Pamukkale Univ Muh Bilim Derg, 27(2), 114-121, 2021

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

Estimating the difficulty of Tartarus instances

Tartarus örneklerinin zorluklarının tahminlenmesi

Kaya OĞUZ[1*]

[1]Department of Computer Engineering, Faculty of Engineering, Izmir University of Economics, Izmir, Turkey.
kaya.oguz@ieu.edu.tr

Received/Geliş Tarihi: 24.03.2020 Revision/Düzeltme Tarihi: 24.04.2020 doi: 10.5505/pajes.2020.00515
Accepted/Kabul Tarihi: 24.04.2020 Research Article/Araştırma Makalesi

Abstract

Tartarus is a commonly used benchmark problem for genetic programming. However, it has never been fully explored for its difficulty tuning property. Using the data from a previous study in which we have executed millions of Tartarus instances, we contribute to the literature with an equation to estimate their difficulty. Our approach uses four metrics that are embedded into the equation. These metrics are related to the number of clusters and clusters sizes, the distances of boxes to the edges of the board grid, the number of boxes around the agent, and the minimum number of actions for the agent to reach the largest cluster. The coefficients of these metrics have been fit to the data using the general linear model and a mean residual error of ~0.1 has been achieved. This is the first study that can estimate the difficulty of a Tartarus board without modifying the problem in any way.

Keywords: Tartarus problem, Difficulty estimation, General linear model.

Öz

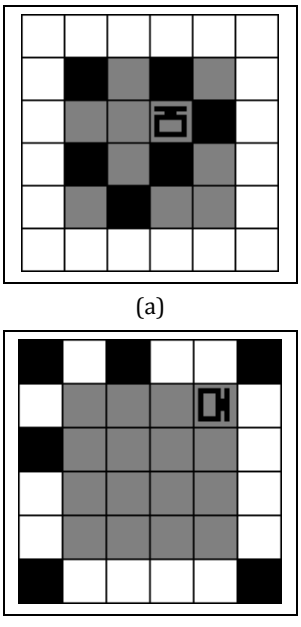
Tartarus genetik programlamada sıkça kullanılan bir kıyaslama problemidir. Fakat zorluk ayarı özelliği henüz tam olarak araştırılmamıştır. Literatüre milyonlarca Tartarus örneği çalıştırdığımız önceki bir çalışmanın verilerini kullanarak zorluklarını tahmin edebilen bir denklemle katkıda bulunuyoruz. Yaklaşımımız denklemin içinde yer alan dört yeni metrik kullanıyor. Bu metrikler küme sayıları ve büyüklüklerine, kutuların kenarlardan uzaklığına, yazılım etmeninin etrafındaki kutuların sayısına ve etmenin en büyük kümeye varması için gereken hareket sayısına bağlıdır. Metriklerin katsayıları veriye genel doğrusal model ile uyarlanmış ve ortalama ~0.1 kadar bir hata başarısına ulaşılmıştır. Bu çalışma Tartarus probleminde bir değişiklik yapmadan problemin zorluğunu tahmin edebilen ilk çalışmadır.

Anahtar Kelimeler: Tartarus problemi, Zorluk tahmini, Genel doğrusal model.

1 Introduction

The Tartarus problem has been proposed by Teller for realizing software agents that can handle an environment using temporal and spatial sensory input [1]. It has been used as a benchmark problem in genetic programming because of its desirable characteristics, such as having a means of difficulty tuning, being precisely defined, relevant, independent of representation, easy to interpret and compare [2]. The focus of this study is its difficulty tuning property which has been looked over because the problem has never been fully explored yet.

In its original proposal, a Tartarus board is defined as a 6×6 grid with impenetrable walls. The initial board has 6 boxes and a bulldozer as an agent, all randomly placed in the inner 4×4 grid, as shown with the board in Figure 1(a) where white squares represent empty spaces, blacks represent the boxes, and the gray ones represent the inner 4×4 grid. The agent can only sense its 8-neighborhood and can perform an action of moving forward or turning left or right in place. If the agent chooses to move forward when there is a box in front of it, it can push this box only if the immediate cell in the direction of movement is empty. The task of the agent is to push the boxes from their initial positions in the inner 4×4 grid towards the impenetrable walls. The agent is given 80 actions for this task and it executes all of them. At the end of these actions, the board is scored a single point for each wall next to a box. For 6 boxes, the top score of 10 is gained by getting two points for the boxes at each corner, where they are next to two walls, and a single point for each remaining box that are next to a wall. Similarly, the lowest score is 0, when there are no boxes next to a wall. A final state that is worth 10 points is shown with the board in Figure 1(b). All reported scores are given in the scale between 0 and 10 throughout this text.



