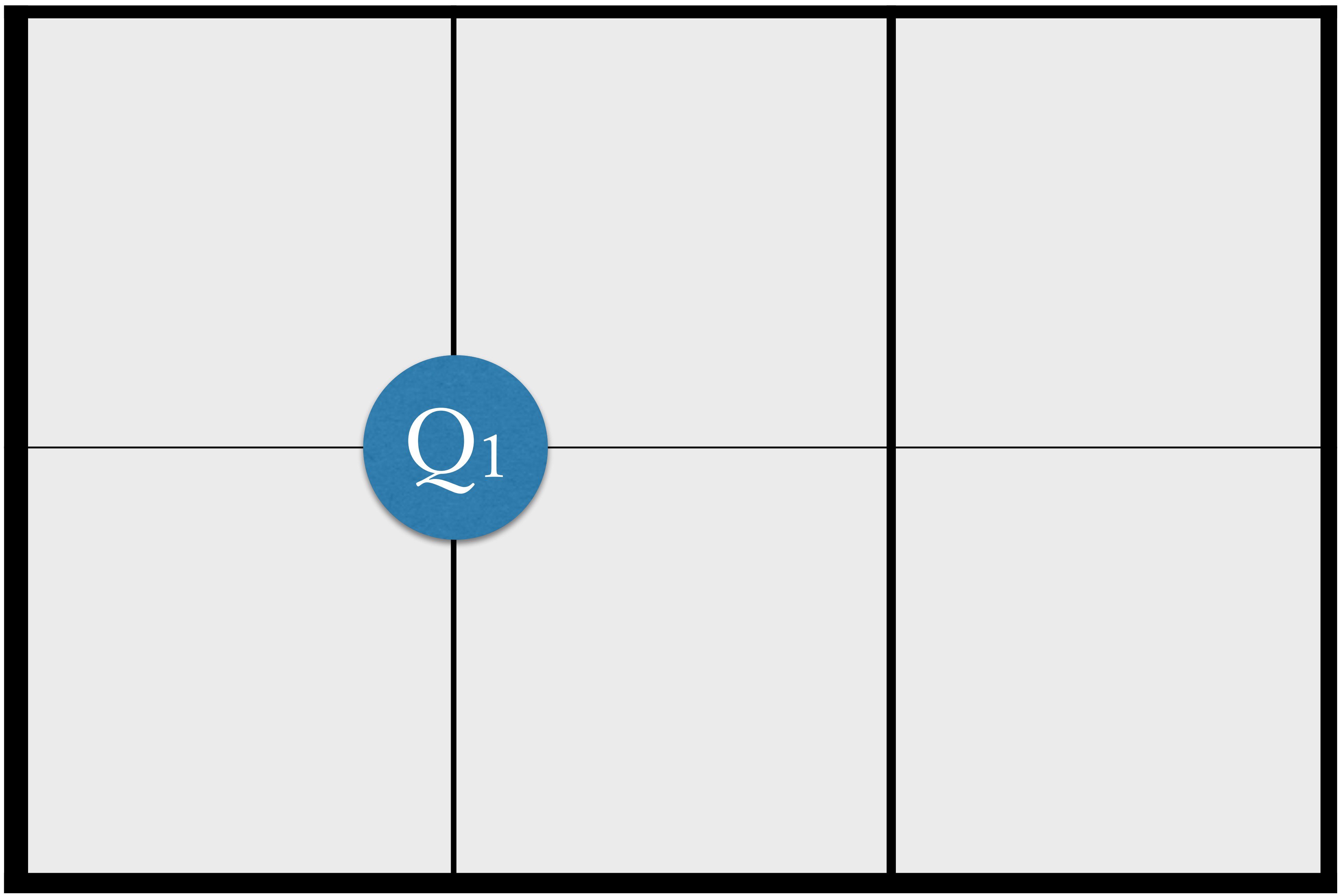
Abduction 2 解説

tozangezan

