

がくぶぜみ 2 しゅうめ

おの

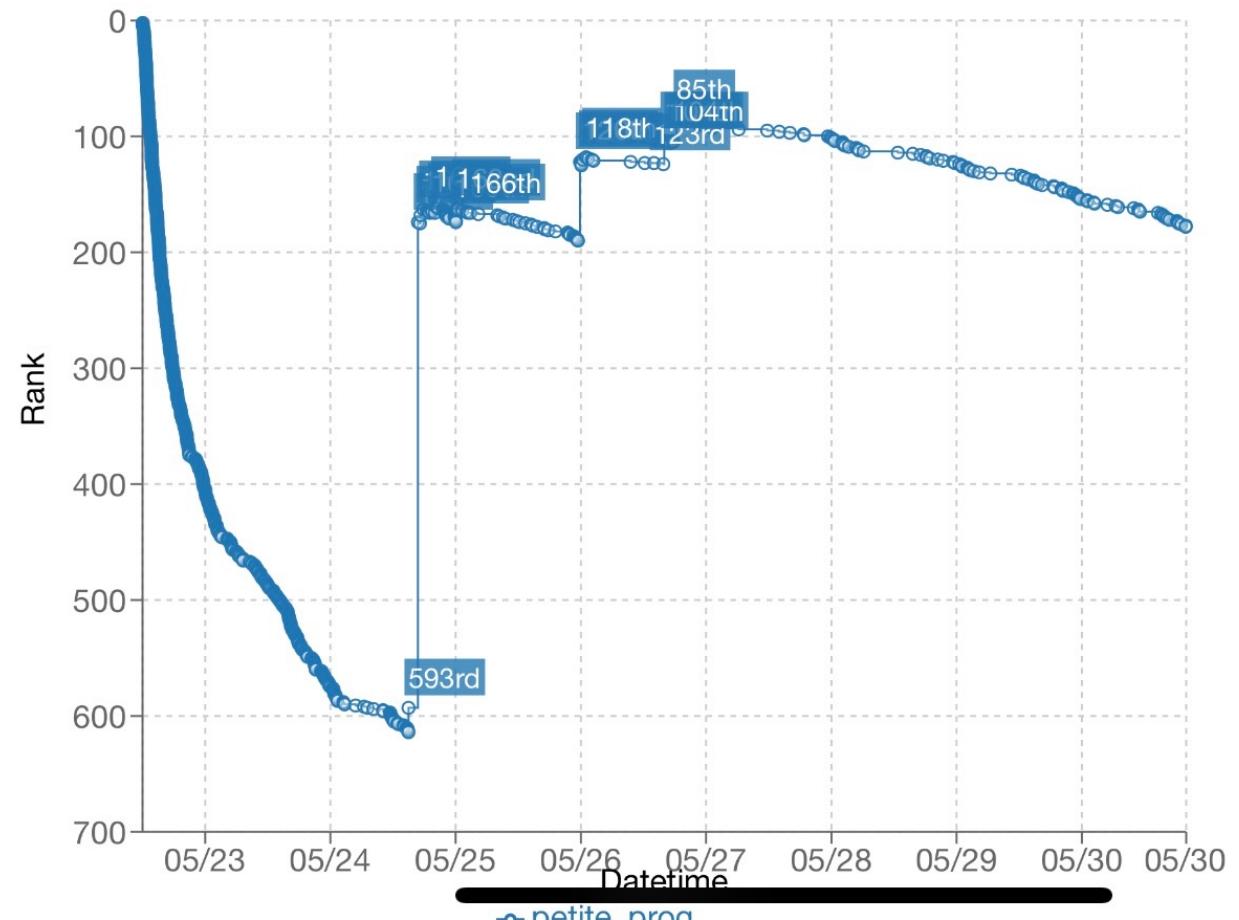
AHC003 リンク

最終順位:175位/1000人

AtCoder Marathon Replay



Replay of AtCoder Heuristic Contest 003



ストーリー

AtCoder社は最短路アルゴリズムを活用した道案内アプリを開発している。サービスの対象エリアは30×30個の頂点をグリッド状に繋いだ道路網で表される。ユーザーが現在位置の頂点と目的地の頂点を指定すると、アプリはその間の最短経路を出力する予定だ。困ったことに、アプリのリリース予定日が迫っているにもかかわらず、最短経路の計算に必要不可欠な各辺の長さの計測が全く出来ていない。そこで、事前に辺の長さを計測することを諦め、最短でないパスの出力も許すこととした。ユーザーが目的地に到着するまでに実際にかかった時間の情報をもとに、出力した各辺の長さを推測することで、徐々に性能の改善が可能であるはずだ。