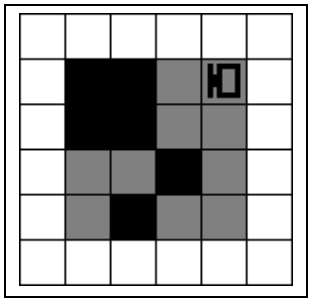
(a)

(b)

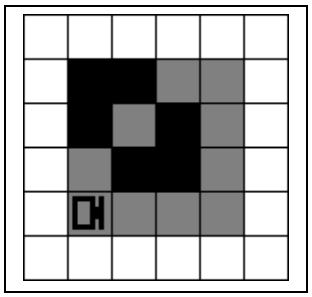
Figure 1. A sample initial Tartarus board is shown in. (a): and a final board that has a score of 10 in shown in (b).

*Corresponding author/Yazışılan Yazar

There are two specific configurations of initial boards that should be emphasized. It is possible that the initial random positions of the boxes can be a 2×2 formation where it is not possible for the agent to push any of these boxes. All the existing studies remove these configurations from the initial set of random boards. The other case is the Willson configuration, named after one of the authors in [3], which leads to an inevitable 2×2 formation. Both initial conditions are shown in Figure 2.



(a)



(b)

Figure 2(a): Shows the 2×2 formation. (b): Shows a Willson configuration.

The purpose of a benchmark is to provide a toy problem that can be used to compare the relative performance of existing and proposed algorithms. To achieve this, it needs to have several characteristics that have been discussed in the literature [2],[4]. The Tartarus problem is apparently well suited as a benchmark when it is evaluated with respect to these characteristics. For example, Tartarus is *relevant* to real applications since it emulates an agent, such as a robot or a drone, in an environment where sensors are used to decide on the best available action to maximize the usefulness. Since the evolutionary algorithms use a lot of instances of a problem, the problem should be *fast* in terms of running time; the agent is only allowed 80 actions in which it simply changes the positions of several boxes and these actions do not have any significant time or space complexity. The problem is *easy* to implement in terms of basic programming structs. The results are also comparable, since they are integer scores, and therefore easy to *interpret and compare*. Tartarus is *representation independent* since it does not require any specific feature of any algorithm. Finally, the problem is defined *precisely*, the rules and the evaluation of the board do not have any ambiguity.

The Tartarus problem ensures all these characteristics, but so far it has not been studied thoroughly so that its *difficulty* can be *tuned*. Teller binds the size of the board and the number of boxes to an *N* variable so that the difficulty of the problem could be changed [1], but several existing approaches do not focus on the difficulty assessment of the problem and divert their efforts to come up with solutions that will improve the score for the 6×6 grid with 6 boxes. This hinders a wider usage of the problem as a benchmark for the evaluation of artificial intelligence algorithms.

Currently, the survey [4] lists Tartarus under the "Path-finding and Planning" category of benchmarks. In this category, the most common benchmark is the artificial ant problem. We contribute to the study of the Tartarus problem by proposing a method that can evaluate its difficulty and therefore promote it as an applicable benchmark for a wider range of genetic programmers. With this regard, we rely on our previous study in which we have observed that for all the possible boards of size 6×6 that have 6 boxes, the difficulty of the problem changes considerably [5]. This means that the size of the grid and the number of boxes are not the only metrics for the difficulty assessment of Tartarus. Instead, we have noticed that the difficulty should also be assessed by evaluating the initial configuration of the boxes and taking the point of view of the agent into consideration.

To the best of our knowledge, there are no studies so far that report the true scores of Tartarus solutions, as we have done in our previous work [5]. We also found no other approach that assess the difficulty of a Tartarus problem using all possible cases, which provide a more robust evaluation of a Tartarus board. We contribute to the literature by defining an equation with four parameters that can be used to estimate the difficulty of any given board. These parameters are based on the initial configuration of boxes, and the initial position and direction of the agent. The coefficients of these parameters are estimated by the general linear model and we have achieved a mean residual error of ~0.1.

The rest of the paper is structured as follows. In the next section we review the existing approaches for difficulty tuning of Tartarus. In Section 3 we discuss the details of how we created the statistical information in our previous study. Section 4 is about the difficulty evaluation of Tartarus, along with the description of parameters and the general linear model to fit the data to the equation. Final section discusses the results and concludes the paper with future remarks.

