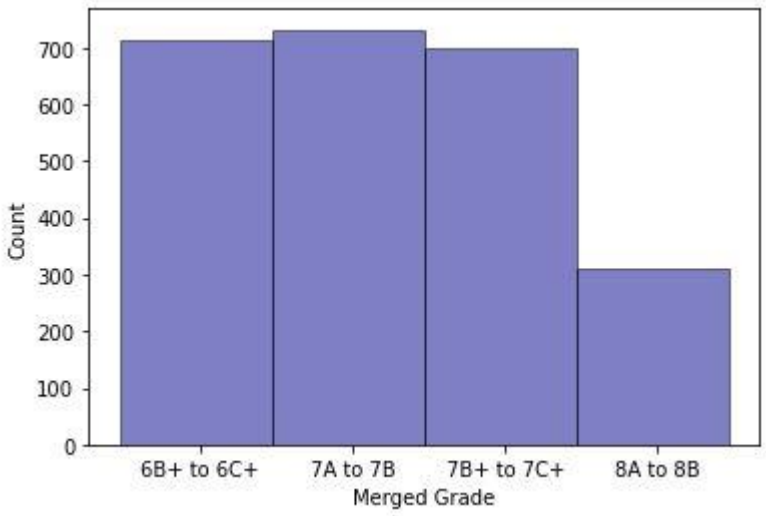

Figure 13: The Merged Grade Distribution of Moves in Beta/Solution Dataset

3.2.1.4.1. *An Example Computer-Readable Move Data*

The beta dataset consists of moves extracted from solutions. Below there are two figures related to two tables. The two figures show the climber's two consecutive moments on a MoonBoard problem. *Figure 14* shows the climber's starting position of a move, and *Figure 15* shows the end position of the move. The sum of these two positions makes one move. *Table 3* shows the computer-readable raw data of this move. *Table 4* shows the processed version of this raw data and the obtained features.

Figure 14: Starting position of a move on MoonBoard

Figure 15 Ending position of a move on MoonBoard

Table 3: Computer-readable raw data of a move

Move ID	Grade	Merged Grade	HN1 Quadruple	HN2 Quadruple
2628460008	6B+	0	['G8', 'H11', 'K5', None]	['E13', 'H11', 'K5', None]

The explanation the columns of the *Table 3* is below:

- The ‘Move ID’ column contains the move id.
- The 'Grade' column indicates the Fontainebleau grade of the problem in which the move is made.
- The 'Merged Grade' column shows the problem’s grade reduced to 3 grade categories.
- The 'HN1 Quad' lists the climber's limb positions. It consists of left-hand, right-hand, left-foot, right-foot, respectively. If a limb does not touch a hold on the board, it is represented as 'None'. HN1 represents the starting position of a move.
- The 'HN2 Quad' lists the climber's limb positions. It consists of left-hand, right-hand, left-foot, right-foot, respectively. If a limb does not touch a hold on the board, it is represented as 'None'. HN2 represents the ending position of a move.

Table 4: Computer-readable processed data of a move

Move ID	Grade	Merged Grade	HN1 Quadruple	HN2 Quadruple	Total # of 'None's	COG	Distance	Total None Cat.
26284 60008	6B +	0	['G8', 'H11', 'K5', None]	['E13', 'H11', 'K5', None]	2	(153.33, 178.33)	126. 16	0

The explanation the columns of the *Table 4* is below:

- The ‘Move ID’ column contains the move id.
- The 'Grade' column indicates the Fontainebleau grade of the problem in which the move is made.
- The 'Merged Grade' column shows the problem’s grade reduced to 3 grade categories.
- The 'HN1 Quad' lists the climber's limb positions. It consists of left-hand, right-hand, left-foot, right-foot, respectively. If a limb does not touch a hold on the board, it is represented as 'None'. HN1 represents the starting position of a move.
- The 'HN2 Quad' lists the climber's limb positions. It consists of left-hand, right-hand, left-foot, right-foot, respectively. If a limb does not touch a hold on the board, it is represented as 'None'. HN2 represents the ending position of a move.

- The 'Total # of 'None's' column gives the total number of None in the HN1 Quadruple and HN2 Quadruple. In other words, it provides the total number of limbs not touching the board in the starting and ending positions of the move.

- The 'COG' column gives the approximate point of Center of Gravity of the climber at the starting position of the move. Treats the board as a coordinate plane and returns an X, Y value.

- The 'Distance' column gives the Euclidean distance from the climber's COG(Center of Gravity) to the new hold it contacts in an ending position.

- The 'Total None Cat.' column shows the value in the 'Total # of 'None's' column converted to the 3-level None Category (0, 1, or 2).

3.2.1.5. Problem Dataset of the Planning Model

The problem dataset to test the network consists of 30 problems. The network is explained in *Chapter 3.3.1*. These problems are benchmark problems. For the figure of an example benchmark problem, see Chapter 3.2.1.2. For an explanation of the benchmark/non-benchmark distinction, see Chapter 2.1. There is a balanced distribution in terms of grade in the dataset: There are seven problems in the 6B+ - 6C+ (merged grade 0) range, eight problems in the 7A-7B range (merged grade 1), eight problems in the 7B+-7C+ range (merged grade 2), and seven problems in the 8A-8B range (merged grade 3). The number of nodes of the problems ranges from 4 to 11. There are an average of 6.7 nodes in 30 problems. See *APPENDIX C* for the problem list of the network.

3.2.2. Models

Bayesian models were used to predict (see Chapter 3.2.2.1) to predict the grade of the routes and to estimate the difficulty of moves. The data described These models were learned from the dataset of route grades and moves described in Section 3.2.1. The second model for predicting the difficulty of moves were used to define the costs in the search algorithm that plans the sequence of moves.

A Bayesian network is a probabilistic graphical model that consists of a graphical structure (Directed Acyclic Graph also known as DAG) and parameters of conditional probability distributions corresponding to the structure (Pearl, 1988; Yet et al., 2016). The graphical structure of a Bayesian network is composed of nodes representing variables and directed edges representing the relations between the variables. The parameters of a Bayesian network represent the nature and strength of these relations represented by the edges. Both Bayesian Networks designed in this study are trained using the PyMC package developed in Python (Salvatier et al., 2016).

DAG is a directed acyclic graph in which there are no directed circles. A DAG consists of nodes and edges. DAGs are used to represent causal relations between variables. They are helpful tools to investigate causal relations between variables and enable us to decide what

effect can be calculated given the causal relations. In this study, two DAGs were drawn for both models (see *Figure 16* and *Figure 17*).

3.2.2.1. Bayesian Network for Estimations of Grades of Problems

A Bayesian network was designed to estimate problem grades. The dependent variable of this network is grade. As stated earlier (see Grade in chapter 2.1.), grade is a categorical, ordinal variable". Ordered Logit was preferred because it is more suitable for the categorical, ordinal variable. The variables of this Bayesian network are listed as follows:

- Number of holds (N): Total number of holds in the problem.
- Distance (D): The average distance between the holds
- Percentage of Yellow holds (Y): What percentage of holds are yellow.
- Incut (I): The average of the sum of the incut sizes of the holds
- Grade (G): Merged grade of the problem

In APPENDIX A.1 the average values, and the standard errors of a variable for each merged grade is drawn. The dashed line in each figure shows the overall average of the variable. Standardized values are used.

The DAG below represents the Bayesian Network. The parameters and conditional dependencies of the structure shown in this figure are below.

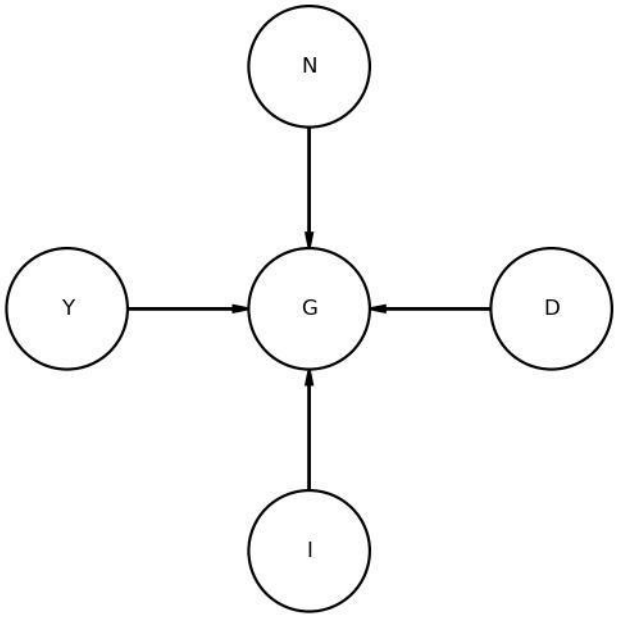

Figure 16: Directed Acyclic Graph (DAG) of Problem Grade Model

Parameters and equations of the model:

$$G_i \sim \text{OrderedLogit}(\phi_i, \alpha)$$

$$\phi_i = \beta_D D_i + \beta_N N_i + \beta_Y Y_i + +\beta_I I_i$$

$$\beta_D, \beta_N, \beta_Y, \beta_I \sim \text{Normal}(0,1)$$

$$\alpha_j \sim \text{Normal}(0,1)$$

3.2.2.2. Bayesian Network for Estimations of Move Difficulty

A Bayesian network was designed to estimate move difficulties. The dependent variable of this network is grade. As stated earlier in Chapter 2.1, grade is a categorical, ordinal variable. Ordered Logit was preferred because it is more suitable for the categorical, ordinal variable. The dependent and independent variables are following:

Independent Variables:

- Distance (D): Distance from the target hold to the center point of contact points in the starting position of the move

- Number of limbs has no contact to any hold (N) Total number of holds in the problem

Dependent Variable:

- Grade (DG): Merged grade of the problem.

In APPENDIX A.2 the average values, and the standard errors of a variable for each merged grade is drawn. The dashed line in each figure shows the overall average of the variable. Standardized values are used.

Figure 17 shows the structure of this model. The parameters and conditional dependencies of the structure shown in this figure are below.

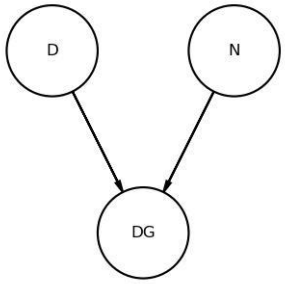

Figure 17: Directed Acyclic Graph (DAG) of Move Grade Model

$$G_i \sim \text{OrderedLogit}(\phi_i, \alpha)$$

$$\phi_i = \beta_{D,H[i],T[i]} D_i + \beta_N \sum_{j=0}^{N_i - 1} \delta_{j\beta_D}$$

$$\beta_D \sim \text{Gamma}(2,2)$$

$$\beta_N \sim \text{Normal}(0,1)$$

$$\alpha_j \sim \text{Normal}(0,1)$$

$$\delta \sim \text{Dirichlet}(\alpha)$$

3.3. Planning

One of this thesis aims is to develop an AI agent that finds accurate solutions to MoonBoard problems. This task is defined as a routing problem on a network. The grid structure of MoonBoard enables us to define this task as a routing problem on a network.

Below the details of the network and transformation of the network into a hypernetwork are explained.

3.3.1. Network

MoonBoard is conceived as a raw network. The holds attached to the MoonBoard are the nodes of this network. However, regarding this problem, this raw network is not adequate yet for our routing task. A transformation is needed since a climber uses more than one node at any moment while climbing a problem. Up to four limbs of a climber are in contact with the MoonBoard at any moment. Therefore, this network is transformed into a hypernetwork in which nodes are transformed into hypernodes. While a node consists of one hold, a hypernode consists of up to four nodes. This hypernetwork will be called the network in the remainder of this study.

Multiple networks are created for each problem. The reason for creating more than one network for a problem is as follows. Every problem has a specific starting node/s and ending node/s. Multiple possible start and end body positions (hypernodes) containing these nodes are possible. For a detailed description of Legitimate start and end positions, see Section 3.1.2. Networks containing all possible start/end position combinations are defined for each problem. After finding the shortest path with the Dijkstra algorithm in these networks, the lowest cost one is the solution found by the planning model. A detailed explanation of the algorithm and the elements that make up these networks is below.