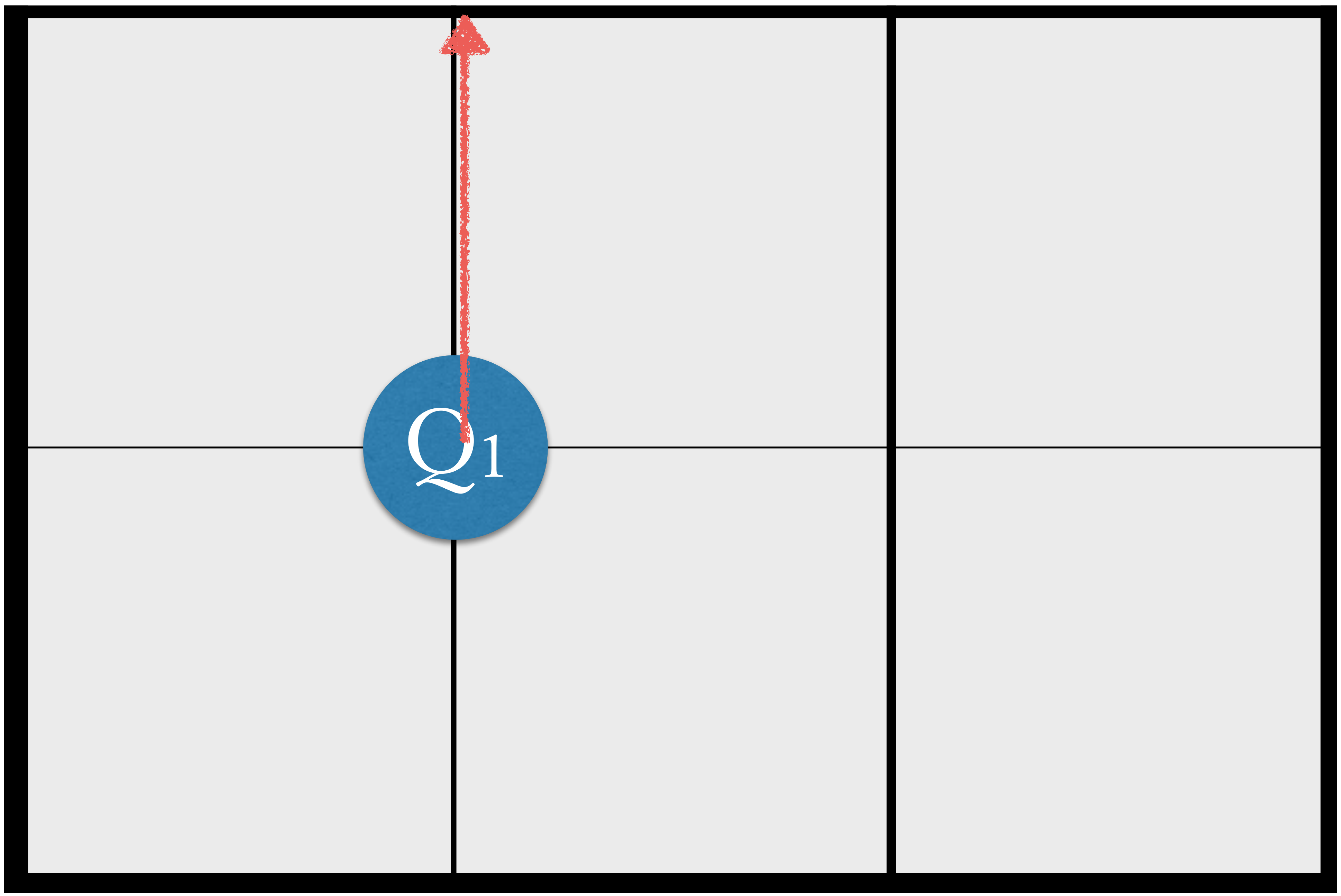
問題概要

- H * W