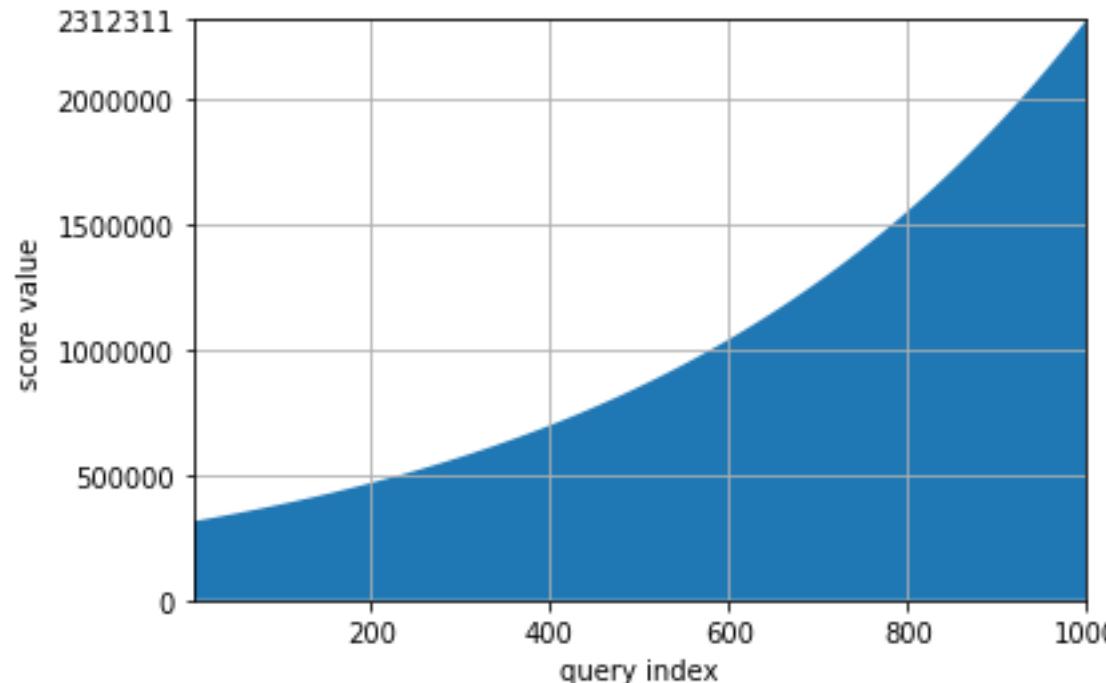
問題定義

- ・ インタラクティブ.
- ・ 30×30 のグリッドグラフで最短距離クエリが1000件飛んでくる.
- ・ 辺の重みはわからない.
- ・ 経路を答えるとその経路長が返答されるので、辺の重みを推定しながら最短経路に近いものを答えよ.

スコア計算式

$$\text{score} = \text{round}(2312311 \times \sum_{k=1}^{1000} 0.998^{1000-k} \times \frac{a_k}{b_k})$$

- a_k : k 個目のクエリに対する真の最短距離長
- b_k : k 個目のクエリに対して答えたパスの長さ



各クエリで得られる
スコアの範囲
出典:s

クエリについて

- 頂点 $s_k = (si_k, sj_k), t_k = (ti_k, tj_k)$ が与えられ, その頂点間のパスを答える.
- その経路長を b_k としたとき, 返答は $[0.9 \times b_k, 1.1 \times b_k]$ の範囲から一様ランダムな値が返答される.
- 与えられる頂点対 (s_k, t_k) は必ずマンハッタン距離10以上.
$$\Leftrightarrow |si_k - ti_k| + |sj_k - tj_k| \geq 10$$

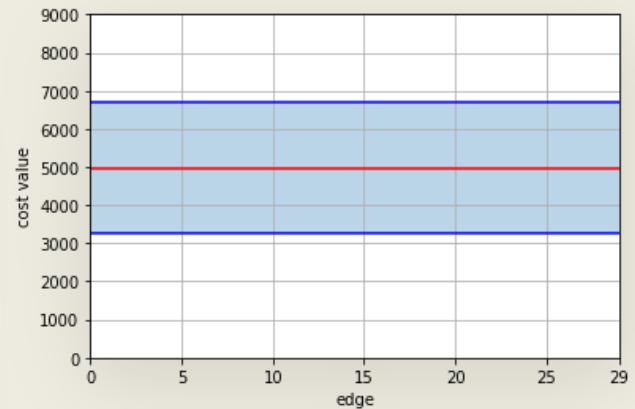
辺コスト

行方向, 列方向の辺コストは独立 (以降, 行について述べるが列も同様).

