Telepathy

Author: Hirotaka Yoneda (square1001)

0 Introduction

The objective of this problem "Telepathy" is to devise a strategy for two people who are in different places to meet as soon as possible. One important characteristic of the problem is that, unlike normal Communication Tasks, the two cannot communicate at all. So, we need to create a strategy utilizing the *tree structure*[*1] of the road network.

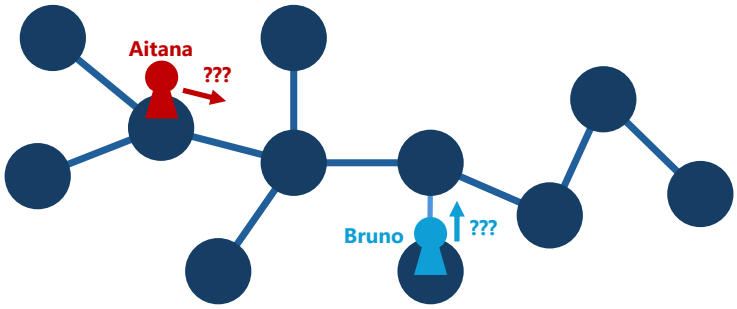

Figure-1 The image diagram of this problem "Telepathy"

This problem consists of three subtasks:

- **Subtask 1 (40 points)** The labelings of Aitana's and Bruno's maps are the same
- **Subtask 2 (40 points)** The road network forms a straight line (that is, a path graph)
- **Subtask 3 (20 points)** No additional constraints.

Also, the score will be decided by the turn number k^* when Aitana and Bruno meet again. The solution to this problem can be divided into two types: the solution that k^* is proportional to the number of vertices N (15% to 25% scores), and the solution that k^* is proportional to the distance of two players d (40% to 100% scores). In this editorial, Chapter 1 explains the former, and Chapters 2 and 3 explain the latter.

[*1] Tree structure is a term in graph theory, which refers to an undirected connected graph without cycles.

1 Fundamental Solutions

In Chapter 1, we explain "how to make Aitana and Bruno meet again", without considering the number of turns too much. For the solutions in this chapter, k^* will be proportional to the number of vertices N.

1.1 Subtask 1: When the Labelings are the Same

One of the most natural strategies is to designate the vertex labeled 0 as the *meeting place*, and Aitana and Bruno directly go to the vertex labeled 0. Then, they can meet again within N turns. This solution is worth 25% of Subtask 1 (that is, 10 points).

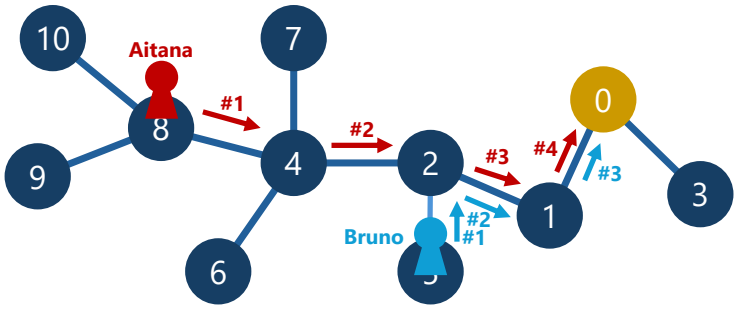

Figure-2 An example of movements in Section 1.1's solution: meet again in 4 turns

To implement this, one needs to compute the path from label v ($v = S, T$) to label 0. This can be computed by an algorithm with Depth-First Search (DFS) or by finding the shortest path using Breadth-First Search (BFS).

1.2 Subtask 2: Path Graph Case

When the road network forms a straight line, it is natural to set the meeting place to the center.

However, when N is an even number, there are two "central vertices", but only one of them can be the meeting place. When the labelings of two maps are different, we have no way to distinguish the two central vertices. How do we solve this issue?

One solution is that, when a player arrives at one of the central vertices, she stays at the vertex if Aitana, and he repeatedly moves between the two central vertices back and forth if Bruno.

Since they can meet again within 1 turn after both arrive at the center(s), the worst-case performance is $\frac{1}{2}N$ turns. This solution is worth 25% of Subtask 2 (that is, 10 points).