2 Related work

There are several approaches for solving the Tartarus problem with a board size of 6×6 and 6 boxes, such as the studies by Ashlock et al., Oğuz, and Dick [3],[5],[6], however, existing work on the difficulty assessment of Tartarus is very limited. To the best of our knowledge, there is only one study that focuses on the evaluation of Tartarus boards that provides a method to tune the difficulty of an instance [2]. Before we discuss this study by Griffiths and Ekárt, it should be emphasized that Teller [1] has bound the size of the board to an N variable, which is also used to define the number of boxes approximately as $(N-2)^2/3$ in the inner $(N-2) \times (N-2)$ grid. He calculates the number of moves to complete a tour of the board as $N^2 + 2N - 3 = 45$ and sets the maximum number of moves of the agent to 80, a little short of two complete tours of the board for $N = 6$. However, while there are studies which mention that the difficulty of a Tartarus board depends on the size of the board, most of them use a board size of 6×6 with 6 boxes.

In their concluding remarks, Ashlock and Freeman ask how the hardness of Tartarus would be affected if the board size changed and emphasized that understanding the behavior at different sizes would improve Tartarus as a test problem [7].

Ashlock and Warner study the fitness of agents for sets of Tartarus geometries by using an agent-based metric between these boards [8]. They hypothesize that if the training boards are chosen from a well spaced-out collection of boards, it would be possible to train superior agents that are more generalized. However, they had to reject this hypothesis.

Dick states that the difficulty of the problem can be adjusted by changing the number of allowed actions, the grid size, or the number of boxes [6]. He does not discuss the issue further, but in another paper, he calculates the number of possible boards for other grid sizes and number of boxes [9].

The study by Griffiths and Ekárt is the only study that focuses on the evaluation of a Tartarus board [2]. Their first proposition is to evaluate the state of the board at any time during its execution rather than evaluating it only at the end. They propose an evaluation method in which the agent is rewarded more points if it can push the boxes closer to the edges. This approach also provides a possibility for the genetic programming algorithms to evaluate the current state of the board before the agent runs out of moves. However, this modifies the original definition of the problem, because while the evaluation method rewards the agent for partial success, the final scores of the board are different from the original proposition. Additionally, the agent is only allowed to sense its 8-neighborhood; providing more information about the board is not in the problem definition.

Their next proposition is on the baseline values which enable the comparison of generated Tartarus instances. They remark that the difficulty increases with the board size n and define the number of moves, $m(n)$, and number of blocks, $B(n)$, as functions of n.

Their final proposition is to estimate the difficulty, denoted by D, and therefore tune it if necessary, by using these functions in the following equation where number of impossible-to-move blocks is B_I and the user set number of blocks is B.

$$D = 0.5 \cdot \frac{m(n)}{m} + 0.5 \cdot \frac{B(n)}{B} + \frac{B_I}{B} \quad (1)$$

The authors set the difficulty as *impossible* for the case where $B_I = B$. This equation gives the difficulty of the 6×6 board with 80 moves and 6 boxes as 1. As the number of moves is decreased the difficulty increases, too. However, there is no indication in the original definition that the 6×6 board configuration is the standard, and therefore should have a base difficulty of 1, and other board sizes should be evaluated relative to it.

Furthermore, we have observed in our own trials that even if the board size, the number of boxes, and the number of moves are the same, the difficulty of the problem changes considerably when all possible boards are evaluated. The difficulty does not only depend on these variables but the box combinations that the agent faces. Among the existing approaches, none of them, save our previous study [5], consider the problem from the point of the agent. Therefore, we hypothesize that since it is the agent that solves a Tartarus instance, the point of view of the agent should also be included in difficulty estimation.

3 Tartarus from the agent's point of view

In our previous study we have approached the problem from the agent's point of view. It has improved the understanding of the number of possible boards, as well as the number of possible combinations an agent can come across, and it produced solutions that can solve the Tartarus problem successfully by scoring 8 or above in 88% of all possible boards [5]. This section is a brief summary of that study, but to maintain brevity we have only focused on the parts that are within the scope of this paper.

An initial configuration of a Tartarus board is randomized by placing 6 boxes on the inner 4×4 grid, then, the remaining 10 positions on this inner grid are used to determine the random position of the agent. Since the agent is indifferent to its position and direction on the board, some of the boards are the same from the agent's point of view. We have showed that there are 1869 unique initial board configurations. Therefore, the number of initial possible boards is defined by 4 starting directions, 10 random positions, and 1869 random configurations of boxes which result in a total of $4 \times 10 \times 1869 = 74{,}760$ boards. As will be discussed in the following sections, the initial starting position and the direction of the agent play an important role in the difficulty evaluation of a Tartarus instance.