This network will be explained through an example problem. In this example, 3 figures will be referred: *Figure 18, Figure 14, and Figure 15*. *Figure 18* is an example input of raw network. This figure shows the graphic of a MoonBoard problem named “Bitter”. The circled holds are the allowed holds on MoonBoard to grab while climbing the problem. *Figure 14* and *Figure 15* are the consecutive states of a climber on the problem. The elements of the network are following.

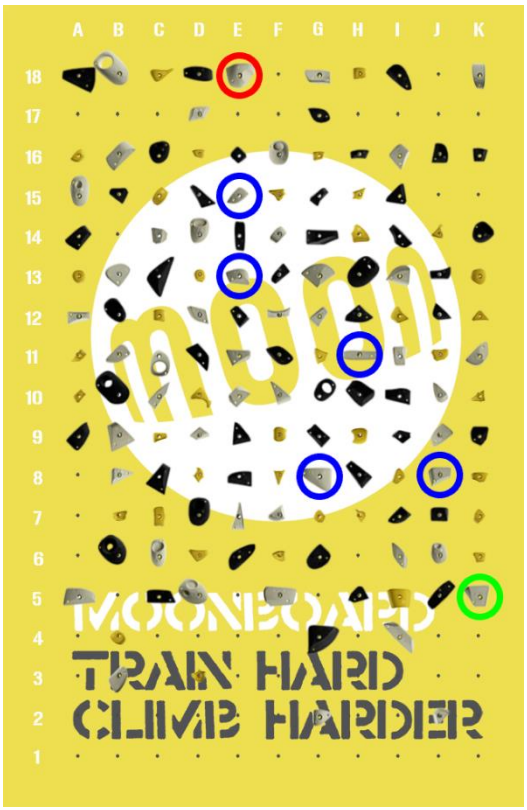

Figure 18: Problem named Bitter (Adapted from (Moon Climbing, 2022g))

3.3.1.1. Node

A raw network's nodes are the holds permitted to use in a solution. There are three types of nodes in any raw network:

- Holds that define the problem. In this example, holds that are circled: "K5", "I8", "G8", "H11", "E13", "E15", and "E18".
- There are 10 foot holds below the main board. All problems, by default, share these holds.

3.3.1.2. Hypernode

Hypernodes consist of nodes. Hypernode is a representation of a state of a climber while climbing a problem. In hypernode, the limbs of a climber are represented. The form of the generic hypernode is [“Left-Hand”, “Right-Hand”, “Left-Foot”, “Right-Foot”]. In hypernode, limbs are always represented in a specific order. For example, *Figure 14* is a photo of the climber on the problem (see *Figure 17).* The hypernode representation of the *Figure 14* is [“G8”, “H11”, “None”, “K5”]. *Figure 15* is the next state comes after the state pictured in *Figure 12.* The hypernode representation of *Figure 15* is [“E13”, “H11”, “None”, “K5].

A hypernode consists of 4 permutations of all nodes in the raw network. After network transformation, in principle, there are 73,440 possible hypernodes in the network of this problem. 4 permutations of 18 (7 circled nodes, 1 “None” node, 10 shared foot holds) equals 73440. However, not all these hypernodes are included in the network. Infeasible hypernodes are filtered. *Chapter 3.3.1.5.* gives the details of this elimination.

3.3.1.3. Hyperedge

Hyperedges connect hypernodes. Hyperedges represents moves of climbers. Hyperedges are directed. Each hyperedge has a cost value. The cost of an hyperedge detailed in *Chapter 3.3.1.4*. For example, there is a hyperedge that connects hypernode representations of *Figure 14* and *Figure 15.*

After the network transformation, in principle, there are more than five billion possible hyperedges in the network of the problem *in Figure 18.* However, not all these hyperedges are included in the network. Infeasible hyperedges are filtered. *Chapter 3.3.1.5.* gives the details of this elimination.

3.3.1.4. Hyperedge Cost

To be able to find a path with minimum cost, hyperedges need cost values. *Table 3* shows the raw data of a hyperedge that connects the hypernode representations of *Figure 14* and *Figure 15*. This raw hyperedge data is processed to get cost values of hyperedges . The output values of this process are given in *Table 4*. These output values are the input parameters of a hyperedge cost function. The coefficients of these parameters are obtained after training of Bayesian Networks explained in *Chapter 2.2.2*. For the details of the coefficients, see *Chapter 3.2.2.2.*

3.3.1.5. Filtering out Infeasible Hypernodes and Hyperedges

On average, there are roughly seven holds in a typical MoonBoard problem. In principle, this makes more than seventy thousand possible hypernodes and more than five billion possible hyperedges. Not all these hypernodes and hyperedges are feasible. These infeasible hypernodes and hyperedges are filtered out. These filtrations are handled with two functions: one function is for infeasible hypernodes, and another is for infeasible hyperedges.

The first function is to filter out infeasible hypernodes. Recall that a hypernode is a representation of the limbs of a climber. Some of the possible hypernodes are beyond the physical capabilities of any human being. For example, some possible hypernodes require 4 meters tall person or a person with 3 meters arm-span. Hypernodes that are beyond physical capabilities are filtered out. The physical limits are derived from the beta / solution dataset and expert knowledge. They are defined in a dictionary, and each possible hypernode is tested to determine whether it exceeds these limits.

The other function is to filter out infeasible hyperedges. Hyperedge is a representation of a move of a climber. Some hyperedges were beyond the physical capabilities of any human being. For example, some of the possible hyperedges require 3 meters jump or simultaneous changes in all four limbs. The limits are derived from the beta / solution dataset and expert knowledge. This expert knowledge and the derived limits are reflected in a function. Each possible hyperedge is tested to determine whether it is in parallel with the expert knowledge and within the derived limits.

These two functions reduced the sizes of the networks immensely. As mentioned above, there are more than seventy thousand possible hypernodes and more than five billion possible hyperedges in a network of a typical MoonBoard problem with seven holds. Thanks to these two filtering functions, there are around five hundred hypernodes and fifteen thousand hyperedges in a network of a typical MoonBoard problem with seven holds.

3.3.1.6. Routing Algoritm

The routing algorithm used in this study is Dijkstra's Shortest Path Algorithm. There are several reasons why Dijkstra's Shortest Path Algorithm is preferred in this study. First, climbers always seek solutions for problems with minimum physical demand to climb efficiently. Second, difficulties of any climbing problems are graded according to the most efficient and easiest solution, including our domain of MoonBoard. Third, the networks in this study have positively weighted edges, which makes our task suitable for Dijkstra's algorithm. Finally, Dijkstra's Algorithm is guaranteed to find the shortest path.

After hypernodes and hyperedges are generated for a problem, to find the solutions with the shortest paths, or in other words, solutions with minimum costs, an open source Python package NetworkX (Hagberg et al., 2008) is used. Digraph object of the package is preferred since hyperedges in this study are directed and have positive weights. This problem has 22 feasible start hypernodes and 17 goal hypernodes. A network was created for all binary combinations of these starts and ends. Then, `shortest_path` method applied to each network, which implements the Dijkstra's algorithm. Among all the shortest paths found, the one with the lowest cost was chosen as the path found by the planning model.

CHAPTER 4

RESULTS

In this chapter, the posterior parameters of the Bayesian models described in Section 3.2.2 are presented (Section 4.1) and an example of the outputs of the route planning model described in Section *3.3.1*. is shown (Section 4.2). Validation results of the grade classification, movement cost estimation and planning models are presented in Sections 4.2, 4.3 and 4.4 respectively.

4.1. Posteriors of the Bayesian Models

The Bayesian models are trained with the datasets described in *Section 3.2.1*. The posteriors of both models are computed with PyMC using Markov chain Monte Carlo methods (Salvatier et al., 2016). PyMC is a probabilistic programming library. Below are the posterior sampling results for both models reported as a summary table which are obtained using ArviZ. ArviZ is a Python package developed for exploratory analysis of Bayesian models (Kumar et al., 2019). In addition to these posterior sampling summary tables, both models' out-of-sample prediction accuracies are reported. The out-of-sample performances of both models is computed by Pareto Smoothed Importance Sampling - Leave-one-out-cross-validation (PSIS-LOO). (See (Vehtari et al., 2017) for the details of the PSIS-LOO approach). Following the reporting guidelines proposed by Kruschke (Kruschke, 2021), the results tables and descriptions are provided below for each model.

4.1.1. The Results of the Model for Problem Grade

This model is trained to estimate the problem grade and to study the causal effects of the variables that have been explained. The details of the dataset and variables are given in *Chapter 3.2.1*. *Table 5* shows the summary statistics table of the posterior samplings of the variables. *Table 6* shows the model's PSIS-LOO validation. The sampling was done with four chains and two thousand draws with the target acceptance rate of %95. Each chain is tuned by two thousand draws.

The columns of *Table 5*, which are 'mean,' 'sd,' 'hdi_3%,' and 'hdi_97%,' are related to the model's parameters. The rest of the columns, which are 'mcse_mean,' 'mcse_sd,' 'ess_bulk,' 'ess_tail,' and 'r_hat,' is related to the sampling efficiencies of those parameters. The high-

density intervals (HDI) of parameters reported here cover the 3%-97% range. The dataset is standardized before model fit.

alpha[0] and alpha[1] are the cutpoints of the estimand grade. bY, bI, bD, and bN are the parameters of the independent variables; Y (the percentage of yellow the holds), I (the average of the sum of the incut sizes of the holds), D (the average distance between the holds), N (the total number of the holds), respectively. The summary statistics shows that each independent variable has an effect on the estimand.

The effects of the variables are either positive or negative. The average distance between the holds has the highest positive effect on the difficulty of a problem(0.867 - 1.491). The percentage of yellow holds had a positive effect (0.169 - 0.746). In contrast, the average of the incut sizes of the holds has the highest negative effect on the difficulty of a problem. The easier the holds to grab, the easier the problem. The total number of holds has a negative effect (-0.421 - 0.137).

The summary statistics regarding sampling efficiencies of the model show that our chains sampled efficiently, according to the paper of Vehtari et al. (2019). All R-hat values are 1 indicating efficient sampling. Effective Sample Size (ess_bulk and ess_tail) summary statistics of all parameters also show that our sampling is successful. We took a total of 8000 steps; recall that there are four chains with two thousand draw for each. If Vehtari et al.'s suggestion were adapted to this model, these ess values would have to be greater than 1600 for successful sampling. The ess values of all parameters are greater than five thousand, which indicates that the sampling was successful.

Table 5: Summary Statistics of the Model for Problem Grade

	mean	sd	hdi_3%	hdi_97%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
alpha[0]	-0.113	0.155	-0.405	0.176	0.002	0.001	7459.0	6142.0	1.0
alpha[1]	3.434	0.298	2.872	3.981	0.004	0.003	6723.0	6272.0	1.0
bY	0.442	0.152	0.169	0.746	0.002	0.001	6895.0	5741.0	1.0
bI	-1.277	0.178	-1.605	-0.935	0.002	0.002	5565.0	6470.0	1.0
bD	1.181	0.167	0.867	1.491	0.002	0.002	5947.0	5812.0	1.0
bN	-0.143	0.149	-0.421	0.137	0.002	0.001	6703.0	5792.0	1.0

The following interpretation is suitable here regarding the summary statistics given in *Table 5*. The variables Y and D have positive effects, whereas the variable I and N have negative effects on the estimand grade. In summary, if there are more holds or holds that are easier to grab in a problem, the problem is easier. A problem becomes more difficult

if the percentage of yellow holds in the problem is large or the average distance between the holds increases.

4.1.2. The Results of the Model for Move Difficulty

This model is trained to estimate the move difficulty and to study the causal effects of the variables that have been explained. The details of the dataset and variables are given in *Chapter 3.2.1.4* and *Chapter 3.2.2.2*, respectively. *Table 6* shows the summary statistics table of the posterior samplings of the variables. *Table 7* shows the model's PSIS-LOO validation. The sampling was done with four chains and two one draws with the target acceptance rate of %95. Each chain is tuned by one thousand draws.