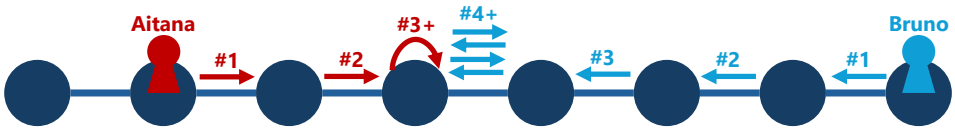

Figure-3 An example of movements in Section 1.2's solution: meet again in 4 turns

1.3 Subtask 3: General Case and Tree's Center

In order to generalize the Subtask 2 solution to the general tree, one may come up with the following question: *how to set convenient meeting places for general trees, like the central vertices for the path graph?*

One answer to this question is to use the *center* of the tree. The centers of a tree are the vertices where the "distance to the farthest vertex" takes the minimum value. It is known that, for any tree, there are one or two centers, and when there are two centers they are adjacent.[*2]

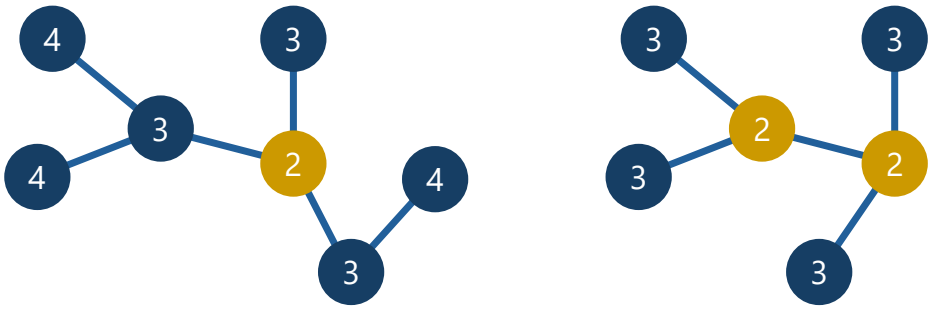

Figure-4 Two examples of tree centers. The number in each vertex represents the distance to the farthest vertex.

Therefore, if we designate the tree centers as meeting places, they can meet again using a similar strategy to the Subtask 2 solution. Since the distance to the meeting place is less than $\frac{1}{2}N$, the worst-case performance is $\frac{1}{2}N$ turns. This solution is worth 25 points.

1.4 Alternative Solution with Euler Tour

We also introduce a completely different strategy for this problem. Let's consider when Aitana stays at the initial vertex forever. Then, Bruno's objective becomes "visiting all vertices".

[*2] The center of trees is a well-known property in graph theory, so if you do not know, we recommend checking about it. Let the longest path of the tree be $v_0, v_1, \ldots, v_\ell$. If ℓ is an even number, the center of the tree is $v_{\ell/2}$, where the distance to the farthest vertex is $\ell/2$. If ℓ is an odd number, the centers of the tree are $v_{(\ell-1)/2}$ and $v_{(\ell+1)/2}$, where the distance to the farthest vertex is $(\ell+1)/2$.

Bruno can traverse each edge twice and go back to the initial vertex. This way is called *Euler tour* of a tree.[*3] Then, Bruno can visit all vertices using $2(N-1)$ turns, and he can meet Aitana in some of the turns. This solution is worth 15 points.

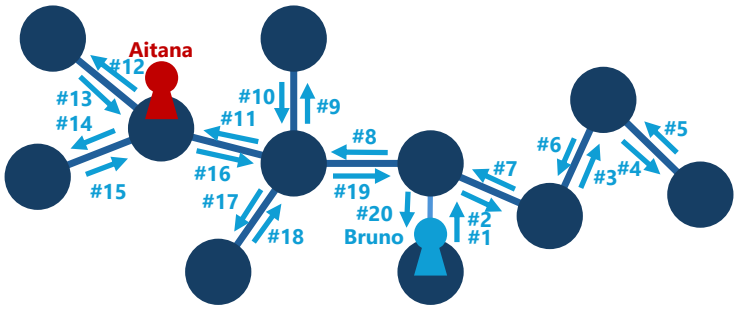

Figure-5 An example of movements in Section 1.4's solution: meet again in 11 turns

2 Solutions for Path Graph

From this chapter, we explain the solution where Aitana and Bruno meet within the turns proportional to the distance d. This means that, regardless of how large N is, they must meet soon if d is small enough. To get some solution ideas, we first consider the case where the road network forms a straight line (Subtask 2).