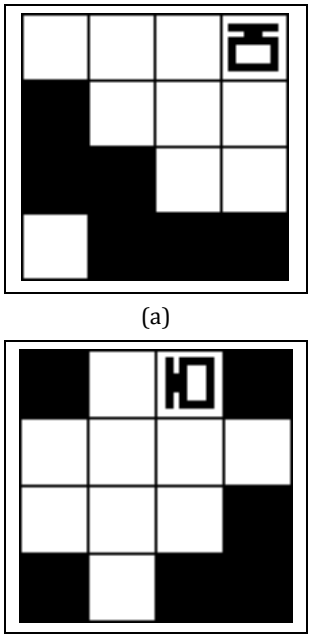
The number of possible combinations for the 8-neighborhood of the agent can be empty cells, cells with boxes, or walls, which means that the cell is out of the edges of the grid. The existing literature uses $3^8 = 6561$ number of combinations while admitting that some of them are not possible, such as the agent being surrounded by all boxes. We have showed that there are only 383 possible combinations. As will be discussed shortly, we have performed millions of Tartarus runs from which we have stored statistical information about how frequently the agent comes across with these combinations. This data is used to understand how hard it is for an agent to handle a specific combination, or how rare it is for the agent to come across some combinations.

Using these updated properties, we have come up with an adaptive genetic algorithm (GA) that breeds finite state machines (FSM) on the GPU (graphics processing unit) to solve the Tartarus problem. We have varied the population sizes from 256 to 2048 with steps of 256 which results in 8 different values. The number of testing boards for each individual in the population was set to 128 and 256, which results in 2 different values. The number of states has been varied from 4 to 12, resulting in 9 different state values. All these configurations have been executed 2 times for each 3 designs. This results in $8 \times 2 \times 9 \times 2 \times 3 = 864$ GA runs. Each GA run has 2000 generations. Even for the lowest case we have at least $256 \times 128 \times 2000 = 65{,}536{,}000$ Tartarus runs in a single GA run. Having millions of Tartarus runs for each GA configuration, we have stored the number of occurrences each 8-neighborhood combination, as well as the most fit solution for each GA run. The most fit 864 solutions are run on all possible 74,760 boards to report the first true scores of Tartarus agents in the literature. Therefore, we had at our disposal a large number of statistics on the occurrence and fitness of each 8-neighborhood combination, as well as 864 different solutions that are run on each possible Tartarus board. This data provides a unique opportunity to study how well the boards are handled with many solutions, and how the agent's point of view can be used to evaluate the difficulty of a Tartarus instance.

4 Difficulty evaluation of Tartarus

Since there are 864 scores for each of the 74,760 boards, the first step to analyze these results is to check how the scores vary statistically for each board. The agent can score between 0

and 10, therefore, we consider boards with lower average scores to be harder than those with higher average scores. Figure 3 shows the hardest and easiest boards when the scores for all 864 runs are averaged. Only the inner 4 × 4 grid is shown for the remaining initial boards to preserve space and to focus on the initial conditions. 864 agents scored an average of 0.43 and 9.53 for the board on in Figure 3(a), and the board in Figure 3(b), respectively. It should be stated that the minimum score of 0.43 was tied with another board that has the same initial configuration of boxes but with the agent facing west, not north.



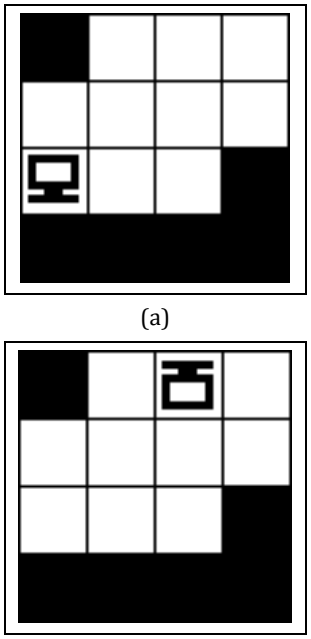
(a)

(b)

Figure 3. The board in (a): was the hardest for 864 different agents, getting an average score of 0.43. The board in (b): Was the easiest, getting an average score of 9.53.

At first observation, these two boards have a distinction and a similarity. Both boards have the L shape, a single box missing from a 2 × 2 formation. The distinction is that the hardest board has a single cluster of boxes; this makes the boxes very hard to move, since they block each other's movement. The easiest board has 4 clusters, a single box in three of them, and three boxes in one. We wanted to investigate further and generated the images of all the boards that scored between 0 and 1 inclusively. There are 27 of these boards, and they have either a single cluster, or two clusters that are unevenly divided, such as where one of the clusters have a single box, and the rest of the boxes are in the larger cluster. Since each board is different not only by the configuration of boxes, but also by the position and direction of the agent, we notice that several of these boards have the same box configuration, but different agent positions and directions.