The columns of *Table 6*, which are 'mean,' 'sd,' 'hdi_3%,' and 'hdi_97%,' are related to the model's parameters. The rest of the columns, which are 'mcse_mean,' 'mcse_sd,' 'ess_bulk,' 'ess_tail,' and 'r_hat,' is related to the sampling efficiencies of those parameters. The high-density intervals (HDI) of parameters reported here cover the 3%-97% range. The dataset is standardized before model fit.

alpha[0], alpha[1] and alpha[1] are the cutpoints of the estimand grade. bD[0, 0], bD[0, 1], bD[1, 0], bD[1, 1], bNS, delta_ns[0], delta_ns[1] and delta_ns[2] are the parameters of the independent variables; D (distance from the target hold to the center point of contact points in the starting position of the move), N (the total number of limbs that has no contact to any problem). Delta parameters are the parameters of the Dirichlet distribution. Dirichlet distribution was preferred to better reflect the effect of the increased number of limbs not in contact with the board in the move on the difficulty. The delta values here are the values of Dirichlet prior.

Note that there are four different parameters for variable D. The shape of the parameter is two by two. The first index shows whether the move is a hand or a foot move. 0 indicates foot move, and 1 indicates hand move. The second index shows whether the move is longer than a certain threshold distance. We have determined such a threshold because we believe that moving longer than a certain threshold will cause more effect than a linear difficulty increase. This belief became valid when we fit the model. For both thresholds, parameter considerable increases were seen for moves with distances below the thresholds and above. The detailed increase is reported below.

The number of limbs that has no contact with the wall has the highest effect on move difficulty (1.195 - 2.904). The distance also had a positive effect. The distance increase of hand and foot moves affects the difficulty increase. In addition, the effects increased when the distance thresholds were exceeded. Exceeding the distance threshold in hand moves caused more increase of effect(from 0.004 - 0.109 to 0.295 - 0.582) than exceeding the threshold of foot moves (from 0.138 - 0.349 to 0.284 - 0.753).

The summary statistics (see *Table 6*) of regarding sampling efficiencies of the model show that our chains sampled efficiently (Vehtari et al., 2019). All R-hat values are 1 indicating efficient sampling. Effective Sample Size (ess_bulk and ess_tail) summary statistics of all parameters also show that our sampling is successful. We took a total of 4000 steps; recall that there are four chains with one thousand draw for each. If Vehtari et al.'s suggestion were adapted to this model, these ess values would have to be greater than 400 for successful sampling. The ess values of all parameters are greater than two thousand, which indicates that the sampling was successful.

Table 6: Summary Statistics of the Model for Move Difficulty

	mean	sd	hdi_3%	hdi_97%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
alpha[0]	-0.730	0.048	-0.822	-0.643	0.001	0.001	4313.0	3394.0	1.0
alpha[1]	0.576	0.047	0.491	0.665	0.001	0.000	6503.0	2976.0	1.0
alpha[2]	2.212	0.068	2.083	2.336	0.001	0.001	6704.0	3230.0	1.0
bD[0, 0]	0.248	0.056	0.138	0.349	0.001	0.001	4906.0	2430.0	1.0
bD[0, 1]	0.513	0.127	0.284	0.753	0.002	0.001	4329.0	2340.0	1.0
bD[1, 0]	0.051	0.032	0.004	0.109	0.000	0.000	3742.0	2196.0	1.0
bD[1, 1]	0.439	0.076	0.295	0.582	0.001	0.001	4686.0	2851.0	1.0
bN	2.049	0.472	1.195	2.904	0.008	0.006	3372.0	3111.0	1.0
delta_ns[0]	0.383	0.102	0.217	0.591	0.002	0.001	3248.0	2571.0	1.0
delta_ns[1]	0.413	0.122	0.181	0.623	0.002	0.001	4450.0	2718.0	1.0
delta_ns[2]	0.204	0.113	0.023	0.408	0.002	0.001	4914.0	2623.0	1.0

4.2. Outputs of the Planning Model

The problem dataset to test the network described in Section 3.3.1 consists of 30 problems. (for the dataset *see APPENDIX C).* There is a balanced distribution in terms of grade in the dataset:

- Seven problems in the 6B+ - 6C+ range (merged grade 0)
- Eight problems in the 7A-7B range (merged grade 1)
- Eight problems in the 7B+-7C+ range (merged grade 2)
- Seven problems in the 8A-8B range (merged grade 3)

For the planning model to find solutions to boulder problems, start nodes, intermediary nodes, and end nodes in Table 12 are given as input to the planning model. Each problem

consists of the sum of these three node types. A network was generated from the hypernodes created by these nodes. The expected output path from the network can be summarized as follows: the network will develop a sequence of moves, the first hypernode of this sequence contains the start nodes specified in the input, the last hypernode will contain the end nodes from the input, Hypernodes in this sequence of moves can only have nodes specified in the input or a 'None' node.

The outputs of the planning model are the paths with the lowest cost that Dijkstra's algorithm finds on networks developed separately for each problem, i.e., move sequences. The shortest paths consist of hypernodes. The shortest paths are shared in *Appendix D*, and their validation is presented in Section *4.5*.

I will describe the input and output of the planning model over a problem. *Figure 19* is an image of the nodes of this problem. The green circle signs the start node. Blue circles mark intermediate nodes. The red circle also indicates the end node. This node information is given to the planning model as input (list of nodes: F5, H11, E15, and E18).

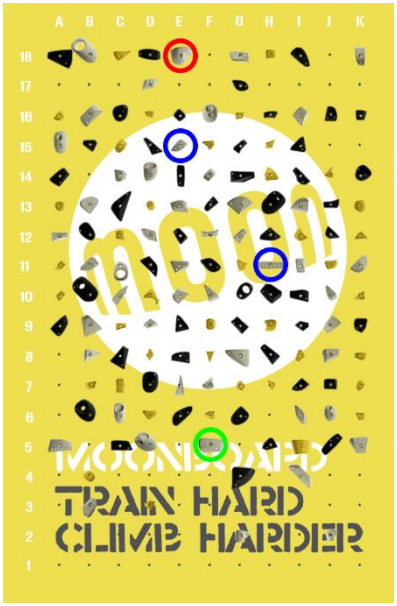

Figure 19: The Nodes of a Problem (Adapted from (Moon Climbing, 2022a))

The model derives all possible hypernodes from these input nodes. A hypernode consists of 4 permutations of all input nodes plus default nodes(10 kicker nodes and a 'None' node). After network transformation, in principle, there are 32.760 possible hypernodes in the network of this problem. Four permutations of 15 (4 circled nodes, 1 “None” node, ten shared foot holds) equals 32,760. Of these possible hypernodes, the infeasible ones are discarded. The number of feasible hypernodes is 134 after this filtering.

Among these hypernodes, 3517 out of 17822 possible hyperedges are feasible. After calculating the costs of the hyperedges with the posterior distributions of the Bayesian

model's (the model detailed in *Chapter 3.2.2.2.*) parameters, the network transformation is completed.

There are multiple start hypernodes and goal hypernodes on this network that match the problem description. The start hypernode must contain all of the start nodes marked in green. The goal hypernode must contain all of the red-marked end nodes. The Dijkstra algorithm finds the shortest paths on the network for all two possible permutations of these start and goal hypernodes. The output of the planning model is a list of hypernodes which has the lowest cost of these shortest paths. The result of the planning model for a problem is this shortest path's hypernode sequence.

Visualizing the entire network is challenging, as even a four hold problem has 134 possible hypernodes and 3517 hyperedges. However, you can refer to *Figure 20* for the visual of the network consisting of hypernodes only in the shortest path of the above problem. NetworkX's related method used to draw the figure (Hagberg et al., 2008).

In *Figure 20*, the green hypernode is the start hypernode, the blue hypernodes are the intermediate hypernodes, and the red hypernode is the goal hypernode. The green hypernode(F5, F5, None, F0) is a legitimate start hypernode because the hypernode's hand nodes(F5) contain the start nodes(F5) of the problem. The red hypernode (E18, E18, None, H11) is a legitimate goal hypernode because the hypernode's hand nodes contain the end nodes of the problem (E18).

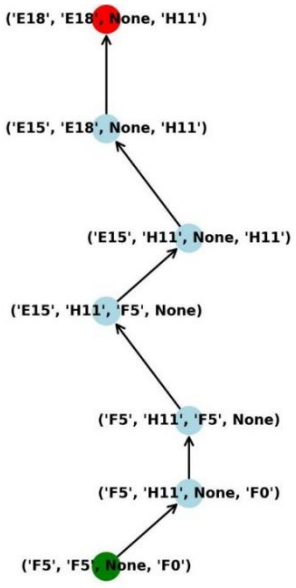

Figure 20: The Network of the Shortest Path

4.3. Validation of the Model for Problem Grade

For validating the problem grade classification model, 451 benchmark problems were. Note that, only the benchmark problems graded by the MoonClimbing moderator team were used to make validation more reliable (benchmark – non-benchmark distinction explained in Section 2.1.9). MoonClimbing is the manufacturer of the artificial climbing board on which the study was conducted, and the moderators team consists of experts in this field.

Firstly, PSIS-LOO cross validation was done using PyMC. *Table 7* shows the results. All estimated shape k parameters are less than 0.5. This makes the estimations reliable, according to the above-mentioned paper of Vehtari et al (2017).

Table 7: PSIS-LOO Validation – Problem Grade

	Estimate	SE
elpd_loo	-269.85	12.43
p_loo	5.39	-

Pareto k diagnostic values:

		Count	Pct.
(-Inf, 0.5]	(good)	451	100.0%
(0.5, 0.7]	(ok)	0	0.0%
(0.7, 1]	(bad)	0	0.0%
(1, Inf)	(very bad)	0	0.0%

Secondly, classification performance of the posterior predictions of the grade classification model were evaluated. Most probable grades in the posterior predictions were compared with the true grades. The 451 benchmark problems were divided into train and test sets. The train-test split was done using the Python module Scikit-learn's train_test_split method (Pedregosa et al., 2011) with 70-30 ratio. This split was done in a stratified fashion since the dataset set was imbalanced. There were more problems in the easier grades compared to harder grades. Scikit-learn's train_test_split method has a stratify parameter that enables one to do so. Doing so ensures that the grade proportions were the same in the train and test datasets.

Table 8 and *Figure 21* reports the results of the second validation. *Table 8* is the classification report of the validation obtained with the classification_report object of the mentioned library (Pedregosa et al., 2011). *Figure 21* is a heatmap of the confusion matrix of the validation obtained with the heatmap method of Seaborn (Waskom, 2021) , which is an open-source data visualization library of Python. The original labels of the dummy variables '0','1' and '2' of *Figure 21* are 'Intermediate', 'Advanced', and 'Elite' respectively.