2.1 What to Solve: The Treasure Problem

First, we consider the strategy where Aitana stays at the initial vertex forever. Then, the problem for Bruno can be interpreted in the following way:

Problem Bruno is standing at coordinate 0 in the number line. The treasure is hidden at coordinate x, but Bruno does not know this position. In each turn, Bruno can move 1 position to the right or to the left. Is it possible for Bruno to visit coordinate x and find the treasure within $10\,|x|$ turns?

Here, the main problem when Aitana is initially at the vertex with ID s and Bruno is initially at the vertex with ID t, corresponds to the treasure problem of $x = s - t$.

[*3] Consider a new graph with $2(N-1)$ edges where each edge of the tree is doubled. The degrees of all vertices of the new graph will be even numbers, so there exists an *Eulerian circuit*, which is a way to pass each edge once and go back to the initial vertex. This way corresponds to the Euler tour. An algorithm using Depth-First Search (DFS) is a common method for computing an Euler tour.

2.2 Repeating Back-and-Forth

Let's consider how Bruno should move. First, suppose Bruno moves right. Considering the possibility of $x = -1$, he needs to visit there within 10 turns, so he needs to turn left at some point (specifically, before turn 5). He then moves left, but considering the possibility of $x = 5$, he needs to visit there within 50 turns, so he must turn right at some point. In this way, he must move like going right, going left, going right, going left, back and forth.

We write this idea mathematically. Let $r > 1$ be a real constant. Bruno first moves right toward coordinate 1, then moves left toward coordinate $-r$, then moves right toward coordinate r^2, then moves left toward coordinate $-r^3$, and so on.

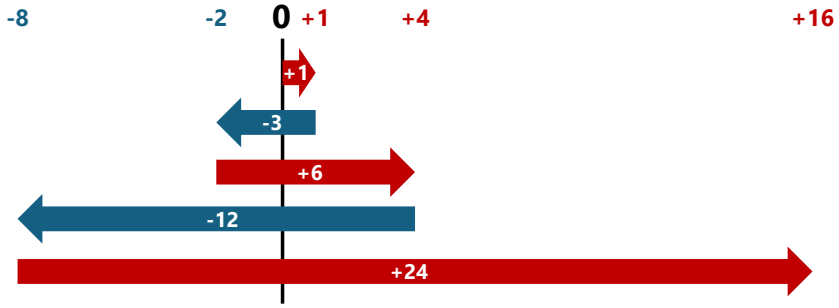

Figure-6 The strategy to find the treasure (when $r = 2$)

Here, r needs to be adjusted to the optimal value. If r is too small, we need to change the direction too much to go farther, but if r is too large, the loss caused by changing direction being too late becomes significant.

So let's calculate the worst-case value of (the time spent to find treasure) / $|x|$. One of the worst cases is when $x = r^{2k} + 1$. In this case, the time spent to find treasure is:

$$1 + (r + 1) + \cdots + (r^{2k+1} + r^{2k}) + ((r^{2k} + 1) + r^{2k+1}) = \frac{2(r^{2k+2} - 1)}{r - 1} + (r^{2k} + 1) = \frac{2r^2 + r - 1}{r - 1}x - \frac{2(r^2 + 1)}{r - 1}$$

Here, the minimum value of $\frac{2r^2+r-1}{r-1}$ is 9 when $r = 2$. Another worst case is when $x = -r^{2k+1} - 1$, but it leads to the same conclusion due to symmetry. Therefore, $r = 2$ is optimal, and he can find treasure in $9\,|x|$ turns.