While this initial observation looked promising, we have decided to check the boards that scored greater than 8 and less than or equal to 9, as well as boards that are greater than 9 and less than or equal to 10 to get a broader view. Figure 4 shows two boards; Figure 4(a) has an average score 0.99, and Figure 4(b) has an average score of 8.36. This shows that most agents were successful in handling a large cluster of boxes, depending on their starting position and direction but with the same configuration of boxes. This observation shows that the agent's initial position and direction is vital in deciding the difficulty of a board.



(a)

(b)

Figure 4. Same distribution of boxes with different starting points creates two very distinct scores. The board in (a): Has an average score of 0.99, and (b): Has an average score of 8.36.

A final visual clue that we have noticed is the abundance of boxes in the central cells of the inner 4 × 4. This agrees with the study of Griffiths and Ekárt [2] where instead of scoring each box next to a wall, they also give points to boxes that are close to the edges on the grounds that the agent has performed positive action and it should be rewarded. However, in our case, we include their distances to the edges as a component in deciding the difficulty of the initial board.

Using these observations, we hypothesize that the difficulty metric of an initial Tartarus board rests on the agent's position and direction, the initial clusters of boxes, and the distances of the boxes to edges. Cluster information and distances of boxes to edges depend on the initial configuration of boxes on the Tartarus board, and the other values depend on the agent's point of view. All these variables are listed along with their notation in Table 1.

Table 1. Variables that affect the difficulty evaluation.

Parameter	Description
h	The difficulty of the board
c	Cluster ratio
d	Average distance to edges
f	Quotient for the number of boxes around the agent
m	Minimum number of actions to largest cluster

4.1 Number of clusters and cluster sizes

We define a cluster as a collection of boxes that are connected by their 4-neighborhood since the agent can move and push boxes in these four major axes. We hypothesize that having a

large cluster of boxes reduces the number of ways a box can be pushed, and therefore increases the difficulty of the board.

The 6 boxes on the board can be grouped in several different clusters. For example, the board can be made up of a single 6-box cluster; or in two clusters where each cluster has 3 boxes. In another configuration, such as the one in Figure 4, two clusters can have an uneven distribution of boxes, such as 5 to 1. Since two of these two cluster configurations are not the same, and since we want to be able to differentiate such cases, we define this metric as the ratio of the largest cluster size to the number of clusters, and we denote it with *c*. In this regard, the board in the first example where there are equal number of boxes in two clusters has a *c* value of $c = \frac{3}{2} = 1.5$, whereas the board in the second example has a *c* value of $c = \frac{5}{2} = 2.5$. This generates a greater value for the board that has an uneven distribution of boxes in its clusters.

Since the number of these combinations are limited, we can say that the highest value we can get for *c* is 6, where there is only a single cluster with 6 boxes, and the lowest value is 0.166 where there is a single box in each 6 clusters.

4.2 Average of distances of boxes to edges

Another metric is the distances of boxes to the edges, regarding the number of moves required to push them. Having most of the boxes in the central location would make a board more difficult. We have defined this as the arithmetic average of each box to the nearest edge, and denote it with *d*. This can be simply computed by their coordinates on the board.

As an example, the two boards in Figure 4 have the same *d* values, 1, since all the boxes are 1 cell away from the edges. It should be noted once again that the figure only shows the inner 4×4 grid, and the boxes are 1 cell away from the edges of the board.

4.3 Number of boxes around the agent

It is important that both metrics *c* and *d* depend on the configuration of boxes and would be the same for both boards in Figure 4, because the position and direction of the agent is not taken into consideration. The agent can be in one of the 10 remaining cells in the inner 4×4 grid, and it can be facing any of the four directions. This results in 40 different cases for the same box configuration, but with the same values for metrics *c* and *d*. Since the difference in these boards are related to the agent, and since we can only take the initial position into consideration for evaluating the difficulty of a board, we define another metric which is related to the number of boxes around the agent in its initial position.

As mentioned earlier, in our previous study we have executed several millions of Tartarus boards in a GA run and for each Tartarus board we have stored the information about how many times the agent comes across a specific combination of boxes in its 8-neighborhood. Since the total number of runs are in the billions, we believe that these frequencies are as close to real probabilities as possible. In this study, we make use of this information from another perspective and find out how frequently a combination occurs in Tartarus runs. To do so, we have summed all the number of occurrences for each combination for all these runs.