Table 8: Classification Report - Problem Grade

	precision	recall	f1-score	support
0	0.76	0.81	0.78	72
1	0.61	0.63	0.62	52
2	0.83	0.42	0.56	12
accuracy			0.71	136
macro avg	0.74	0.62	0.65	136
weighted avg	0.71	0.71	0.70	136

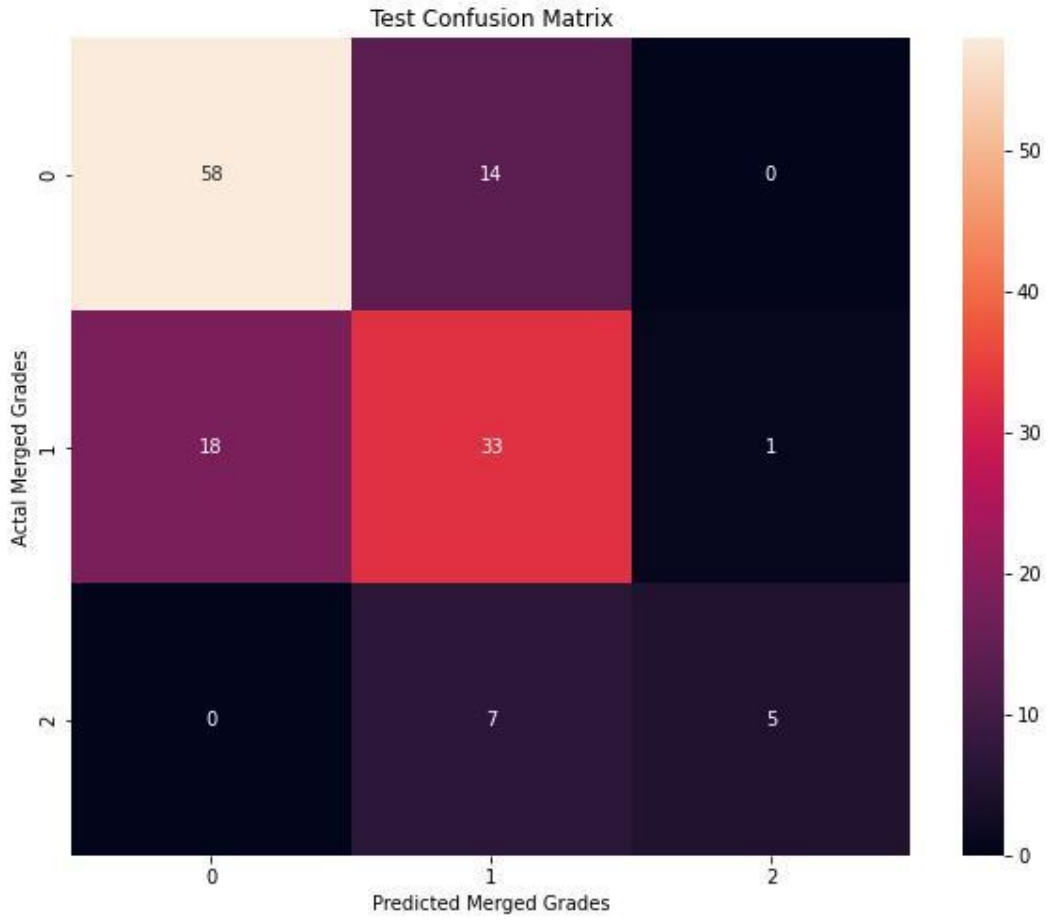

Figure 21: Confusion Matrix - Problem Grade

4.4. Validation of the Model for Move Difficulty

The validation method is the Pareto-smoothed Importance Sampling Leave-one-out Cross (PSIS-LOO) validation mentioned above. For this validation, there are 2456 individual moves in the dataset. PSIS-LOO is computed using ArviZ's loo method implemented in Python (see *Table 9*). All estimated shape k parameters are less than 0.5. This makes the estimations reliable, according to the paper mentioned above by Vehtari et al 2017).

Table 9: Pareto-smoothed Importance Sampling Leave-one-out Cross (PSIS-LOO) Validation – Move Difficulty

```
         Estimate       SE
elpd_loo -3231.66    17.19
p_loo        8.47        -
------

Pareto k diagnostic values:
                         Count   Pct.
(-Inf, 0.5]   (good)     2456  100.0%
 (0.5, 0.7]   (ok)          0    0.0%
   (0.7, 1]   (bad)         0    0.0%
   (1, Inf)   (very bad)    0    0.0%
```

4.5. Validation of the Planning Model

The results of the planning model validated with the suggested solutions in the MoonBoard application. The planning model found solutions to all thirty boulder problems in the problem dataset (see *Appendix C*). Each solution found by the model is compared with the suggested solution. This study used four performance metrics to validate the planning model: distance, cost, number of moves, and node accuracies. The predicted solution and the suggested solution to each problem were compared. Tables with validation results are in *Appendix D.2*. In this section, first, a general evaluation of the results will be made. Then, the results of each metric will be discussed separately. The relationship between them will be indicated.

The performance metrics are the following:

- the number of moves (see *Table 13* in *Appendix D.2.)*
- the distance (see *Table 14* in *Appendix D.2.)*
- the path cost (see *Table 15* in *Appendix D.2.)*
- the node accuracies (see *Table 16* in *Appendix D.2.)*

If we evaluate the model across the entire dataset, it found the correct hand nodes with an average accuracy of 62%. In addition, the number of hand moves made by the model is on par with the suggested solutions. Suggested solutions had an average of 6.26 hand moves, while the model made an average of 6.2 hand moves. However, the model performed poorly in foot moves. The model found the correct foot nodes with an average accuracy of 24%. Because the model made 62% fewer foot moves. Suggested solutions have an average of 6 foot moves, while the model has 2.26. So, the overall node accuracy of the model is 39%. The model's tendency to make fewer foot moves kept the path cost and distance of the solutions low. The average path cost of the solutions found by the model is 38% less than that of the suggested solutions (6.80 and 11.05). Also, the total distance of the solutions found by the model is 30% less than that of the suggested solutions (768.82 and 111.69).

In addition, the solutions that the planning model found used almost all available nodes in the problems. It showed similar performance with the suggested solutions. On average, the model used %88 of available nodes. The suggested solutions used %91 of all nodes. The model skips very few nodes. In benchmark problems, unnecessary nodes occur very rarely. In the problem where there is no unnecessary node, skipping a node indicates an efficiency problem. The model showed that it did not make this mistake by skipping very few nodes.

The first metric to be discussed is 'number of moves'. The three types of moves in each solution were calculated separately: moves that a hand leads, moves that include only foot changes, and all moves. For the comparison table of this metric, see *Table 13*. All but one predicted solution has fewer moves than the suggested solution. This is because predicted solutions involve fewer foot moves. Although this may seem like a good thing at first, it is actually due to the fact that the model is not able to catch foot move needs. It is the feet that allow the climber to transfer weight on the board. More footwork means breaking up the weight transfer work into smaller pieces. The model does not reflect this need well. This deficiency will show itself indirectly in other metrics as well.

The second metric is distance. The total distances of suggested and predicted solutions are calculated. *Table 14* shows the results. The distances of all predicted solutions except one are smaller than those of the suggested solutions. Again, this is because the planning model tends to take fewer foot moves to reach the goal hypernode. It is expected that this distance difference will not be observed in a model that can reflect that making more foot moves is more cost-efficient.

The third metric is cost. The total path costs and mean path costs are calculated. *Table 15* shows the results. The total path costs of all predicted solutions except one are smaller than those of the suggested solutions. Again, this is because the planning model tends to take fewer foot moves to reach the goal hypernode. This causes the total path cost to decrease. This effect is also seen when we focus on mean path costs. This time, all mean path costs except four are larger in predicted solutions. Although the total path cost is less

in predicted solutions, the average difficulty per move is higher. The model fails to capture the benefit of dividing the weight transfer work into smaller parts with more footwork.

The last metric is node accuracy. Node accuracies of each solution were calculated. The percentages of three node accuracies are calculated:

- The percentage of true nodes
- The percentage of hand nodes
- The percentage of foot nodes

Chapter 4.2. described the outputs of the planning model using an example. The same example will be used here in the context of validating the planning model. The shortest path for this example will be compared with the suggested path in the MoonBoard application in terms of the performance metrics above.

Figure 22 shows the result of the model on the left and the suggested path in the MoonBoard application on the right. Performance metrics for paths are shown in *Table 10*. NetworkX's related method used to draw the figure below (Hagberg et al., 2008).

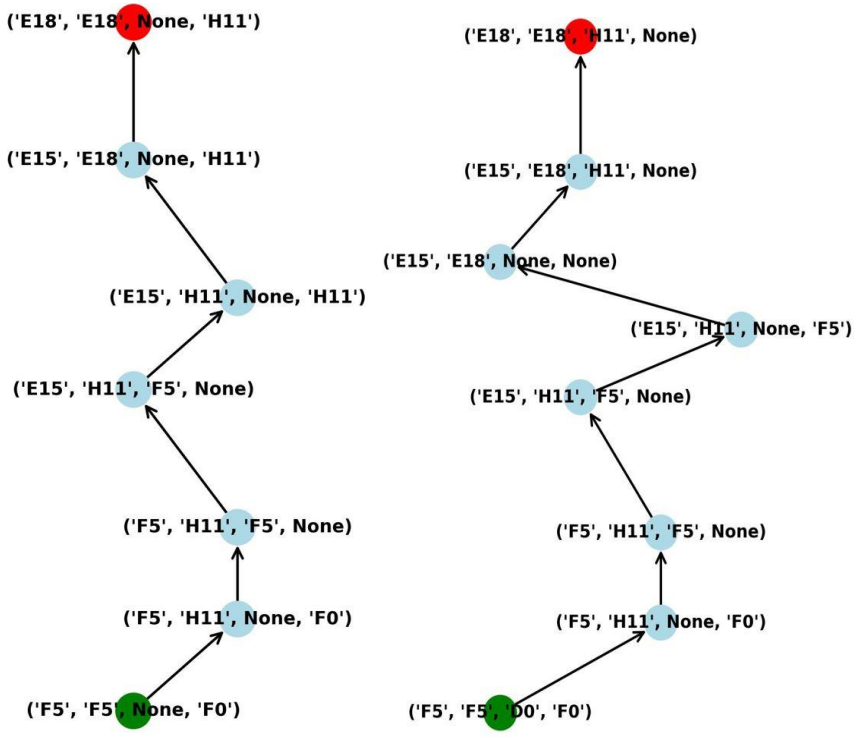

Figure 22: The Comparison of the Predicted Path (Left) and the Suggest Path (Right)

Table 10: The Comparison of the Performance Metrics of the paths in *Figure 22*

Problem ID: 50356					
# of the pred. moves	# of the sug. moves	# of hand moves in the pred. path	# of hand moves in the sugg. path	# of foot moves in the pred. path	# of foot moves in the sugg. path
6	7	4	4	2	3
total predicted distance			total suggested distance		
569.42			776.21		
path cost of the predicted path	path cost of the suggested path		mean predicted move cost		mean suggested move cost
5.38	7.66		0.9		1.09
percentage of accurate nodes		percentage of accurate hand nodes		percentage of accurate foot nodes	
0.64		1		0.43	

The number of moves metric contains three different numbers. The total number of all moves, the number of moves to the new hand node, and the number of moves to the new foot node. Each hyperedge in *Figure 22* is a move. In this problem, the number of predicted moves is one less than the number of suggested moves. In the predicted solution, 1 less foot move is made. The first two nodes of each hypernode are hand nodes, and the last two are foot nodes. In the new hypernode in the directed part of hyperedge, if the hand node has changed compared to the previous hypernode, it is a hand move. If only the foot node in the new hypernode in the directed part of the hyperedge has changed compared to the previous hypernode, it is a foot move. In this problem, the number of moves in both solutions is very close. The fact that the model solves the problem with such a close number of moves indicates that it is a human-like solution. The closeness of these move numbers implies success in other metrics as well. The planning model's performance across all 30 problem validation datasets will sometimes differ from this. A general evaluation of this metric across the entire dataset is provided will be made in detail in Chapter 5.1. The explanations for all the metrics in this example are as follows.

The distance metric shows the sum of the distances of the new node in each move to the midpoint of the nodes of the previous hypernodes. In this problem, the total distance in the predicted path is 569.42 units, while the total distance in the suggested path is 776.21. There are two reasons for the approximately 35% difference. The suggested path has more moves, and the moves in the suggested path are longer moves. The predicted path found here is more suitable for short climbers than the suggested path.

The path cost metric shows the total path cost of both paths. In this problem, the total path cost of the predicted path is 5.38 units, while the path cost of the suggested path is 7.66. One reason for this difference is that the number of moves is one more in the suggested path. The other reason is that moves with more distance have more costs. As mentioned above, in this problem, there was a total distance difference of about 35% from the predicted path in the suggested path.

The last metric is the node accuracy metric. As in 'number of moves', three different accuracy percentages were calculated: all nodes, hand nodes, and foot nodes. If the new node reached in any move of the predicted path is present in the suggested path, then this node is evaluated as true. The only condition is that the node must have been used with the same limb in both paths. In this problem, the overall percentage of accurate nodes is 0.64, the percentage of accurate hand nodes is 1, and the percentage of accurate foot nodes is 0.43. Although the model was good on both overall and hand accuracies, it could not find an accurate solution in foot moves. Poor performance in foot moves was also seen in other problems. This is discussed in Chapter 5.1.