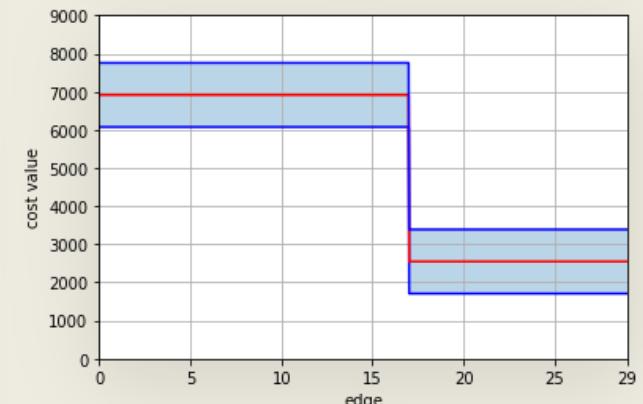
1. まず最初に $D \in [100, 2000]$ と $M \in \{1, 2\}$ が一様ランダムに選ばれる.
2. $M = 1$ のとき, 各 i で:
 1. $H \in [1000 + D, 9000 - D]$ が一様ランダムに生成される.
 2. 各 $j \in \{0, \dots, 28\}$ の頂点 $(i, j), (i, j + 1)$ 間の辺コストは $[H - D, H + D]$ から一様ランダムに選ばれる.
3. $M = 2$ のとき, 各 i で:
 1. $H_0, H_1 \in [1000 + D, 9000 - D], x \in \{1 \dots 28\}$ が一様ランダムに生成される.
 2. 各 $j \in \{1, \dots, x - 1\}$ の頂点 $(i, j), (i, j + 1)$ 間の辺コストは $[H_0 - D, H_0 + D]$ から一様ランダムに選ばれる.
 3. 各 $j \in \{x, \dots, 28\}$ の頂点 $(i, j), (i, j + 1)$ 間の辺コストは $[H_1 - D, H_1 + D]$ から一様ランダムに選ばれる.