The main problem is with the path graph, not the number line, so the "coordinate" may exceed the endpoints of the path graph. In such a case, he can just stay at the endpoint. This solution achieves $9d$ turns, which is worth 40% of Subtask 2 (that is, 16 points; note that we can actually get a bit better score due to $N \leq 200$).

2.3 Cooperation for the Efficiency

In the previous solution, Aitana stayed at the initial vertex, so there is some room for improvement when Aitana moves in cooperation with Bruno. Let's consider the effect in the number line setting where Aitana and Bruno can move one position per turn.[*4]

Let's focus on the changes of (Aitana's coordinate) - (Bruno's coordinate). If this value becomes zero, they meet again. This value can change by 2 if Aitana and Bruno move in different directions in the same turn. So, for the treasure-finding problem in Section 2.1, we can allow moving 2 positions to the right or to the left. However, note that if we move 2 positions in one turn, the place of 1 position ahead is jumped and not regarded as visited. Therefore, we move 1 position per turn when the players are in the range of "visiting for the first time", as shown in Figure 7.

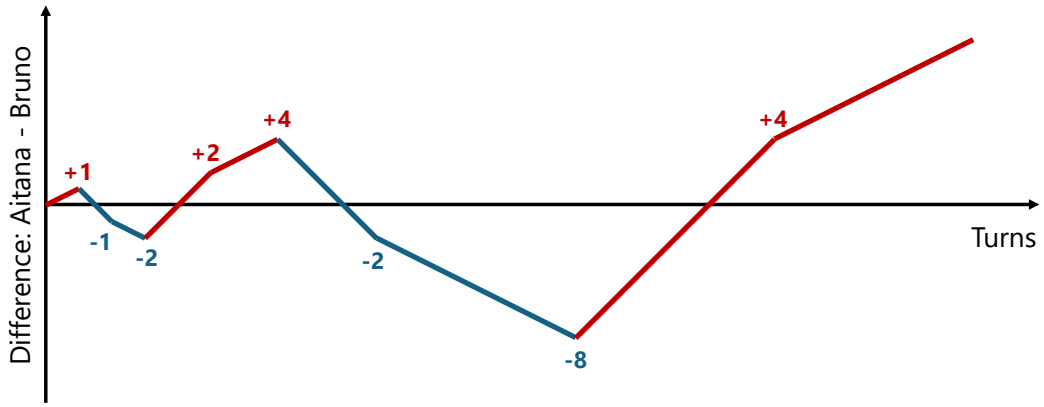

Figure-7 The graph of (Aitana's coordinate) - (Bruno's coordinate) (when $r = 2$)

Let's consider the optimal r for this way. When $x = r^{2k}$, the time spent to find treasure is:[*5]

$$\sum_{i=0}^{2k-1}\left(\frac{r^{i+1}+r^{i}}{2}+(r^{i+2}-r^{i})\right)+\frac{r^{2k+1}+r^{2k}}{2}\approx\frac{(r^{2k+2}+r^{2k+1})/2+(r^{2k+2}-r^{2k})}{r-1}\approx\frac{3r^{2}+r-2}{2(r-1)}x$$

The optimal value is $r = 1 + \frac{\sqrt{6}}{3}$ which achieves the ratio $\frac{7}{2} + \sqrt{6} \approx 5.95$. Howevere, even if we set $r = 2$ conventionally, we achieve $6\,|x|$ turns.

In the main problem, since the labelings of the maps can differ, we cannot distinguish which direction of the path graph corresponds to the positive direction on the number line. Instead, we can regard the direction

[*4] This setting corresponds to the case where the labelings of Aitana's and Bruno's maps are identical (that is, also satisfying Subtask 1 constraints).

[*5] This analysis has the error of size $O(k)$, but it is negligible.

towards the center as the positive direction. Similarly to the solution in Section 1.2, when the player is in the central vertices, she stays if Aitana, and he repeatedly moves between the two central vertices if Bruno.[*6] Then, we can achieve $5.95d$ turns, which is worth the full score in Subtask 2 (40 points).[*7]

3 Solutions for General Trees

In this chapter, we explain the solution for general trees. We combine the ideas in Chapter 1 and Chapter 2.