The general discussion of all validation results of the planning model is in *Chapter 5.1*.

CHAPTER 5

CONCLUSION

5.1. Conclusion and Discussion

This study focussed on, climbing sport, specifically bouldering, as a decision-making problem and tackles this problem with a goal-based learning agent. This research has three research questions:

Q.1. Can we design an AI agent that shows human-like climbing behavior?

Q.2. Can this agent accurately guess the difficulty of boulder problems?

Q.3. Can this agent find an accurate solution for a novel boulder problem?

A goal-based agent was designed to answer these questions. This agent achieves two tasks: estimates the difficulty of a given problem and finds a solution to a given problem. This agent is supported by three different models: two separate learning models and one planning model. Learning models are developed in the Bayesian framework. On the other hand, the Planning model finds the shortest paths on Digraphs on which the Dijkstra algorithm is run.

This study is unique in its methodology and subject matter. Climbing has not yet been studied as a decision-making problem. The other novelty is that the planning task was not treated as just a distance optimization problem. It was based on data and experience while calculating the costs of all possible moves. The statistical model was developed in the Bayesian framework to calculate those costs. These costs were then used to find an optimized solution to a given bouldering problem.

A learning model was designed in the Bayesian framework to answer Q2. MoonBoard problems were collected from the MoonBoard web application to train this model. This dataset of 451 problems was collected by developing a web bot. The raw data collected was processed before training this model. In the raw data, only the holds that constitute a MoonBoard problem had location information on the board or node information, as it is technically called in this study. After processing the raw data, four different features were obtained for each route: distance, number of nodes, incut information, and percentage of

yellow holds. It is thought that there is a causal relationship between these four features and problem difficulty. See *Figure 23* in Appendix A.1 for the average values and standard errors of these features and the relationship between their difficulty levels.

By looking at the posterior distributions of the Bayesian model, it can be said that increases in distance and percentage of yellow holds increase the difficulty, while increases in the number of holds and incut have a facilitating effect (see *Table 5* in *Chapter 4.1.1.*). If the distance between the holds in a problem increases, the difficulty level of a problem increases. This effect is seen because reaching a hold farther away is more difficult. The problem becomes more difficult if the yellow hold density increases in a problem. The effect observed in the percentage of yellow holds feature is because the yellow holds are consistently smaller in size than the other holds on the board. If there are more holds in a problem, more options exist. According to the climber's position and the angle of the available holds, it is possible to choose the most suitable hold, and this has a facilitating effect on the problem. Regarding incut, an increase in the incut of a hold means that it is easier to grab that hold. It would be misleading to think there is a linear relationship between the hold's size and its incut. A small hold can have a larger incut than a large hold. This makes the small hold easier to hold. A problem consisting of holds with more incut is expected to be easier. The posterior distributions of the parameter of incut support this causal effect (see parameter 'bI' in *Table 5* in *Chapter 4.1.1.*).

The results of the model's estimation performance can be seen in *Figure 21* and *Table 8*. The model's accuracy was over 70% for all merged grade categories (for the explanation of the term merged grade, see *Chapter 2.1.*). However, recall values for categories of merged grades 1,2 and 3 were observed as 81, 63, and 42 for each category, respectively. We think that the reason for this thought is that the dataset is unbalanced. Four hundred fifty-one problems were divided into train-test sets with a rate of 30%. Since the dataset is unbalanced, there are fewer problems in more difficult grades. The train-test split was performed in a stratified way. This stratified fashion preserved the merged grade ratios in the train and test sets. As can be seen from the table, support decreased as the merged grade increased. While there are only 12 problems from the merged grade 2 category in the test data, there are 72 problems from the merged grade 1 category. This unbalance makes it difficult for the model to learn merged grade 2. As a result, the recall value of the model is low in the merged grade 2 category.

The following can be done to improve the model designed in the context of research question 2. recall that four features were used in the model designed for problem difficulty, and three different merged grade category predictions were produced. Scale can be divided into more categories by increasing the merged grade category. For this, the number of problems in the dataset must be increased. Another improvement can be made as follows. This model has four different input features. The number of these features can be increased. For example, a better model may include the features: the angles of the holds and the slopeness of the holds.

Q1 and Q3 are related. Two models were designed to these questions. One model is a learning model in the Bayesian framework. The other model is a planning model that finds solutions to given boulder problems. The learning model supports the planning model. The learning model is trained to estimate the difficulty of a move. The posterior distributions of the parameters of this model are used in the planning model. The path costs of the planning model are calculated by these parameters.

Table 16 shows the results and averages. The model performed well in predicting hand moves. The model showed an average of 62 percent accuracy over 30 problems on hand nodes. However, the model needs to improve in predicting foot moves. The model showed an average of 25 percent accuracy on foot nodes over 30 problems. The model showed an average of 36 percent accuracy in its overall node prediction. The model's success in the hand move was observed at the highest level in 3 problems. In 3 problems, the model found solutions containing the exact hand sequences from the suggested solutions. But in general, the problem mentioned above has also manifested itself here. The model's tendency to do less footwork resulted in a lower overall node accuracy. The average path cost of the solutions found by the model is 38% less than that of the suggested solutions (6.80 and 11.05). Also, the total distance of the solutions found by the model is 30% less than that of the suggested solutions (768.82 and 111.69).

In addition, the solutions that the planning model found used almost all available nodes in the problems. It showed similar performance with the suggested solutions. On average, the model used %88 of available nodes. The suggested solutions used %91 of all nodes. The model skips very few nodes. In benchmark problems, unnecessary nodes occur very rarely. In the problem where there is no unnecessary node, skipping a node indicates an efficiency problem. The model showed that it did not make this mistake by skipping very few nodes.

In terms of Q1, the model performed as expected in hand moves. An athlete can implement hand movement decisions of the model. The model does not produce positions that would exceed the physical limits of any human being. 62% hand node accuracy also supports this. However, the model's tendency to make an average of about 60% fewer foot moves reduces the quality of the solutions found by the model to be applied by an athlete. The tendency to make few foot moves pushes the model to make difficult long dynamic moves. For example, there are problems where the model prefers to approach the goal state with one powerful (long, dynamic move) move instead of making three more foot moves in a less powerful (static) style.

The model's tendency to make fewer foot moves causes it to make a mistake seen in amateur climbers. Amateur climbers have difficulty maintaining foot contact with the wall while climbing on a steep-angled board like MoonBoard. Both feet lose contact with the wall and make contact again while climbing. They cannot keep the tension in their body, and their feet lose contact with the wall from time to time. This is encountered in the solutions developed by the model. Due to the nature of MoonBoard problems, it is normal to a certain point that the feet are not in contact with the wall. There are many problems

to practice dynamic, powerful moves in MoonBoard problems. However, apart from such problems, the model also cuts off the contact of the feet with the wall.

The solutions that the planning model generated are validated with the suggested solutions from MoonBoard mobile App. Although suggested solutions are successful in the sense that they are legitimate and sequences that a human can follow, they may be sub-optimal. Suggested solutions may have skipped holds. It may include some redundant foot moves or hand moves. Fitter and more experienced climbers may suggest another solution to the same problem. Not only her fitness or anthropometric characteristics but also the flexibility of a climber increases the number of possible moves. A flexible climber may increase the number of contact points to the board and thus distribute the load on their limbs more evenly.

There are certain limitations of the planning model. First, the model does not consider the angle of the holds. In some cases, this deficiency leads to more difficult solutions. The planning model can find more accurate solutions if the hold angles information can be included in the model that estimates the move difficulty. Second, the beta dataset consists of only successful moves. There are no failed moves in the dataset. When trying to find a solution to a problem, the climber makes unsuccessful attempts if he is inexperienced. These unsuccessful attempts may be due to physical incapacity or decisions that will put the climber in impossible positions. The absence of such wrong moves in the dataset makes it difficult to distinguish what is feasible and what is not. Third, the model makes some moves that require a lot of strength and power. Although these moves are within the limits of the solution dataset in terms of body positions, some moves that the model finds require a lot of strength and power. The model does not include the data of the force exerted by the climber on the holds. For this reason, the model sometimes makes moves that require a lot of strength and power.

5.2. Suggestions for Further Studies

In future studies, Generalized Additive Models (GAM) and Gaussian Process models can be used for the effect of distance in models that estimate the difficulty of a move.

The planning model found solutions from the starting position to the ending position in one go with the shortest path algorithm. Alternatively, solutions can be developed dynamically. The task can be handled by dividing it into parts, with an algorithm that develops a solution for each part. Approaching this way can prevent possible hold skips.

There has been a significant increase in the number and variety of standardized training boards in recent years. Personalized beta and problem suggestion features can be added to these boards' mobile or web applications. These features will take as input the person's physical characteristics, limits, and problems he has climbed before. By evaluating these inputs, the application will be able to suggest a personalized beta for the given problem or

make problem suggestions according to the topic of the training that day. Thanks to these features, the training efficiency of the climbers can be increased.

The need for automated management of problems was realized while creating the dataset of this research. The main reason is that this training board's problem dataset is very large, namely more than 50 thousand. There is a large community that expands this problem pool. However, there is a limited moderator's team that certifies that the difficulty level of problems is given accurately by route setters and that the routes are of good quality. It is not possible for a limited team to manage such a large problem pool manually. Therefore, an automated moderation can be added to the application. Two different models can evaluate whether the difficulty level proposed by the community is accurate and whether the route is of good quality.

Sensor data can be included in the dataset to make the dataset more detailed and the beta-finding planning model more comprehensive. Three different types of sensors can be used to obtain this sensor data:

-IMU: a specific type of sensor that measures angular rate, and force.

-EMG: measures small electrical signals generated by your muscles when you move them.

-LOADCELL: converts force into measurable electrical output.

With the help of these sensor data, the difficulty of the climber's moves on the board can be better determined and modeled. In fact, with these data, the physical skills of the climbers can be profiled, and problem suggestions can be made to improve their deficiencies.

Climbing style and physical features can be added to the planning model. When creating a Network for a problem, possible body positions (Hypernodes) and moves (Hyperedges) can be manipulated according to the climber's desired climbing style and physical limits. For a more dynamic climbing style, hyperedges with multiple limb replacements are defined, while for more static, conservative climbing styles, hyperedges involving multiple limb replacements may be restricted. For climbers with more flexibility, body positions with the feet closer to the hands can be included in the network. Such hypernodes may be restricted for climbers with little flexibility.

The study validated the planning model with suggested solutions in the application. Further analysis can do two things for a more comprehensive assessment of the solutions developed by this model. First, experienced climbers can evaluate solutions through a survey. Second, the climbers can try the solutions on the board. After trying them on the board, climbers can evaluate the solutions through a survey. In this way, it can be more accurately evaluated whether the solutions produced are human-like, efficient or not, and possible.

REFERENCES

Çelik, R. (2022). *Model-Based Difficulty Estimation of Indoor Bouldering Routes*. Utrecht University.

Climbing Magazine. (2022, December 1). *Outdoor boulder problem*. Https://Www.Climbing.Com/Photos/Re-Gram-9-Dyno-Photos/.

Dobles, A., Sarmiento, J. C., & Satterthwaite, P. (2017). *Machine Learning Methods for Climbing Route Classification*. www.moonboard.com

Duh, Y.-S., & Chang, R. (2021). *Recurrent Neural Network for MoonBoard Climbing Route Classification and Generation*. http://arxiv.org/abs/2102.01788

Flanagan, D. (Rock climber). (2013). *Bouldering Essentials : The Complete Guide to Bouldering*. Three Rock.

Hagberg, A. A. , Schult, D. A. , & Swart, P. J. (2008). *Exploring Network Structure, Dynamics, and Function using NetworkX*. http://conference.scipy.org/proceedings/SciPy2008/paper_2

Hörst, E. J. (2008). *Training for climbing : the definitive guide to improving your performance*. Falcon Guides.