Among 383 different combinations only 247 of them are possible on an initial board because the boxes and the agent are placed in the inner 4×4 grid. Therefore, we have filtered the occurrence frequencies to these 247 combinations. However, having 247 different values is not practical when calculating the difficulty of a Tartarus board. We have further processed the data and grouped the combinations by the number of boxes and normalized them with respect to the most frequent one. This yielded a vector of size 7, for each number of boxes from 0 to 6, using the values in Table 2.

Table 2. Normalized frequencies of the number of boxes in the 8-neighborhood of an agent.

Number of boxes	Normalized Frequency
0	0.0741
1	0.4849
2	1.0
3	0.7616
4	0.2710
5	0.0488
6	0.0032

Table 2 shows that the most frequent number of boxes is the ones with 2 boxes, whereas 6 boxes are very rare. Instead of using the number of boxes, we have used these values as the metric *f* to represent how frequently an agent can come across any number of boxes. We hypothesize coming across a combination more frequently makes it easier for the agent to handle it, therefore makes the board easier. Once again, for the boards in Figure 4, the one in 4(a) has an *f* value of 1, and the one in 4(b) has an *f* value of 0.0741.

4.4 Minimum number of actions to the largest cluster

The final metric depends on the position and the direction of the agent. We define this distance metric based on the Manhattan distance of the agent to the largest cluster, and we denote it by *m*.

The Manhattan distance is simply the absolute differences in the *x* and *y* coordinates of two points. In our case, we are looking for the minimum number of actions, therefore we also have to take turning left and right actions into consideration. So, we also must find out the positions of the clusters relative to the position of the agent.

We have applied the following approach which is straightforward to implement. While the number of clusters and the cluster sizes are being calculated for the first metric, we have kept information on a separate list of tuples regarding the clusters and their sizes. At this step, we refer to this information to check the cluster a box belongs to and updated the minimum distance only if the cluster size is greater than or equal to the current minimum distance. For cases where there are equal number of cluster sizes, such as two 3-box clusters, or three 2-box clusters, there will be more than one largest cluster. For each position we find the Manhattan distance to every box and add to this distance value the number of turn actions required. The turns are calculated by considering the position of the agent and the target box. For each direction, the differences in *x* and *y* coordinates between the positions of the agent and the box are evaluated to decide the number of turns.

Being closer to a larger cluster increases the chances of having more boxes around the agent. Therefore, we hypothesize that the further away the agent is, the easier the board for it to handle.

The *m* values for boards in Figure 4 are 1 and 5, respectively. For the board in Figure 4(a), only a single forward action is enough to reach the largest cluster. For the board in Figure 4(b),

the agent should turn right (action 1), move forward (action 2), turn right again so that it faces south (action 3), then perform two more forward actions (actions 4 and 5) to reach the largest cluster. Naturally, it is possible that there would be more than one way to reach the largest cluster, but the parameter requires the minimum value.

4.5 Difficulty Estimation

The values for these parameters have been calculated for all 74760 boards and stored in a matrix of size 74760×4. The mean scores of 864 GA runs on each of these 74760 boards are also stored in a matrix of size 74760×1. We define the difficulty of a board, denoted by h, to be the mean value of these GA runs: the higher the value, the easier the board. We hypothesize that the difficulty has a linear relationship with these parameters and therefore should comply with the following equation

$$h = \beta_1 \cdot c + \beta_2 \cdot d + \beta_3 \cdot f + \beta_4 \cdot m + \varepsilon \quad (2)$$

where the β values represent the coefficients that must be estimated to fit the data, and ε represents the residual error. This is a multivariate normal regression problem, and the coefficients can be estimated with the general linear model (GLM) [10] because we can safely assume that the errors in the residual will follow a normal distribution.

GLM is a commonly used tool and is implemented in several mathematical software packages, such as R, SPSS, and MATLAB. The MATLAB implementation uses the maximum likelihood estimation and returns the estimated regression coefficients, estimated variance-covariance matrix, and the residuals.

Using these metrics and the mean values for each board, we have obtained a vector of coefficients which can be used to compute the difficulty of a random Tartarus board. The mean value for the residuals is 0.0923; that is, there will be around a ~0.1 error on average when the difficulty of a board is evaluated.