1列分の辺コスト例

$M = 1$



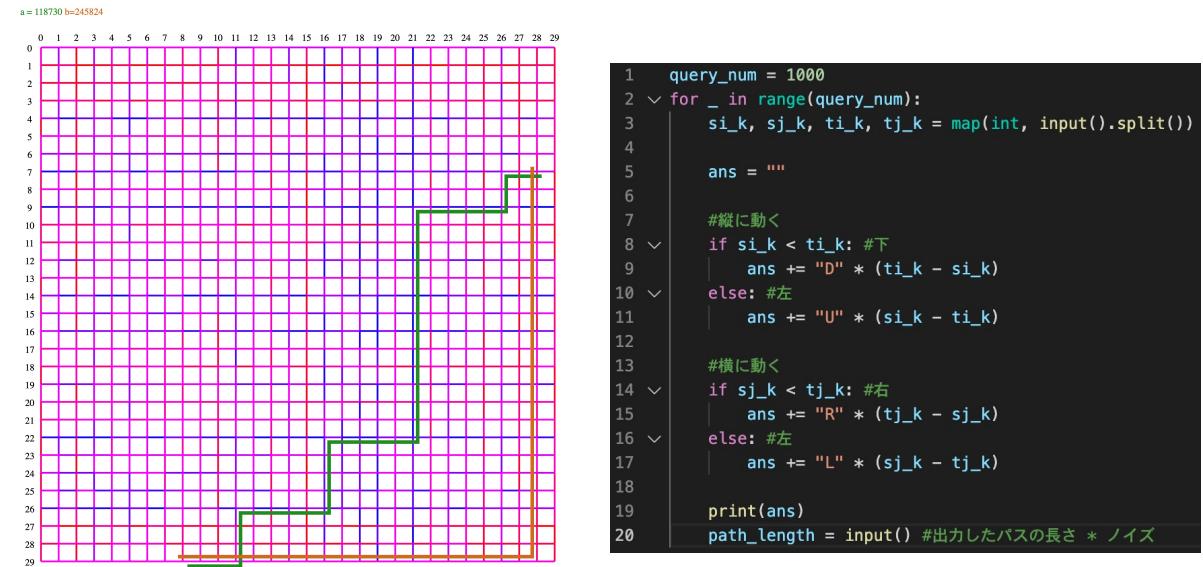
$M = 2$



$M = 2$

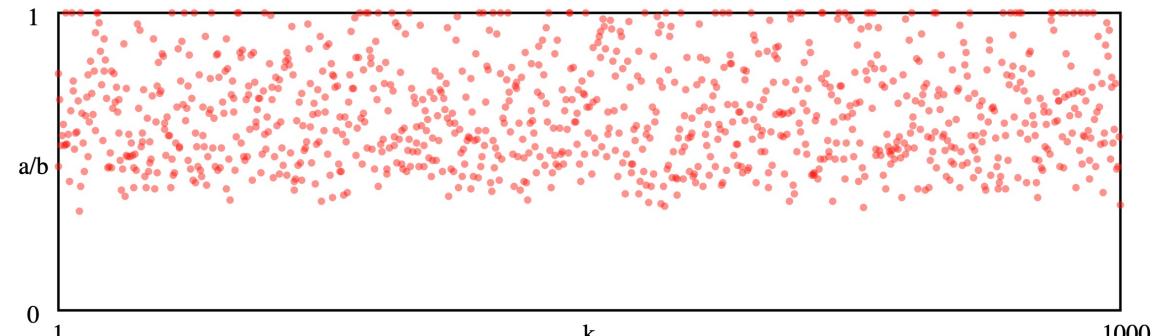
解法1

Y軸合わせるように縦に動いてから
X軸合わせるように横に動くだけ



緑:最適な経路
茶:出力した経路

```
1  query_num = 1000
2  for _ in range(query_num):
3      si_k, sj_k, ti_k, tj_k = map(int, input().split())
4
5      ans = ""
6
7      #縦に動く
8      if si_k < ti_k: #下
9          ans += "D" * (ti_k - si_k)
10     else: #左
11         ans += "U" * (si_k - ti_k)
12
13     #横に動く
14     if sj_k < tj_k: #右
15         ans += "R" * (tj_k - sj_k)
16     else: #左
17         ans += "L" * (sj_k - tj_k)
18
19     print(ans)
20     path_length = input() #出力したバスの長さ * ノイズ
```



クエリごとのa/bの値

解法ごとの スコア表 (1000ケース)

| 解法 | スコア平均 |
|---------|---------------|
| 解法1(縦横) | 628830730.865 |
| ? | ? |
| ? | ? |
| ? | ? |
| ? | ? |
| ? | ? |
| ? | ? |

解法2(Dijkstra法)

各辺に(合計評価値, 評価回数)という情報を持たせる

クエリで渡したパス p_k , 経路長として得た $c_k := b_k \times \text{ノイズ}$

各辺 $e (\in p_k)$ に評価値 $d_e := \frac{c_k}{|p_k|}$ を与える

※ b_k : パス p_k の経路長
 $|p_k|$: p_k に含まれる辺の数

各辺 $e (\in p_k)$ は, 合計評価値に d_e を足し, 評価回数を1増やす.

解法2(Dijkstra法)

各辺の重み w_e を評価値の平均値に設定する.

ただし, 評価回数が0回である辺は定数init_costを用いる.

$$w_e = \begin{cases} \frac{\text{合計評価値}}{\text{評価回数}} & (\text{評価回数} \neq 0) \\ \text{init_cost} & (\text{評価回数} = 0) \end{cases}$$

例)

各辺がもっている情報

| 辺番号 | … | 5 | 6 | 7 | … |
|-----|---|-------|--------|--------|---|
| 情報 | … | (3,1) | (12,3) | (10,2) | … |

各辺の情報から計算した重み

| 辺番号 | … | 5 | 6 | 7 | … |
|-----|---|---|---|---|---|
| 重み | … | 3 | 4 | 5 | … |

解法2(Dijkstra法)

各辺の重み w_e を評価値の平均値に設定する。

ただし、評価回数が0回である辺は定数init_costを用いる。

$$w_e = \begin{cases} \frac{\text{合計評価値}}{\text{評価回数}} & (\text{評価回数} \neq 0) \\ \text{init_cost} & (\text{評価回数} = 0) \end{cases}$$

例)