Jahns, M. (2021, June 30). *7 Facts about Climbing Queen Janja Garnbret*. Https://Www.Ispo.Com/En/People/Janja-Garnbret-7-Facts-about-Slovenian-Climbing-Queen.

Kruschke, J. K. (2021). Bayesian Analysis Reporting Guidelines. In *Nature Human Behaviour* (Vol. 5, Issue 10, pp. 1282–1291). Nature Research. https://doi.org/10.1038/s41562-021-01177-7

Moon Climbing. (2022a, January 9). *moonboard problem - crane direct*. Https://Www.Moonboard.Com/Problems/View/50356/the-Crane-Direct.

Moon Climbing. (2022b, November 29). *MoonBoard 2016 Layout*. Https://Www.Moonboard.Com/HoldSetups/Index.

Moon Climbing. (2022c, November 29). *MoonBoard Problem - ID: 20249*. Https://Www.Moonboard.Com/Problems/View/20249/Captain-Fitzroy.

Moon Climbing. (2022d, November 29). *MoonBoard Problem - ID: 431907*. Https://Www.Moonboard.Com/Problems/View/431907/%EC%9D%B4%EA%B1%B0-%EA%B9%A8%EB%A9%B4%EC%8C%89%EA%B3%A0%EC%88%98.

Moon Climbing. (2022e, November 29). *MoonBoards World Map*. Https://Www.Moonboard.Com/MoonBoard/Overview.

Moon Climbing. (2022f, December 14). *MoonBoard Problem - ID: 26620*. Https://Www.Moonboard.Com/Problems/View/26620/Jens-a-Watching.

Moon Climbing. (2022g, December 20). *MoonBoard Problem - ID:262846*. Https://Www.Moonboard.Com/Problems/View/262846/Bitter.

Nick, D., Couceiro, J., Fryer, S., & Dickson, T. (2011). Reporting Climbing Grades and Grouping Categories for Rock Climbing. *Isokinetics and Exercise Science*. https://doi.org/10.3233/IES20110424

Norvig, P., & Russell, S. J. (2020). *Artificial Intelligence a Modern Approach* (Fourth Edition). Prentice Hall.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Second Edition). Morgan Kaufman Publishers, Inc.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Varoquaux, G., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in. *Journal of Machine Learning Research*, *12*, 2925–2830. http://scikit-learn.sourceforge.net.

Salvatier, J., Wiecki, T. v., & Fonnesbeck, C. (2016). Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, *2*, e55. https://doi.org/10.7717/peerj-cs.55

Seal, D., & Seal, R. (2022). *Optimum Route Computation in a Chaotic Artificial Climbing Wall* (pp. 671–680). https://doi.org/10.1007/978-981-16-5987-4_68

Stapel, F. (2020). *A Heuristic Approach to Indoor Rock Climbing Route Generation*.

Tai, C.-H., Wu, A., & Hinojosa, R. (2020). *Graph Neural Networks in Classifying Rock Climbing Difficulties*. https://github.com/gestalt-howard/moonGen

The International Olympic Committee. (2016, August 3). *Olympics - Sports Climbing*. Https://Olympics.Com/Ioc/News/Ioc-Approves-Five-New-Sports-for-Olympic-Games-Tokyo-2020.

Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, *27*(5), 1413–1432. https://doi.org/10.1007/s11222-016-9696-4

Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., & Bürkner, P.-C. (2019). *Rank-normalization, folding, and localization: An improved Rhat for assessing convergence of MCMC*. https://doi.org/10.1214/20-BA1221

Vereide, V., Andersen, V., Hermans, E., Kalland, J., Saeterbakken, A. H., & Stien, N. (2022). Differences in Upper-Body Peak Force and Rate of Force Development in Male Intermediate, Advanced, and Elite Sport Climbers. *Frontiers in Sports and Active Living*, *4*. https://doi.org/10.3389/fspor.2022.888061

Waskom, M. (2021). Seaborn: Statistical Data Visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Yet, B., Constantinou, A., Fenton, N., Neil, M., Luedeling, E., & Shepherd, K. (2016). A Bayesian network framework for project cost, benefit and risk analysis with an agricultural development case study. *Expert Systems with Applications*, *60*, 141–155. https://doi.org/10.1016/j.eswa.2016.05.005

APPENDICES

APPENDIX A

Appendix A.1. The Plots of the Variables of the Bayesian Model for Problem Grade

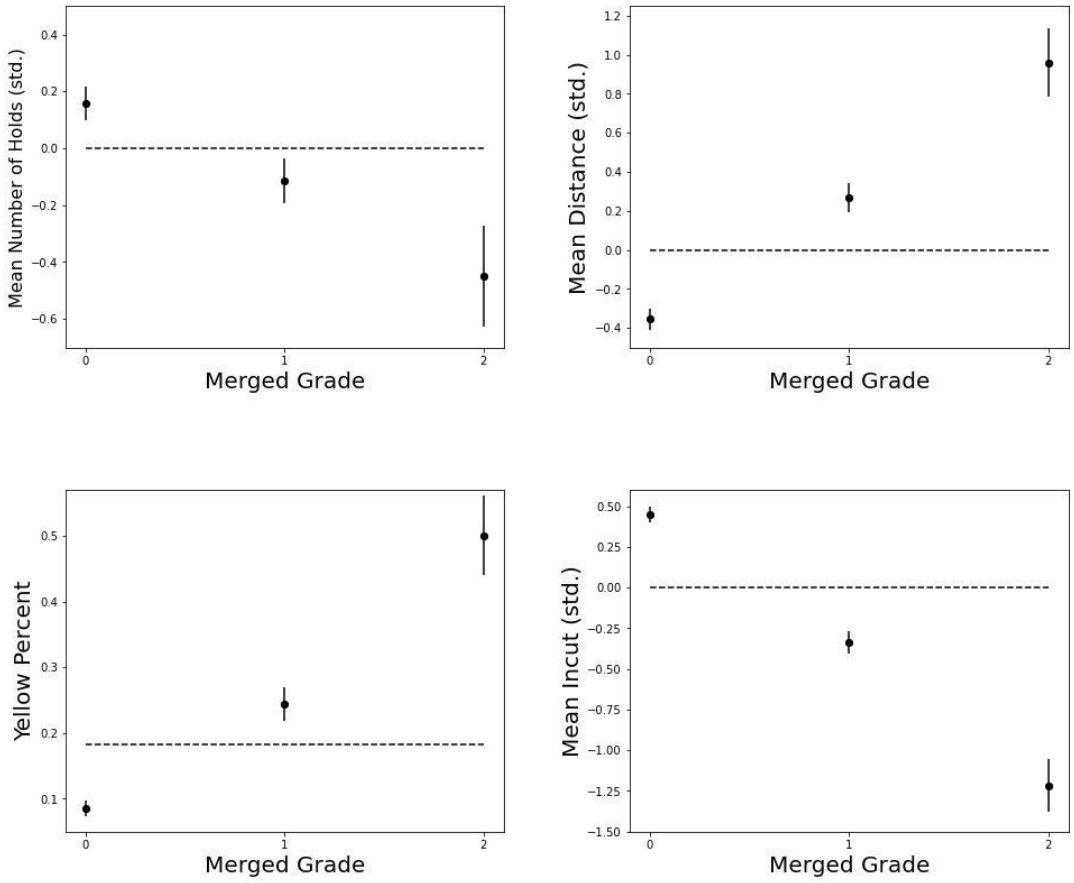

Figure 23: The Plots of Each Variable with The Average Values and Standard Error Bars for each Merged Grade

Appendix A.2. The Plots of the Variables of the Bayesian Model for Move Difficulty

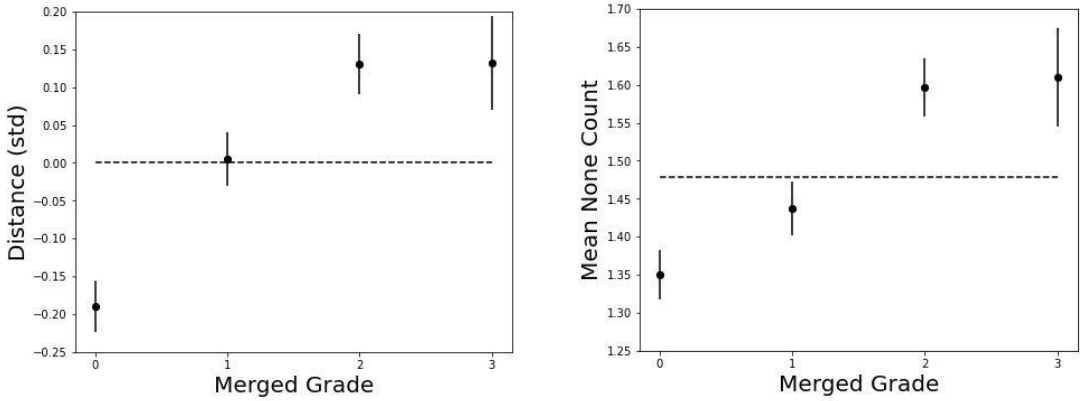

Figure 24: The Plots of Each Variable Standardized or Mean Values and Standard Error Bars for each Merged Grade

APPENDIX B

MoonBoard Holds Classification

Table 11: MoonBoard Hold Classification

name	color	incut	match	name	color	incut	match
A5	white	0	TRUE	F12	white	0	FALSE
A9	black	1	FALSE	F13	black	1	FALSE
A10	yellow	0	FALSE	F14	white	2	TRUE
A11	yellow	0	FALSE	F15	yellow	1	FALSE
A12	white	0	FALSE	F16	white	0	FALSE
A13	yellow	1	FALSE	G2	white	2	TRUE
A14	black	2	FALSE	G4	black	1	TRUE
A15	white	1	FALSE	G6	black	2	TRUE
A16	yellow	1	FALSE	G7	yellow	0	FALSE
A18	black	1	TRUE	G8	white	1	TRUE
B3	white	2	TRUE	G9	black	2	FALSE
B4	yellow	1	TRUE	G10	black	1	FALSE
B6	black	1	TRUE	G11	yellow	1	FALSE
B7	yellow	0	FALSE	G12	white	1	FALSE
B8	white	1	FALSE	G13	white	2	TRUE
B9	white	1	FALSE	G14	black	1	TRUE
B10	black	2	FALSE	G15	black	2	FALSE
B11	white	2	TRUE	G16	yellow	0	FALSE
B12	black	0	FALSE	G17	black	1	FALSE
B13	white	2	TRUE	G18	white	2	TRUE
B15	black	0	FALSE	H5	black	2	TRUE
B16	white	1	FALSE	H7	yellow	0	FALSE
B18	white	2	TRUE	H8	black	1	FALSE
C5	black	1	TRUE	H9	yellow	2	FALSE
C6	white	0	TRUE	H10	black	2	TRUE
C7	yellow	2	TRUE	H11	white	2	TRUE
C8	black	1	FALSE	H12	black	0	FALSE
C9	yellow	0	FALSE	H13	black	2	FALSE
C10	white	2	FALSE	H14	yellow	2	FALSE
C11	white	0	FALSE	H15	yellow	0	FALSE
C12	yellow	1	FALSE	H16	black	1	FALSE
C13	black	2	TRUE	H18	yellow	1	TRUE
C14	white	1	FALSE	I4	white	1	TRUE