のグリッドの辺上を、ある始点から移動を行う
- 最初はどの方向にでも移動できる
- 交差点に差し掛かった時、より人通りの多い道を選んで(曲がる/直進)する
- Q 個の始点が与えられた時、移動可能な最長経路の長さを求めよ

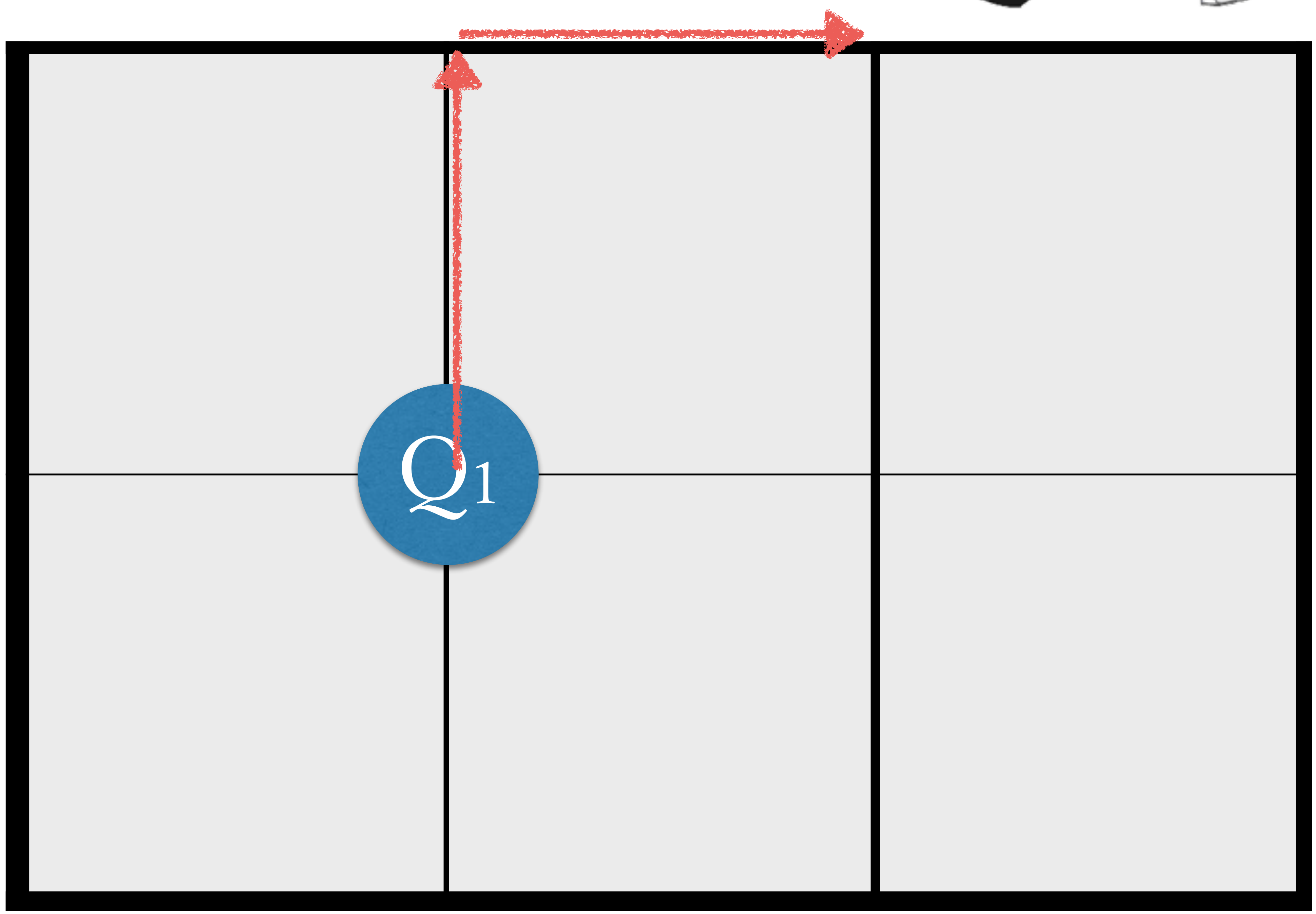