各辺がもっている情報

| 辺番号 | ・ | ・ | ・ | 5 | 6 | 7 | ・ | ・ | ・ |
|-----|---|---|---|-------|--------|--------|---|---|---|
| 情報 | ・ | ・ | ・ | (3,1) | (12,3) | (10,2) | ・ | ・ | ・ |

各辺の情報から計算した重み

| 辺番号 | ・ | ・ | ・ | 5 | 6 | 7 | ・ | ・ | ・ |
|-----|---|---|---|---|---|---|---|---|---|
| 重み | ・ | ・ | ・ | 3 | 4 | 5 | ・ | ・ | ・ |

解法2(Dijkstra法)

init_costの設定

案1. $\text{init_cost} = \infty$

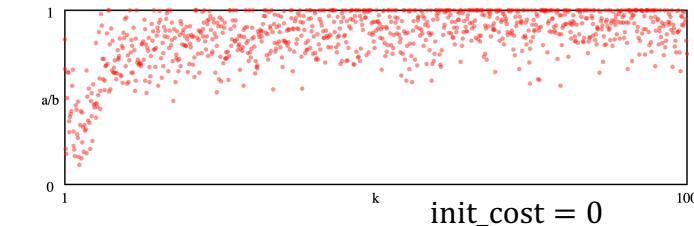
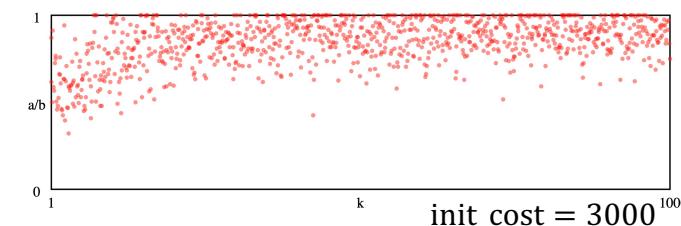
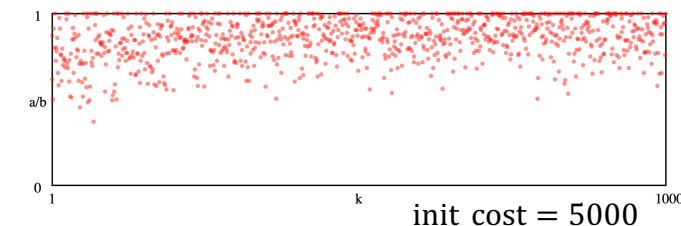
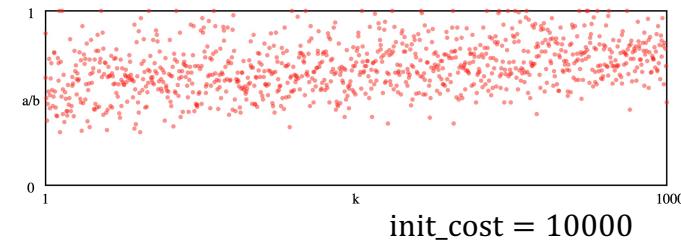
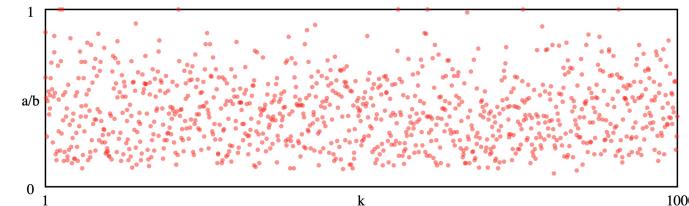
案2. $\text{init_cost} = 10000$

案3. $\text{init_cost} = 5000$

案4. $\text{init_cost} = 3000$

案5. $\text{init_cost} = 0$

| init_cost | スコア平均 |
|-----------|-------------|
| ∞ | 427,870,486 |
| 10000 | 706,245,146 |
| 5000 | 877,264,576 |
| 3000 | 897,599,285 |
| 0 | 880,147,096 |