3.1 Solution with $9d$ Turns

The solution in Section 1.3 employed the strategy of advancing to the center of the tree. However, since Aitana and Bruno advanced at the same pace, they could not meet before reaching the center. To address this issue, we apply the concept introduced in Section 2.2 and derive the following algorithm. We set $r = 2$.

1. First 1 turn: Aitana moves toward the center, and Bruno stays.
2. Next $r + 1$ turns: Bruno moves toward the center, and Aitana stays.
3. Next $r^2 + r$ turns: Aitana moves toward the center, and Bruno stays.
4. Next $r^3 + r^2$ turns: Bruno moves toward the center, and Aitana stays.
5. (and so on)

Note that when the player reaches the center, he/she stays because one cannot move further (for Bruno, he may move back and forth between two centers).

Why does this solution work? For simplicity, think about the case where there is only one center. Consider the rooted tree, which is rooted at the center, and let f_A and f_B be the depths of Aitana's and Bruno's current positions, respectively. Then, in Step 1, 3, 5, ... $f_B - f_A$ will be incremented by 1 per turn, and in Step 2, 4, 6, ... $f_B - f_A$ will be decremented by 1 per turn.

In this algorithm, Aitana and Bruno meet if and only if the following conditions are met:

1. $f_B - f_A = 0$.
2. At least d turns have passed.[*8]

[*6] Note that, if we implement like "once the player reaches the center, he/she does not go outside the center", it may be less efficient than $5.95d$ because, in some cases, it does require going back.

[*7] Aitana and Bruno can be initially at the different sides, but they can meet early enough because the timing where "both players are in the center" arrives early enough.

[*8] This is because, when v is the lowest common ancestor (LCA) of Aitana's and Bruno's initial positions, both players must be at v or the ancestor. The additional condition for this, along with $f_B - f_A = 0$, is "d turns are passed", because $f_A + f_B$ will always be

Here, the change of $f_B - f_A$ will be as identical as Figure 6 in Section 2.2, so they can meet again in $9d$ turns.[*9] This solution is worth 40 points (note that we can actually get a bit better score due to $N \leq 200$).

3.2 Cooperation for the Efficiency: Revisited

To improve from $9d$ turns, let's apply the idea from Section 2.3. That is, change $f_B - f_A$ by 2 in one turn, by an action such as"Aitana advances to the center by one step, while Bruno retreats from the center by one step". Then, we consider the strategy to change $f_B - f_A$ as shown in Figure 7 (Section 2.3), setting $r = 2$.

The arising issue is that, unlike Subtask 2, they may not meet even if $f_B - f_A = 0$ is met, if a sufficient number of turns have not yet passed. Specifically, if the movement is the same as in Figure 7, they only pass even-numbered coordinates when they move 2 positions per turn. Thus, they may not meet again forever when the initial $f_B - f_A$ is an odd number. This issue can be resolved by making the following adjustment: when they move 2 positions per turn, they pass even-numbered coordinates when moving in the positive direction, and they pass odd-numbered coordinates when moving in the negative direction, specifically in the following way:

$$0, +1, -1, -2, 0, +2, +4, +3, +1, -1, -3, -5, -7, -8, -6, -4, -2, 0, +2, +4, +6, +8, +10, \ldots$$

Now, it seems like $6d$ turns is achieved. However, when implemented, this solution unfortunately only achieves $7.5d$ turns, which is worth 70% of the points.[*10] Why?

Since we introduced turns that change $f_B - f_A$ by 2, the second condition of the meeting will become stricter than "d turns have passed". The essential condition is "$f_A + f_B$ has decreased by at least d compared to the beginning". Since $f_A + f_B$ does not change when moving 2 positions in one turn, the second condition in this algorithm becomes "at least d turns which moves 1 position have happened".

Let's examine the impact of this issue on efficiency. The worst case is that, when the players are at coordinate r^{2k} when going to coordinate r^{2k+2}, exactly $d - 1$ turns which moves 1 position have happened, and r^{2k} is an even number. Then, the players further need to move the following path:

$$r^{2k} \to r^{2k+2} \Rightarrow -r^{2k+1} \to -r^{2k+3} \Rightarrow r^{2k}$$

At the beginning of this movement, $(r+1)r^{2k}$ turns which moves 1 position have already happened. Thus, we can regard it as $d = (r+1)r^{2k}$. Also, the turn number that they meet again is, by using the result from Section

decremented by 1 per turn.