例



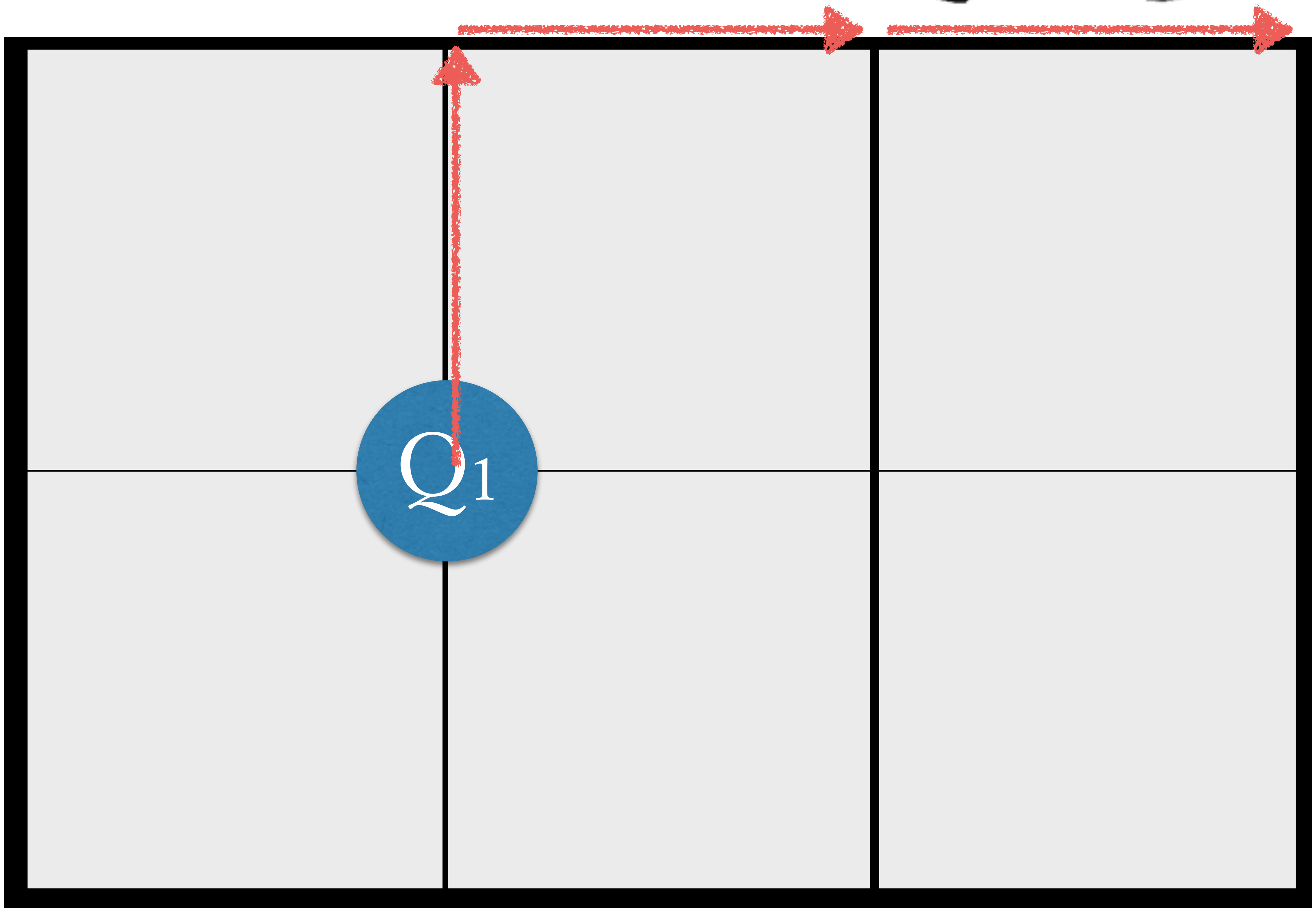
例