For further analysis, we have plotted the residual histogram, as shown in Figure 5, to check how well the data has been fit. The figure shows that there are instances where the error is very large in some cases, but a majority of the boards are identified within an error range of (-2, 2). The total number of boards in bins -2 to 2 are 69,394, which is the 92.82% of all boards. The histogram shows that the equation can estimate the difficulty within a tolerable error for a large percentage of all available boards.

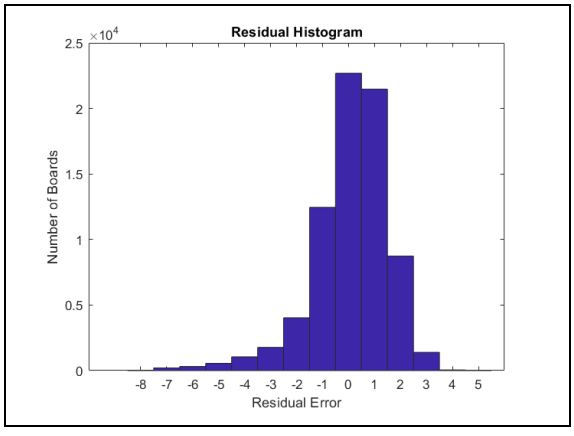


Figure 5. Residual error histogram shows that a large number of instances are accumulated around 0 error.

When β variables are replaced with their values, the equation for h becomes:

$$h = (-0.4261) \cdot c + 5.7447 \cdot d + 0.9230 \cdot f + 0.2242 \cdot m \quad (3)$$

We have made observations on how the parameters would be affecting the board difficulty as they were being introduced in previous sections. The ratio of the largest cluster size to the number of clusters have a negative effect on the difficulty; that is, it makes the h value to be lower, which increases the difficulty. The largest value parameter c can have is 6, as previously mentioned in Section 4.1. This would decrease the estimated difficulty of the board by $6 \times (-0.4261) = -2.5566$, hence making it more difficult.

The estimated coefficient for parameter d is the largest compared to other coefficients. Although we have hypothesized that having boxes closer to the edges would make a board easier, it appears that the value of the coefficient does not support this. Therefore, we have analyzed the values for this parameter further by observing the correlation between the values of d and the difficulty of the boards. Interestingly, there is a negative correlation, a specific value of -0.2028, meaning that as the value for d increases, the score of the board decreases, which means the board becomes more difficult, as we have hypothesized. Having a large value for the coefficient of parameter d could be explained by the dependencies of the parameters on each other; the boxes in the large cluster in Figure 4(a) and 4(b) have a short distance, but the boxes form a large cluster which makes them difficult to move. The general linear model has come up with a coefficient for parameter d which also contributes partially to other parameters.

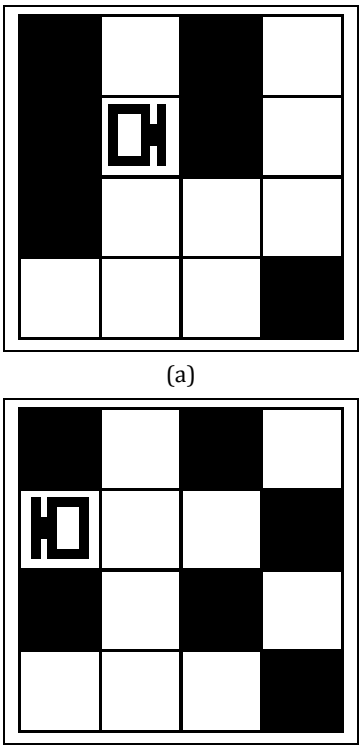
For parameter f, we expected to have an easier board when the number of boxes around the agent are more common. The value of the parameter is in accordance with our initial observation.

Finally, for parameter m, we have hypothesized that being far away from the largest cluster would make the board easier. Having a larger m value increases the value for h, which means it becomes an easier board. This is also in accordance with our initial observation. We have tested the equation on some boards and compared them to the mean scores of 864 GA runs. Two of them are demonstrated using Figure 6.

For the board in Figure 6(a) the value for c parameter is 1, because the largest cluster has 3 boxes, and there are 3 clusters in total. The value for d is 1.1667, because five of the boxes have distances of 1 to the edges, only one box is 2 cells away. Since there are 5 boxes around the agent, Table 2 says that f parameter should be set to 0.0488. Finally, the largest cluster is just behind the agent, but it must perform two turn actions, and a single move forward to reach it, therefore m parameter has a value of 3. When these values are used in the equation for h, we get an estimate of 6.9941, and the mean scores of GA runs is 6.3495.