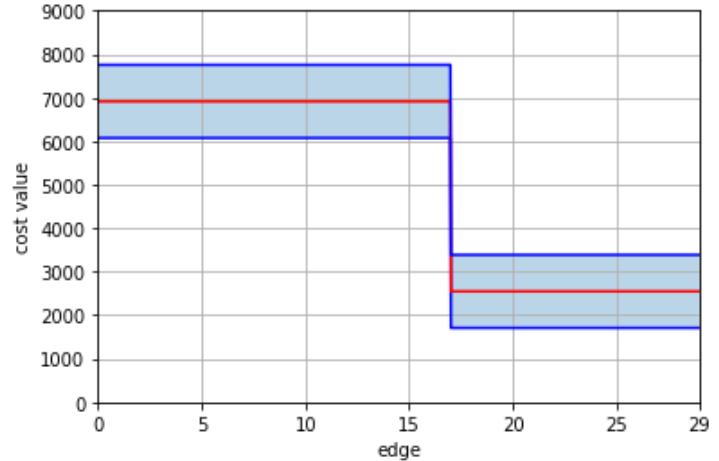
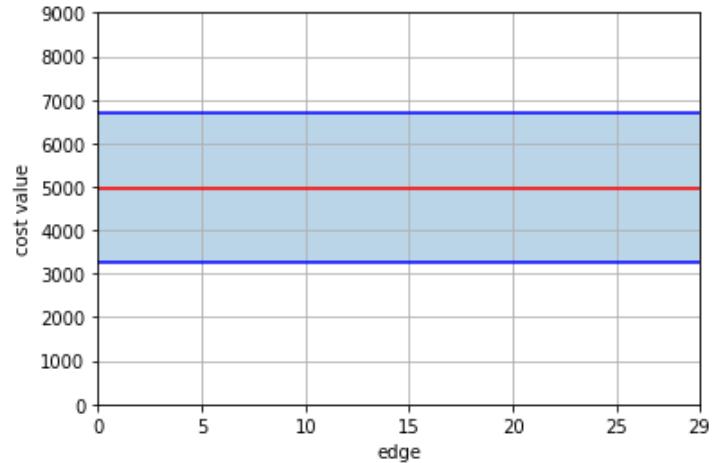
解法ごとのスコア表(1000ケース)

| 解法 | スコア平均 |
|----------------|-------------|
| 解法1(縦横) | 628,830,730 |
| 解法2(Dijkstra法) | 897,599,285 |
| ? | ? |
| ? | ? |
| ? | ? |
| ? | ? |
| ? | ? |

重要な考察

- $M = 1$ のとき, 横(縦)向き同じ行(列)の辺の重みは近い値になっている (2辺の差は高々 $2 \times D$)
- $M = 2$ のとき, 横(縦)向き同じ行(列)の**近い**辺の重みは近い値になっている可能性が高い

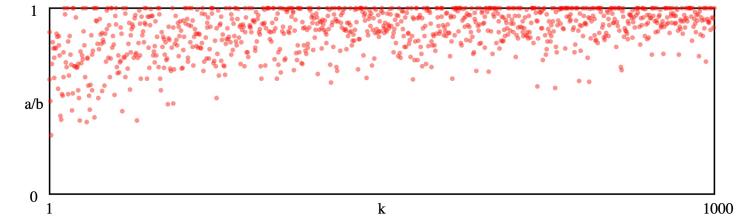
$M = 1$ のときの辺コストの例



$M = 2$ のときの辺コストの例

解法3(未使用辺は近くの辺評価)

- 未使用辺は一番近くの辺の情報(合計評価値, 評価回数)で重み計算
- 一番近くの辺が二つあったら, それぞれの情報を足して使用する



例)

各辺がもっている情報

| 辺番号 | ・ | ・ | ・ | 5 | 6 | 7 | 8 | 9 | ・ | ・ | ・ |
|-----|---|---|---|-------|-------|-------|-------|-------|---|---|---|
| 情報 | ・ | ・ | ・ | (1,1) | (0,0) | (0,0) | (0,0) | (3,1) | ・ | ・ | ・ |

各辺の重み計算に使う情報

| 辺番号 | ・ | ・ | ・ | 5 | 6 | 7 | 8 | 9 | ・ | ・ | ・ |
|-----|---|---|---|-------|-------|-------|-------|-------|---|---|---|
| 情報 | ・ | ・ | ・ | (1,1) | (1,1) | (4,2) | (3,1) | (3,1) | ・ | ・ | ・ |

解法ごとのスコア表(1000ケース)

| 解法 | スコア平均 |
|------------------|-------------|
| 解法1(縦横) | 628,830,730 |
| 解法2(Dijkstra法) | 897,599,285 |
| 解法3(未使用辺は近くの辺評価) | 903,153,319 |
| ? | ? |
| ? | ? |
| ? | ? |
| ? | ? |

解法4(近くの辺を加味して評価)

- 同じ行(列)の辺で、近くの辺は辺重みが近い値になる傾向にある。



- 同じ行(列)でユークリッド距離が定数 l 以下の辺の情報を合わせて評価する($l = 8$ がgood)