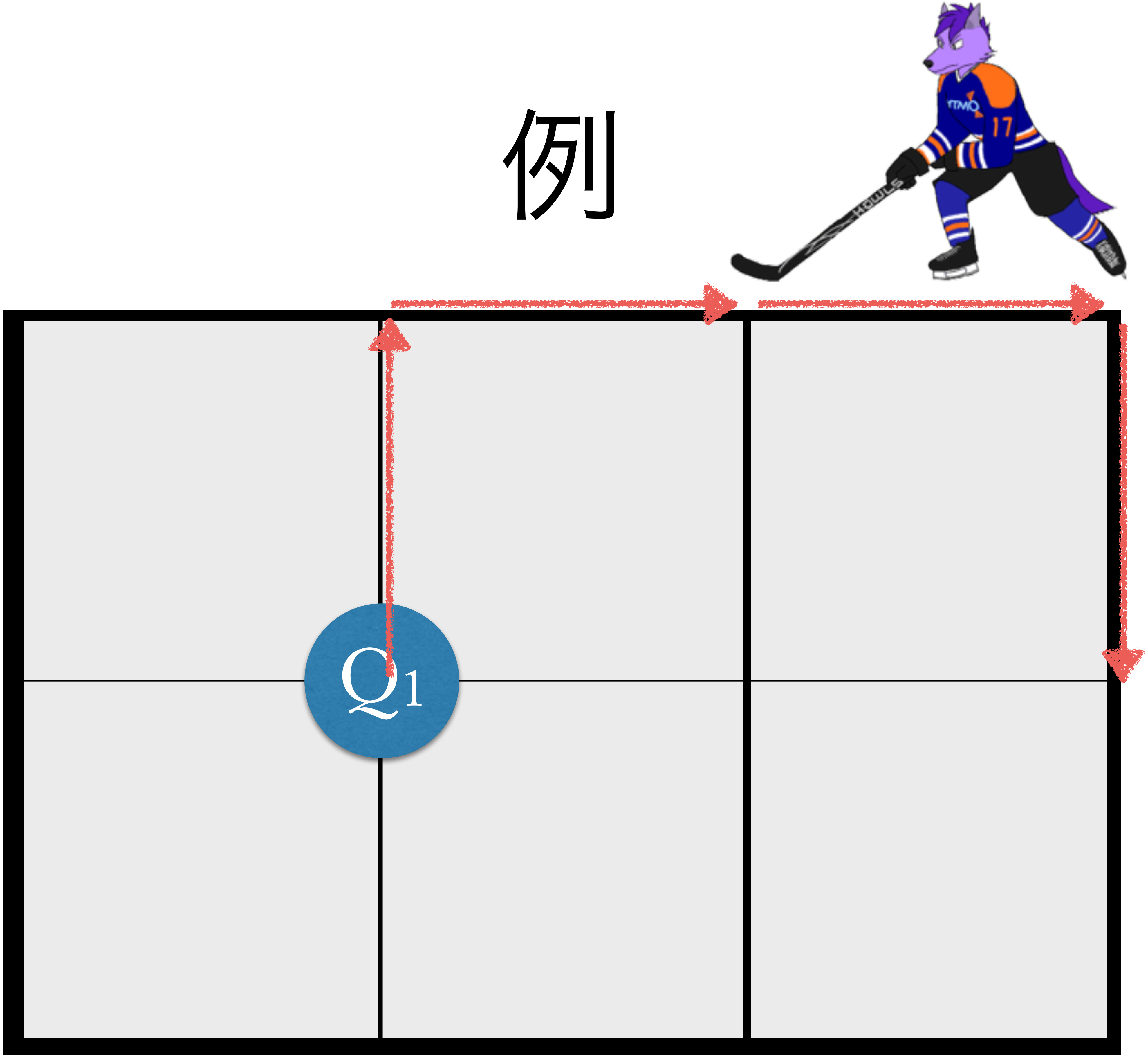
例



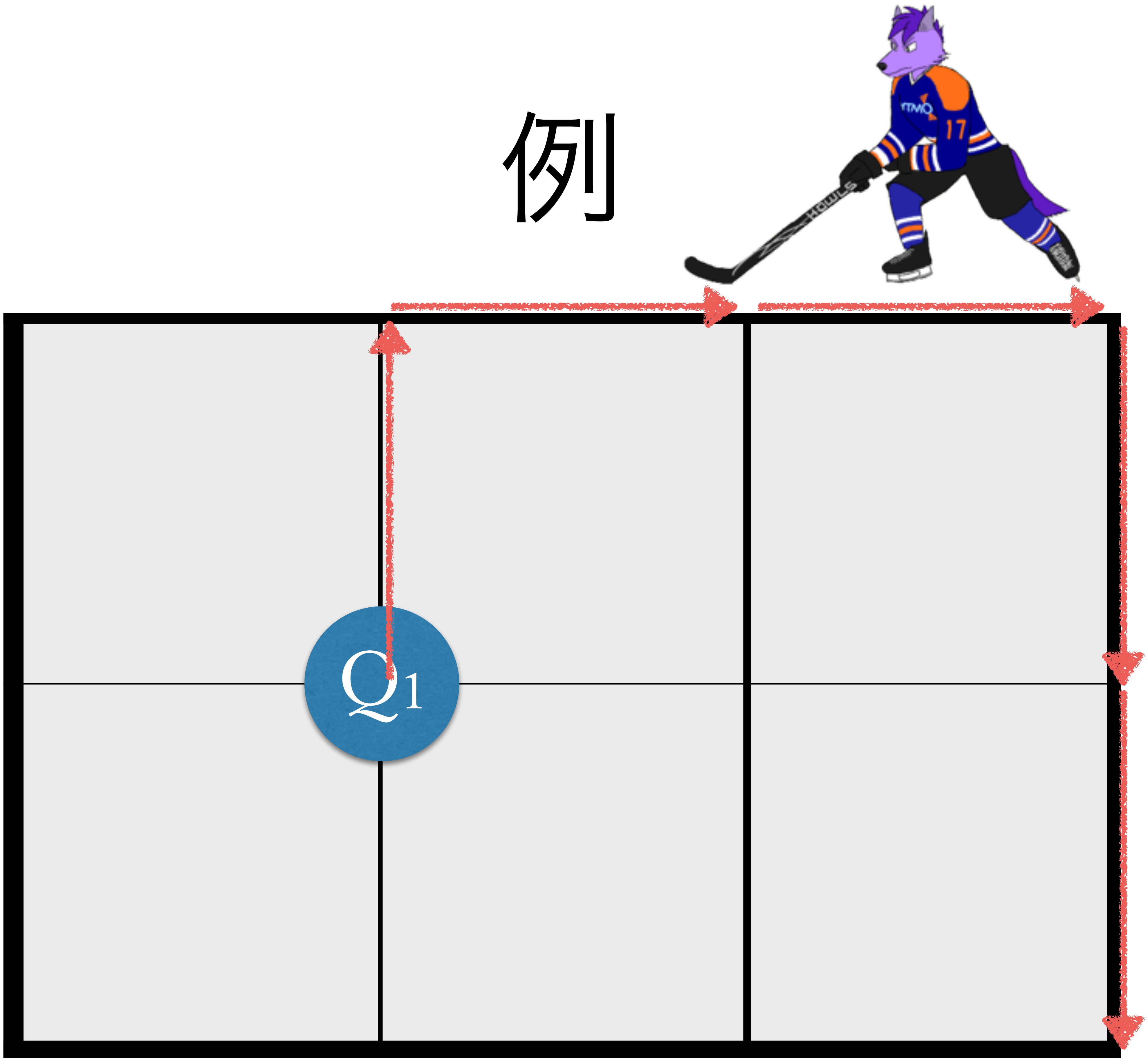