C15	yellow	2	FALSE	I5	yellow	2	TRUE
C16	black	1	FALSE	I6	white	0	TRUE
C18	yellow	1	TRUE	I7	black	0	FALSE
D3	yellow	1	TRUE	I8	yellow	0	FALSE
D5	white	0	TRUE	I9	black	2	TRUE
D6	yellow	0	TRUE	I10	black	1	TRUE
D7	black	0	FALSE	I11	white	1	FALSE
D8	yellow	0	FALSE	I12	yellow	2	FALSE
D9	white	1	FALSE	I13	white	0	FALSE
D10	yellow	2	FALSE	I14	black	2	TRUE
D11	black	2	FALSE	I15	black	1	FALSE
D12	white	2	TRUE	I16	white	1	TRUE
D13	yellow	2	FALSE	I18	black	2	TRUE
D14	white	0	FALSE	J2	white	2	TRUE
D15	black	2	TRUE	J5	black	2	TRUE
D16	yellow	0	FALSE	J6	white	0	TRUE
D17	white	1	TRUE	J7	black	1	FALSE
D18	black	2	TRUE	J8	white	2	TRUE
E6	black	1	TRUE	J9	white	2	FALSE
E7	white	1	TRUE	J10	white	1	FALSE
E8	black	1	TRUE	J11	yellow	0	FALSE
E9	black	0	FALSE	J12	black	0	FALSE
E10	white	2	FALSE	J13	black	2	FALSE
E11	white	1	FALSE	J14	yellow	1	FALSE
E12	black	2	FALSE	J16	black	1	FALSE
E13	white	1	FALSE	K5	white	2	TRUE
E14	black	2	FALSE	K6	yellow	0	TRUE
E15	white	2	TRUE	K7	yellow	0	FALSE
E16	black	2	TRUE	K8	yellow	0	FALSE
E18	white	2	TRUE	K9	black	2	FALSE
F5	white	2	TRUE	K10	yellow	0	FALSE
F6	yellow	0	TRUE	K11	white	2	FALSE
F7	white	1	FALSE	K12	yellow	0	FALSE
F8	yellow	0	FALSE	K13	yellow	0	FALSE
F9	yellow	1	FALSE	K14	black	2	FALSE
F10	white	1	FALSE	K16	black	1	TRUE
F11	black	1	FALSE	K18	white	2	TRUE

APPENDIX C

Table 12: Problem Dataset of the Network

problem id	font. grade	merged grade	number of nodes	start nodes	intermediate nodes	end nodes	nodes
82224	8B	3	4	J5	I9, E14	H18	J5, I9, E14, H18
62575	8A+	3	6	K6	D10, H14, A16, K7	A18	K6, K7, D10, H14, A16, A18
23184	8A	3	8	B4, G7	H7, K8, K10, K13, G16	H18	B4, G7, H7, K8, K10, K13, G16, H18
241045	8A+	3	8	D3, B4	G7, I8, K12, H14, H15	C18	D3, B4, G7, I8, K12, H14, H15, C18
62560	8A	3	5	D6	F9, H9, F15	C18	C18, D6, F15, F9, H9
350783	8A	3	8	G2, E6	J6, F8, J12, F15, F16	A18	E6, G2, J6, F8, J12, F15, F16, A18
371579	8A	3	8	D3, E6	I8, A12, G14, G16, A16	A18	D3, E6, I8, G14, G16, A16, A12, A18
44578	7C+	2	5	A5	C14, E7, H11	G18	A5, C14, E7, G18, H11
43010	7C+	2	6	F6, K6	A10, G11, C15	C18	F6, K6, G11, A10, C15, C18
45211	7C	2	8	H5, D6	E9, E11, J11, J14, I15	H18	D6, H5, E9, E11, J11, J14, I15, H18
20698	7B+	2	6	A5	A9, F11, F13, E16	I18	A5, A9, F11, F13, E16, I18
48939	7B+	2	6	B6, F6	F11, I14, G15	D18	B6, F6, F11, G15, I14, D18
430586	7C+	2	7	B3	D8, C11, K10, I13, H14	K18	B3, D8, C11, H14, I13, K18, K10
227999	7C	2	7	J2, I6	G10, D10, F15, C16	I18	C16, D10, F15, G10, I6, I18, J2
39709	7B+	2	5	G4	C7, H7, E13	C18	C7, C18, E13, G4, H7
345039	7B	1	8	J5	E6, F7, A9, E12, F13, I15	H18	J5, E6, F7, A9, E12, F13, I15, H18
292263	7B	1	6	E6	A9, C9, A13, B15	G18	A9, A13, B15, C9, E6, G18
48723	7A+	1	7	A5	H5, E8, E9, G13, E16	D18	A5, D18, E8, E9, E16, G13, H5
24681	7A+	1	6	I5	H9, D10, H14, D16	H18	D16, D10, H18, H14, H9, I5

313875	7A	1	6	K5	J8, F11, G14, E16	G18	K5, J8, F11, G14, E16, G18
50356	7A	1	4	F5	H11, E15	E18	E15, E18, F5, H11
118171	7B	1	11	J2, K5	A5, E7, F7, J8, B11, C11, F14, G14	K18	A5, B11, C11, E7, F14, F7, G14, J8, J2, K5, K18
202064	7B	1	7	C5	H5, E9, F11, H12, G15	I18	C5, E9, H5, F11, H12, G15, I18
172010	6C	0	8	F5	E7, A9, K12, E12, F14, G17	D18	A9, D18, E7, E12, F14, F5, G17, K12
268943	6C	0	7	A5, D6	E8, D10, H11, F14	G18	A5, D6, E8, D10, H11, F14, G18
134795	6B+	0	9	F5, G2	D12, D9, E15, G8, H14, H11	E18	D12, D9, E15, E18, F5, G2, G8, H14, H11
96879	6B+	0	6	F5	E8, C10, G14, F15	I18	F5, E8, C10, G14, F15, I18
337544	6C	0	10	J5, K5	G6, B7, H8, E10, E12, A14, E15	G18	K5, J5, G6, H8, B7, E10, E12, A14, E15, G18
20038	6C+	0	5	I5	H9, I12, F15	H18	F15, H18, H9, I5, I12
23261	6C+	0	6	F5	H10, D11, E15, B16	E18	B16, D15, D11, E18, F5, H10

APPENDIX D

Appendix D.1. The results of the planning model

- Problem ID: 82224, the path: ('J5', 'J5', 'F0', 'J-1'), ('I9', 'J5', None, 'J-1'), ('I9', 'I9', None, 'J-1'), ('I9', 'I9', 'J5', None), ('E14', 'I9', 'J5', None), ('E14', 'I9', None, 'I9'), ('E14', 'H18', None, 'I9'), ('H18', 'H18', None, 'I9').

- Problem ID: 62575, the path: ('K6', 'K6', 'F0', 'J0'), ('K6', 'K7', 'F0', None), ('D10', 'K7', None, None), ('D10', 'K7', None, 'K6'), ('D10', 'H14', None, None), ('D10', 'H14', 'D10', None), ('A16', 'H14', None, None), ('A16', 'H14', None, 'D10'), ('A16', 'A18', None, 'D10'), ('A18', 'A18', None, 'D10')

- Problem ID: 241045, the path: [('B4', 'D3', 'H-1', 'J-1'), ('D3', 'D3', 'H-1', 'J-1'), ('D3', 'I8', 'H-1', 'J-1'), ('H14', 'I8', None, None), ('H14', 'I8', None, 'I8'), ('H14', 'H15', None, None), ('H14', 'H15', 'I8', None), ('C18', 'H15', None, None), ('C18', 'H15', 'H14', None), ('C18', 'C18', 'H14', None)]

- Problem ID: 23184, the path: [('B4', 'G7', 'F-1', 'F0'), ('B4', 'H7', 'F-1', 'F0'), ('G7', 'H7', None, 'F0'), ('G7', 'K8', None, 'F0'), ('G7', 'K10', None, 'F0'), ('G7', 'K10', 'G7', None), ('K8', 'K10', 'G7', None), ('K13', 'K10', 'G7', None), ('G16', 'K10', None, None), ('G16', 'K10', None, 'K10'), ('G16', 'K13', None, 'K10'), ('G16', 'H18', None, 'K10'), ('H18', 'H18', None, 'K10')]

- Problem ID: 62560, the path: [('D6', 'D6', 'D0', 'H0'), ('D6', 'H9', 'D0', None), ('F9', 'H9', 'D0', None), ('F9', 'H9', None, 'D6'), ('F9', 'F15', None, None), ('F9', 'F15', 'F9', None), ('C18', 'F15', 'F9', None), ('C18', 'C18', 'F9', None)]

- Problem ID: 350783, the path: [('E6', 'G2', 'D0', 'J0'), ('F8', 'G2', None, 'J0'), ('F8', 'J12', None, None), ('F8', 'J12', 'F8', None), ('F15', 'J12', 'F8', None), ('F15', 'J12', 'F8', 'J12'), ('F15', 'F16', 'F8', None), ('A18', 'F16', None, None), ('A18', 'F16', 'F15', None), ('A18', 'A18', 'F15', None)]

- Problem ID: 371579, the path: [('D3', 'E6', 'B0', None), ('A12', 'E6', None, None), ('A12', 'E6', None, 'E6'), ('A12', 'G16', None, None), ('A12', 'G16', 'A12', None), ('A18', 'G16', 'A12', None), ('A18', 'A18', 'A12', None)]

- Problem ID: 44578, the path: [('A5', 'A5', 'D0', 'F0'), ('A5', 'E7', 'D0', 'F0'), ('E7', 'E7', None, 'F0'), ('E7', 'H11', None, 'F0'), ('E7', 'H11', 'E7', None), ('C14', 'H11', 'E7', None), ('C14', 'H11', None, 'H11'), ('C14', 'G18', None, 'H11'), ('G18', 'G18', None, 'H11')]

- Problem ID: 43010, the path: [('F6', 'K6', None, 'F0'), ('F6', 'G11', None, 'F0'), ('F6', 'G11', 'F6', None), ('C15', 'G11', 'F6', None), ('C15', 'G11', None, 'G11'), ('C15', 'C18', None, 'G11'), ('C18', 'C18', None, 'G11')]

- Problem ID: 430586, the path: [('B3', 'B3', None, 'B-1'), ('B3', 'D8', None, 'B-1'), ('B3', 'D8', 'B3', None), ('C11', 'D8', None, None), ('C11', 'D8', None, 'B3'), ('C11', 'I13', None, None), ('C11', 'I13', 'D8', None), ('H14', 'I13', None, None), ('H14', 'I13', None, 'K10'), ('H14', 'K18', None, 'K10'), ('K18', 'K18', None, 'K10')]

- Problem ID: 45211, the path: [('D6', 'H5', 'D0', 'F0'), ('D6', 'E9', 'D0', None), ('E11', 'E9', 'D0', None), ('E11', 'E9', None, 'D6'), ('E11', 'J11', None, 'D6'), ('E11', 'J14', None, 'D6'), ('E11', 'J14', 'E11', None), ('I15', 'J14', 'E11', None), ('H18', 'J14', 'E11', None), ('H18', 'H18', 'E11', None)]

- Problem ID: 227999, the path: [('I6', 'J2', 'D0', 'F0'), ('I6', 'I6', 'D0', 'F0'), ('D10', 'I6', None, 'F0'), ('D10', 'G10', None, 'F0'), ('D10', 'G10', 'I6', None), ('F15', 'G10', 'I6', None), ('F15', 'G10', 'I6', 'D10'), ('C16', 'G10', None, 'D10'), ('C16', 'F15', None, 'D10'), ('C16', 'I18', None, 'D10'), ('I18', 'I18', None, 'D10')]

- Problem ID: 20698, the path: [('A5', 'A5', 'B0', 'F0'), ('A9', 'A5', None, 'F0'), ('A9', 'F11', None, None), ('A9', 'F11', 'A5', None), ('F13', 'F11', 'A5', None), ('E16', 'F11', None, None), ('E16', 'F11', None, 'F11'), ('E16', 'F13', None, 'F11'), ('E16', 'I18', None, 'F11'), ('I18', 'I18', None, 'F11')]

- Problem ID: 48939, the path: [('B6', 'F6', 'F0', None), ('F11', 'F6', 'F0', None), ('F11', 'F6', None, 'F6'), ('F11', 'I14', None, None), ('F11', 'I14', 'F6', None), ('G15', 'I14', 'F6', None), ('G15', 'I14', 'F11', None), ('D18', 'I14', 'F11', None), ('D18', 'D18', 'F11', None)]