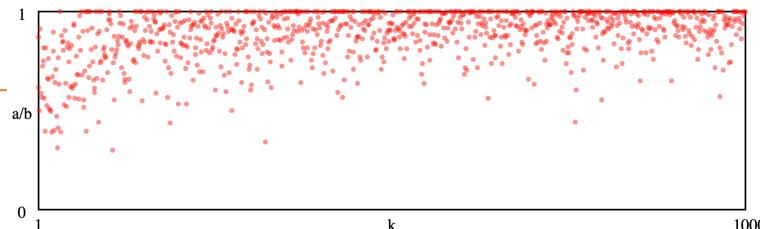
例) $l = 1$

各辺がもっている情報

| 辺番号 | … | 5 | 6 | 7 | 8 | 9 | 10 | … |
|-----|---|-------|-------|-------|-------|-------|-------|---|
| 情報 | … | (0,0) | (2,4) | (3,3) | (3,3) | (3,1) | (0,0) | … |

各辺の重み計算に使う情報

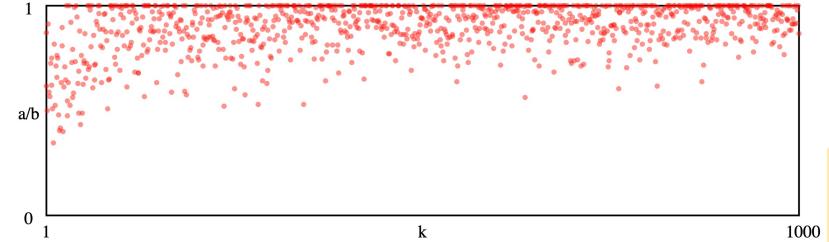
| 辺番号 | … | 5 | 6 | 7 | 8 | 9 | 10 | … |
|-----|---|-------|-------|--------|-------|-------|-------|---|
| 情報 | … | (?,?) | (5,7) | (8,10) | (9,7) | (6,4) | (?,?) | … |



解法ごとのスコア表(1000ケース)

| 解法 | スコア平均 |
|------------------|-------------|
| 解法1(L字) | 628,830,730 |
| 解法2(Dijkstra法) | 897,599,285 |
| 解法3(未使用辺は近くの辺評価) | 903,153,319 |
| 解法4(近くの辺を加味して評価) | 921,744,234 |
| ? | ? |
| ? | ? |
| ? | ? |

解法5（自身の辺情報を重要視）



- 解法4では、近くの辺と自身の辺を同じ重みで足していた。
- でも他の辺より自分の辺の情報の方がより正確だろう。



- 自身の辺の情報の重みを r 倍して計算する。 $(r = 6$ がgood)

例) $l = 1, r = 6$

各辺がもっている情報

| 辺番号 | ... | 5 | 6 | 7 | 8 | 9 | 10 | ... |
|-----|-----|-------|-------|-------|-------|-------|-------|-----|
| 情報 | ... | (0,0) | (2,4) | (3,3) | (3,3) | (3,1) | (0,0) | ... |

各辺の重み計算に使う情報

| 辺番号 | ... | 5 | 6 | 7 | 8 | 9 | 10 | ... |
|-----|-----|-------|---------|---------|---------|--------|-------|-----|
| 情報 | ... | (?,?) | (15,27) | (23,25) | (24,22) | (21,9) | (?,?) | ... |

解法ごとのスコア表(1000ケース)

| 解法 | スコア平均 |
|------------------|-------------|
| 解法1(L字) | 628,830,730 |
| 解法2(Dijkstra法) | 897,599,285 |
| 解法3(未使用辺は近くの辺評価) | 903,153,319 |
| 解法4(近くの辺を加味して評価) | 921,744,234 |
| 解法5(自身の辺情報を重視) | 927,969,469 |
| ? | ? |
| ? | ? |