[*9] The difference is that even if $f_B - f_A$ becomes zero they may not meet if d turns have not yet passed. However, when $f_B - f_A$ becomes zero at turn t, $f_B - f_A$ becomes zero again at turn $(1 + \frac{2}{r-1})t$ or before, which is early enough.

[*10] Since this solution achieves $6d$ turns in Subtask 2, one can get 82 points.

2.3:

$$\left(\frac{3r^2+r-2}{2(r-1)}+(r^2-1)+\frac{r^2+r}{2}+(r^3-r)+\frac{r^3+1}{2}\right)r^{2k} \tag{1}$$

Dividing this by d yields $\frac{3r^3-3r^2+2r-1}{2(r-1)}$. The value is $\frac{15}{2}$ when $r=2$, so it is concluded that this solution only achieves $7.5d$ turns in the worst case.

Now, did we really hit a wall here?

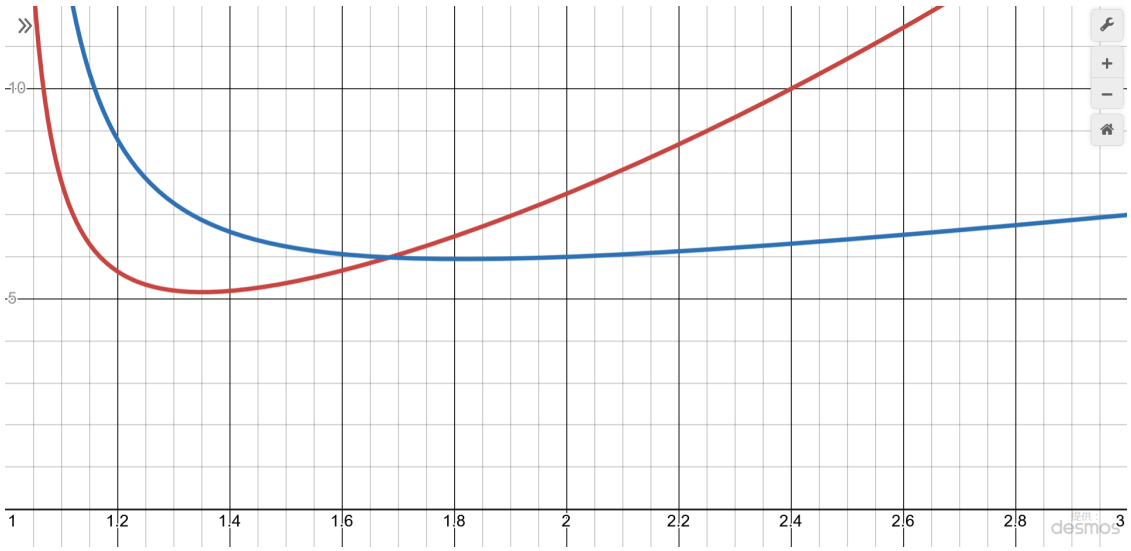

Figure-8 Blue: the efficiency when visiting for the first time $\frac{3r^2+r-2}{2(r-1)}$, Red: the efficiency when visiting for the second or later time $\frac{3r^3-3r^2+2r-1}{2(r-1)}$

In fact, according to Figure 8, $r=2$ is not the optimal choice. The optimal parameter is $r=1.685$.[*11] For this parameter, $5.99d$ turns[*12] is achieved both for the case when visiting for the first time or when visiting for the second or later time.

Therefore, we can get the full score for this problem. Note that, depending on the implementation, the number of turns required can be "$5.99d$ + (constant)", which may exceed $6d$ when d is small, so we need to be cautious about it.

[*11] The solution of $3x^3-6x^2+x+1=0$, approximately $x\approx 1.684695$.
[*12] The solution for $8x^3-56x^2+50x-9=0$, approximately $x\approx 5.987547$