- Problem ID: 39709, the path: [('G4', 'G4', 'D0', 'J0'), ('C7', 'G4', None, 'J0'), ('C7', 'G4', None, 'G4'), ('C7', 'E13', None, None), ('C7', 'E13', 'C7', None), ('C18', 'E13', 'C7', None), ('C18', 'E13', None, 'E13'), ('C18', 'C18', None, 'E13')]

- Problem ID: 345039, the path: [('J5', 'J5', 'F0', 'F-1'), ('E6', 'J5', 'F0', 'F-1'), ('E6', 'F7', 'F0', None), ('A9', 'F7', 'F0', None), ('A9', 'F7', None, 'E6'), ('A9', 'E12', None, 'E6'), ('A9', 'E12', 'A9', 'E6'), ('A9', 'F13', 'A9', 'E6'), ('E12', 'F13', 'A9', 'E6'), ('E12', 'I15', 'A9', None), ('E12', 'I15', 'E12', None), ('F13', 'I15', 'E12', None), ('H18', 'I15', 'E12', None), ('H18', 'H18', 'E12', None)]

- Problem ID: 292263, the path: [('E6', 'E6', 'B0', 'F0'), ('A9', 'E6', None, 'F0'), ('A9', 'C9', None, 'F0'), ('A9', 'C9', 'E6', None), ('B15', 'C9', None, None), ('B15', 'C9', None, 'C9'), ('B15', 'G18', None, 'C9'), ('G18', 'G18', None, 'C9')]

- Problem ID: 118171, the path: [('J2', 'K5', 'F0', 'J0'), ('J8', 'K5', None, 'J0'), ('J8', 'J8', None, 'J0'), ('J8', 'J8', 'K5', None), ('G14', 'J8', 'K5', None), ('G14', 'J8', None, 'J8'), ('G14', 'G14', None, 'J8'), ('G14', 'K18', None, 'J8'), ('K18', 'K18', None, 'J8')]

- Problem ID: 202064, the path: [('C5', 'C5', None, 'D0'), ('C5', 'E9', None, 'D0'), ('C5', 'F11', None, 'D0'), ('C5', 'F11', 'C5', 'D0'), ('E9', 'F11', 'C5', 'D0'), ('E9', 'H12', 'C5', None), ('F11', 'H12', 'C5', None), ('G15', 'H12', 'C5', None), ('G15', 'H12', None, 'E9'), ('G15', 'I18', None, 'E9'), ('I18', 'I18', None, 'E9')]

- Problem ID: 48723, the path: [('A5', 'A5', 'B0', 'F0'), ('A5', 'E8', 'B0', None), ('E8', 'E8', 'B0', None), ('E8', 'E8', None, 'A5'), ('E8', 'G13', None, 'A5'), ('E8', 'G13', 'E9', 'A5'), ('G13', 'G13', 'E9', None), ('D18', 'G13', 'E9', None), ('D18', 'D18', 'E9', None)]

- Problem ID: 24681, the path: [('I5', 'I5', 'D0', 'J0'), ('H9', 'I5', 'D0', 'J0'), ('H9', 'H14', None, None), ('H9', 'H14', 'H9', None), ('H18', 'H14', 'H9', None), ('H18', 'H18', 'H9', None)]

- Problem ID: 313875, the path: [('K5', 'K5', 'J0', 'J-1'), ('J8', 'K5', 'J0', 'J-1'), ('J8', 'J8', 'J0', None), ('J8', 'J8', 'J0', 'K5'), ('G14', 'J8', None, 'K5'), ('G14', 'G14', None, None), ('G14', 'G14', 'F11', None), ('G18', 'G14', 'F11', None), ('G18', 'G18', 'F11', None)]

- Problem ID: 50356, the path: [('F5', 'F5', None, 'F0'), ('F5', 'H11', None, 'F0'), ('F5', 'H11', 'F5', None), ('E15', 'H11', 'F5', None), ('E15', 'H11', None, 'H11'), ('E15', 'E18', None, 'H11'), ('E18', 'E18', None, 'H11')]

- Problem ID: 20038, the path: [('I5', 'I5', 'D0', 'J0'), ('H9', 'I5', None, 'J0'), ('H9', 'I12', None, None), ('H9', 'I12', 'I5', None), ('F15', 'I12', 'I5', None), ('F15', 'I12', None, 'H9'), ('F15', 'H18', None, 'H9'), ('H18', 'H18', None, 'H9')]

- Problem ID: 172010, the path: [('F5', 'F5', 'B0', None), ('A9', 'F5', 'B0', None), ('A9', 'F5', None, 'F5'), ('A9', 'E7', None, 'F5'), ('A9', 'E12', None, 'F5'), ('A9', 'F14', None, None), ('A9', 'F14', 'A9', None), ('F14', 'F14', 'A9', None), ('D18', 'F14', 'A9', None), ('D18', 'D18', 'A9', None)]

- Problem ID: 268943, the path: [('A5', 'D6', None, 'D0'), ('A5', 'E8', None, 'D0'), ('A5', 'E8', 'A5', 'D0'), ('D6', 'E8', 'A5', 'D0'), ('D10', 'E8', 'A5', 'D0'), ('D10', 'H11', 'A5', None), ('F14', 'H11', 'A5', None), ('F14', 'H11', None, 'D10'), ('F14', 'G18', None, 'D10'), ('G18', 'G18', None, 'D10')]

- Problem ID:337544, the path: [('J5', 'K5', 'D0', 'F0'), ('G6', 'K5', 'D0', 'F0'), ('G6', 'G6', 'D0', None), ('E10', 'G6', 'D0', None), ('E12', 'G6', None, None), ('E12', 'G6', None, 'G6'), ('E12', 'E10', None, 'G6'), ('E12', 'E15', None, 'G6'), ('E12', 'E15', 'E10',

'G6'), ('E15', 'E15', 'E10', 'G6'), ('E15', 'G18', 'E10', None), ('G18', 'G18', 'E10', None)]

- Problem ID:134795, the path: [('F5', 'G2', 'F-1', 'J0'), ('F5', 'G8', 'F-1', None), ('D9', 'G8', 'F-1', None), ('D9', 'G8', None, 'F5'), ('D9', 'H11', None, 'F5'), ('D9', 'H11', 'G8', 'F5'), ('D9', 'H14', 'G8', None), ('D12', 'H14', 'G8', None), ('E15', 'H14', 'G8', None), ('E18', 'H14', 'G8', None), ('E18', 'E18', 'G8', None)]

- Problem ID:96879, the path: [('F5', 'F5', 'B0', None), ('C10', 'F5', None, None), ('C10', 'F5', None, 'F5'), ('C10', 'E8', None, 'F5'), ('C10', 'G14', None, 'F5'), ('C10', 'G14', 'E8', 'F5'), ('F15', 'G14', 'E8', 'F5'), ('F15', 'G14', None, 'C10'), ('F15', 'I18', None, 'C10'), ('I18', 'I18', None, 'C10')]

- Problem ID:23261, the path: [('F5', 'F5', None, 'H0'), ('F5', 'H10', None, 'H0'), ('F5', 'H10', 'F5', None), ('D11', 'H10', 'F5', None), ('E15', 'H10', 'F5', None), ('E15', 'H10', None, 'H10'), ('E15', 'E15', None, 'H10'), ('E15', 'E18', None, 'H10'), ('E18', 'E18', None, 'H10')]

Appendix D.2. The validation results of the planning model

Table 13: # of Moves Comparison

Problem ID	# of true moves	# of pred moves	difference	# of hand moves in true path	# of hand moves in pred path	difference	# of foot moves in true path	# of foot moves in pred path	difference
82224	10	7	3	5	5	0	5	2	3
62575	15	9	6	6	6	0	9	3	6
241045	21	9	12	11	6	5	10	3	7
23184	17	12	5	8	10	-2	9	2	7
62560	13	7	6	5	5	0	8	2	6
350783	9	9	0	7	6	1	2	3	-1
371579	10	6	4	6	4	2	4	2	2
44578	9	8	1	7	6	1	2	2	0
43010	10	6	4	5	4	1	5	2	3
430586	11	10	1	7	6	1	4	4	0
45211	14	9	5	7	7	0	7	2	5

227999	11	10	1	6	8	-2	5	2	3
20698	14	9	5	6	7	-1	8	2	6
48939	7	8	-1	5	5	0	2	3	-1
39709	8	7	1	4	4	0	4	3	1
345039	23	13	10	8	10	-2	15	3	12
292263	12	7	5	6	5	1	6	2	4
118171	13	8	5	7	6	1	6	2	4
202064	14	10	4	6	8	-2	8	2	6
48723	10	8	2	6	6	0	4	2	2
24681	12	5	7	5	4	1	7	1	6
313875	10	8	2	6	6	0	4	2	2
50356	7	6	1	4	4	0	3	2	1
20038	10	7	3	6	5	1	4	2	2
23261	13	9	4	7	7	0	6	2	4
172010	11	9	2	5	7	-2	6	2	4
268943	15	11	4	7	9	-2	8	2	6
337544	17	10	7	8	8	0	9	2	7
134795	12	9	3	6	6	0	6	3	3
96879	11	8	3	6	6	0	5	2	3

Table 14: Distance Comparison

Problem ID	Total predicted distance	Total true distance
82224	663.95	975.03
62575	843.55	1520.61
241045	913.83	1819.83
23184	987.36	1672.58
62560	643.36	1187.75
350783	883.09	1243.12
371579	638.03	1044.86
44578	734.06	881.63

43010	546.99	1082.39
430586	907.48	1125.58
45211	780.22	1067.09
227999	1029.43	1098.34
20698	804.53	1227.2
48939	698.78	664.45
39709	601.97	734.31
345039	959.45	1804.58
292263	675.99	1121.07
118171	728.94	1436.5
202064	920.48	1034.77
48723	757.36	978.37
24681	586.89	896.7
313875	734.24	877.04
50356	569.42	776.21
20038	706.95	936.69
23261	682.42	893.72
172010	782.72	1009.56
268943	776.09	781.04
337544	901.22	1355.23
134795	838.19	1136.68
96879	767.89	967.85

Table 15: Cost Comparison

Problem ID	predicted path cost	Suggested path cost
82224	5.81	9.3
62575	7.58	16.69
241045	9.28	18.18
23184	8.8	16.23
62560	5.75	13.15
350783	8.09	11.23
371579	6.37	9.72
44578	6.01	7.87
43010	5.05	11.11
430586	8.38	10.57
45211	6.48	10.91
227999	8.33	10.07
20698	7.08	12.88
48939	6.51	6.03

39709	6.14	7.34
345039	9.07	19.93
292263	5.97	10.75
118171	6.22	14.19
202064	7.39	10.81
48723	5.98	8.7
24681	5.22	8.86
313875	6.65	8.58
50356	5.38	7.66
20038	5.75	9.47
23261	5.78	8.59
172010	6.89	10.06
268943	6.57	8.32
337544	7.9	12.54
134795	7.07	12.03
96879	6.73	9.8

Table 16: The Node Accuracies

Problem ID	Percentage of true nodes	Percentage of true hand nodes	Percentage of true foot nodes
82224	0.57	1	0.33
62575	0.55	1	0.38
241045	0.12	0.18	0.07
23184	0.17	0.25	0.12
62560	0.25	0.4	0.18
350783	0.47	0.71	0.3
371579	0.29	0.17	0.38
44578	0.67	0.71	0.6
43010	0.27	0.4	0.2
430586	0.73	0.71	0.75
45211	0.5	1	0
227999	0.31	0.67	0
20698	0.47	0.83	0.27
48939	0.5	0.4	0.67
39709	0.7	1	0.5
345039	0.28	0.75	0.06
292263	0.47	0.67	0.33
118171	0.11	0.14	0.08
202064	0.33	0.67	0.11
48723	0.25	0.5	0

24681	0.08	0.2	0
313875	0.33	0.5	0.17
50356	0.64	1	0.43
20038	0.53	1	0.3
23261	0.36	0.33	0.38
172010	0.44	0.71	0.22
268943	0.17	0.4	0
337544	0.24	0.43	0.1
134795	0.5	1	0.17
96879	0.56	1	0.3
Average:	**0.36**	**0.62**	**0.25**