重要な考察

図1より、図2の方が1辺が得る情報の正確さは高くなりそう

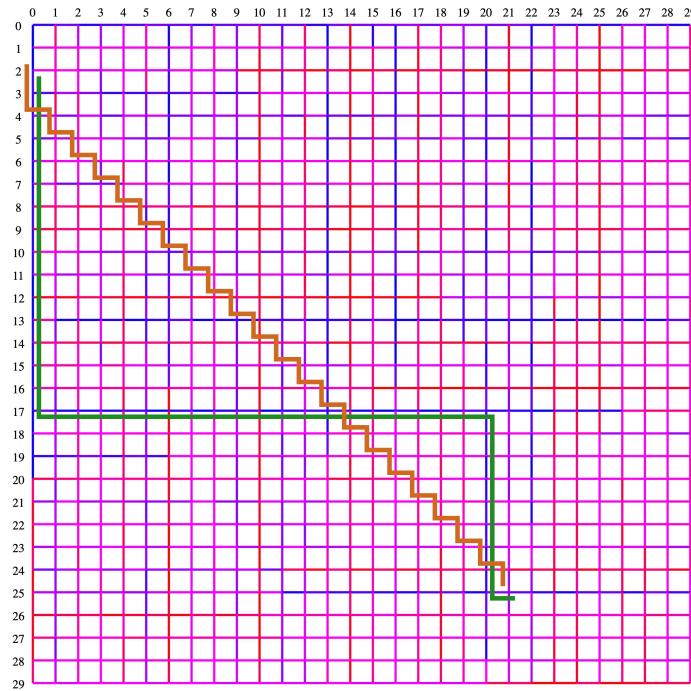


図1

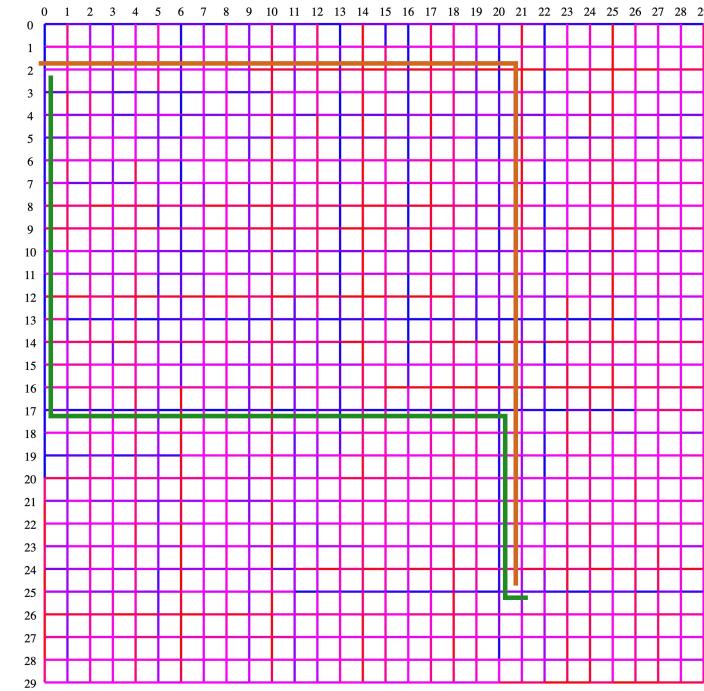


図2

緑:最適な経路 茶:出力した経路

解法6(パス内の同じ列/行の長さで重み付け)

- 同じ列/行の

解法ごとのスコア表(1000ケース)

| 解法 | スコア平均 |
|------------------|-------------|
| 解法1(L字) | 628,830,730 |
| 解法2(Dijkstra法) | 897,599,285 |
| 解法3(未使用辺は近くの辺評価) | 903,153,319 |
| 解法4(近くの辺を加味して評価) | 921,744,234 |
| 解法5(自身の辺情報を重視) | 927,969,469 |
| ? | ? |
| ? | ? |

解法7(パス内の同じ列/行の長さで重み付け(二乗))

解法ごとのスコア表(1000ケース)

| 解法 | スコア平均 |
|------------------|-------------|
| 解法1(L字) | 628,830,730 |
| 解法2(Dijkstra法) | 897,599,285 |
| 解法3(未使用辺は近くの辺評価) | 903,153,319 |
| 解法4(近くの辺を加味して評価) | 921,744,234 |
| 解法5(自身の辺情報を重視) | 927,969,469 |
| ? | ? |
| ? | ? |

解法8(始めの方は情報を得るためのパス)

解法ごとのスコア表(1000ケース)

| 解法 | スコア平均 |
|------------------|-------------|
| 解法1(L字) | 628,830,730 |
| 解法2(Dijkstra法) | 897,599,285 |
| 解法3(未使用辺は近くの辺評価) | 903,153,319 |
| 解法4(近くの辺を加味して評価) | 921,744,234 |
| 解法5(自身の辺情報を重視) | 927,969,469 |
| ? | ? |
| ? | ? |