The board in Figure 6(b) has six clusters with a single box in each. Its c parameter has a value of 0.166. One of these boxes has a distance of 2 to the edges, so the d parameter has a value of 1.1667. The agent has two boxes around it, when referred to Table 2 we get a value of 1 for the f parameter. Finally, each of these clusters are the largest, so the m parameter has a value of 2, since the agent either must turn left or right and move forward to reach one of the single box clusters. When these

values are used in the equation, we get a value of 8.0028, and the mean scores are 8.7593.



(a)

(b)

Figure 6. Two boards selected randomly out of 74,760 possible ones.

4.6 Implementation details

The data generated in our previous study uses C programming language with CUDA version 10.1 libraries to execute the GA runs on the GPU. We have processed these files and generated the input and output matrices using NumPy on Python version 3.7. The general linear model has been fit by MATLAB version 2018a using the *mvregress* function. The board images are generated by the Pillow library on the same Python version.

5 Discussion and conclusion

In this study we have used the data that has been generated in a previous study where billions of Tartarus runs have been executed to evolve software agents that can handle a Tartarus board. This data provided us an opportunity to come up with an equation that can estimate the difficulty of a board using the configuration of boxes and the position and direction of the agent.

Instead of providing the statistical information about the mean scores of 74,760 boards, or the occurrence frequency of 247 combinations of boxes in the agent's 8-neighborhood, we believe that this equation is a much straightforward and compact way to calculate the difficulty of a board. It also shows that the difficulty indeed changes with the configuration of boxes and the position of the agent.

In contrast to the only existing study [2], we evaluate the difficulty of the Tartarus board in its initial configuration when the size of the board and the number of boxes are fixed, rather than how the difficulty varies when they are modified. Our contribution is vital because we do not modify the original Tartarus problem. Using the original definition of the problem enables several researchers use the same problem and compare their results.

As discussed earlier, a benchmark should have a good means to tune the difficulty of the problem. We believe that this study is an important first step for improving the understanding of the Tartarus problem and how its difficulty varies not only with the size of the board and the number of boxes but also with the configuration of boxes and the position and direction of the agent.

Although we have come up with an equation that can estimate the difficulty, we have used the values for a specific board size and number of boxes. In future work, we are planning to look for ways to generalize it to other sizes.

6 Acknowledgments

The author would like to thank the anonymous reviewers whose comments and feedback have improved the text significantly.

7 Author contribution statements

In the scope of this study, Kaya Oğuz contributed formation of the idea, design, supplying the materials, data collection, conducting the analyses, literature review, writing and assessment of the obtained results.

8 Ethics committee approval and conflict of interest statement

There is no need to obtain permission from the ethics committee for the article prepared.

There is no conflict of interest with any person / institution in the article prepared.

9 References

[1] Teller A. *The Evolution of Mental Models.* Editors: Kinnear Jr KE. Advances in Genetic Programming, 199-217, Cambridge MA, USA, MIT Press, 1994.

[2] Griffiths TD, Ekárt A. *Improving the Tartarus Problem as a Benchmark in Genetic Programming*. Editors: McDermott J, Castelli M, Sekanina L, Haasdijk E, García-Sánchez P. Genetic Programming, 278-293, Cham, Springer, 2017.

[3] Ashlock D, Willson S, Leahy N. "Coevolution and Tartarus". *Proceedings of the 2004 Congress on Evolutionary Computation*, Portland, OR, USA, 19-23 June 2004.

[4] McDermott J, White DR, Luke S, Manzoni L, Castelli M, Vanneschi L, Jaskowski W, Krawiec K, Harper R, De Jong KA, O'Reilly UM. "Genetic programming needs better benchmarks". *GECCO '12: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, Philadelphia, USA, 07-11 July 2012.

[5] Oğuz K. "True scores for Tartarus with adaptive GAs that evolve FSMs on GPU". *Information Sciences*, 525, 1-15, 2020.

[6] Dick G. "A true finite-state baseline for Tartarus". *GECCO '13: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, Amsterdam, Netherlands, 6-10 July 2013.

[7] Ashlock D, Freeman J. "A pure finite state baseline for Tartarus". *Proceedings of the 2000 Congress on Evolutionary Computation*, La Jolla, CA, USA, 16-19 July 2000.

[8] Ashlock D, Warner E. "The geometry of Tartarus fitness cases". *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, Hong Kong, China, 1-6 June 2008.

[9] Dick G. "An effective parse tree representation for Tartarus". *GECCO '13: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, Amsterdam, Netherlands, 6-10 July 2013.

[10] Mardia K, Kent J, Bibby J. *Multivariate Analysis*. 1st ed. London, UK, Academic Press, 1979.