例



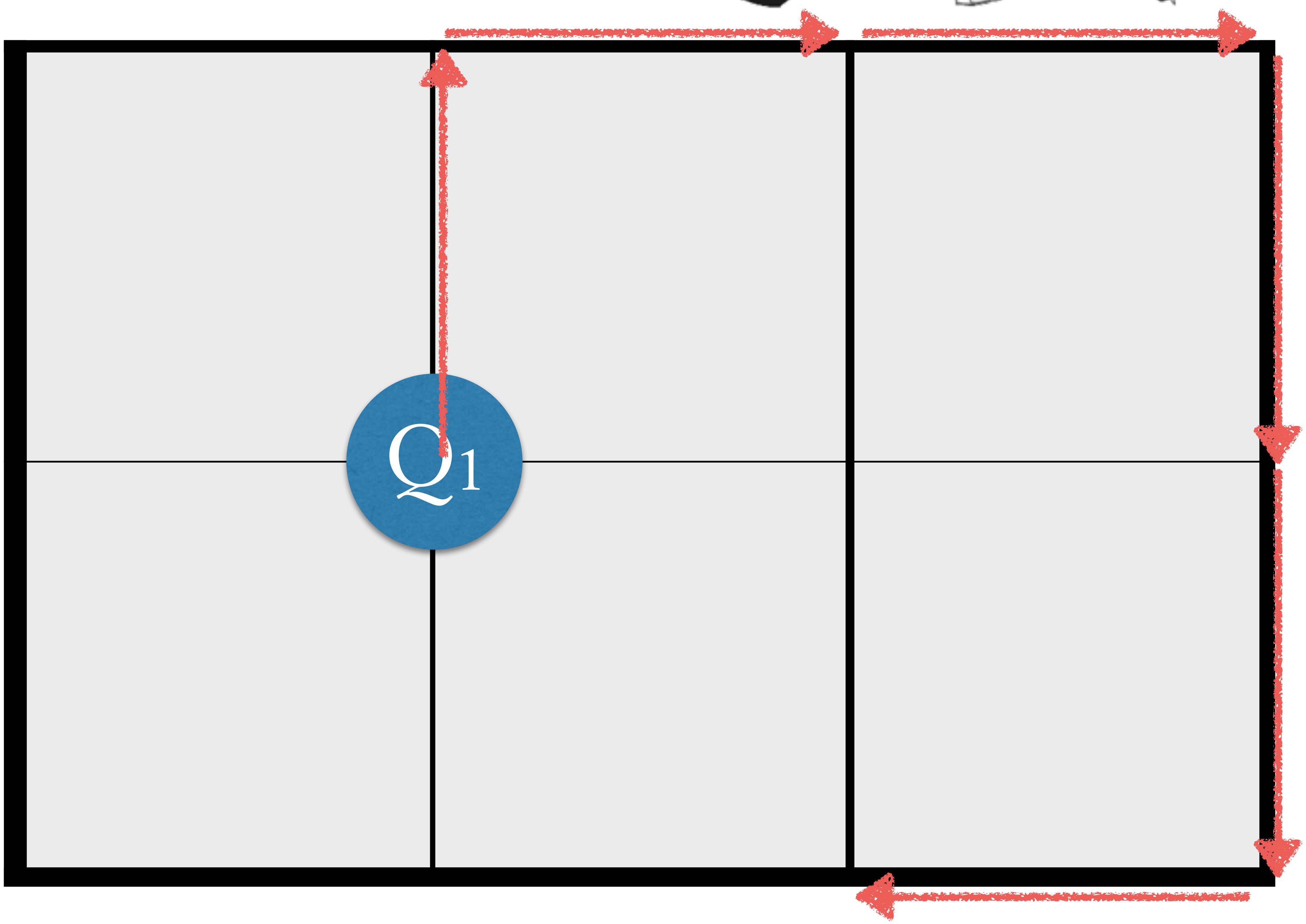
例



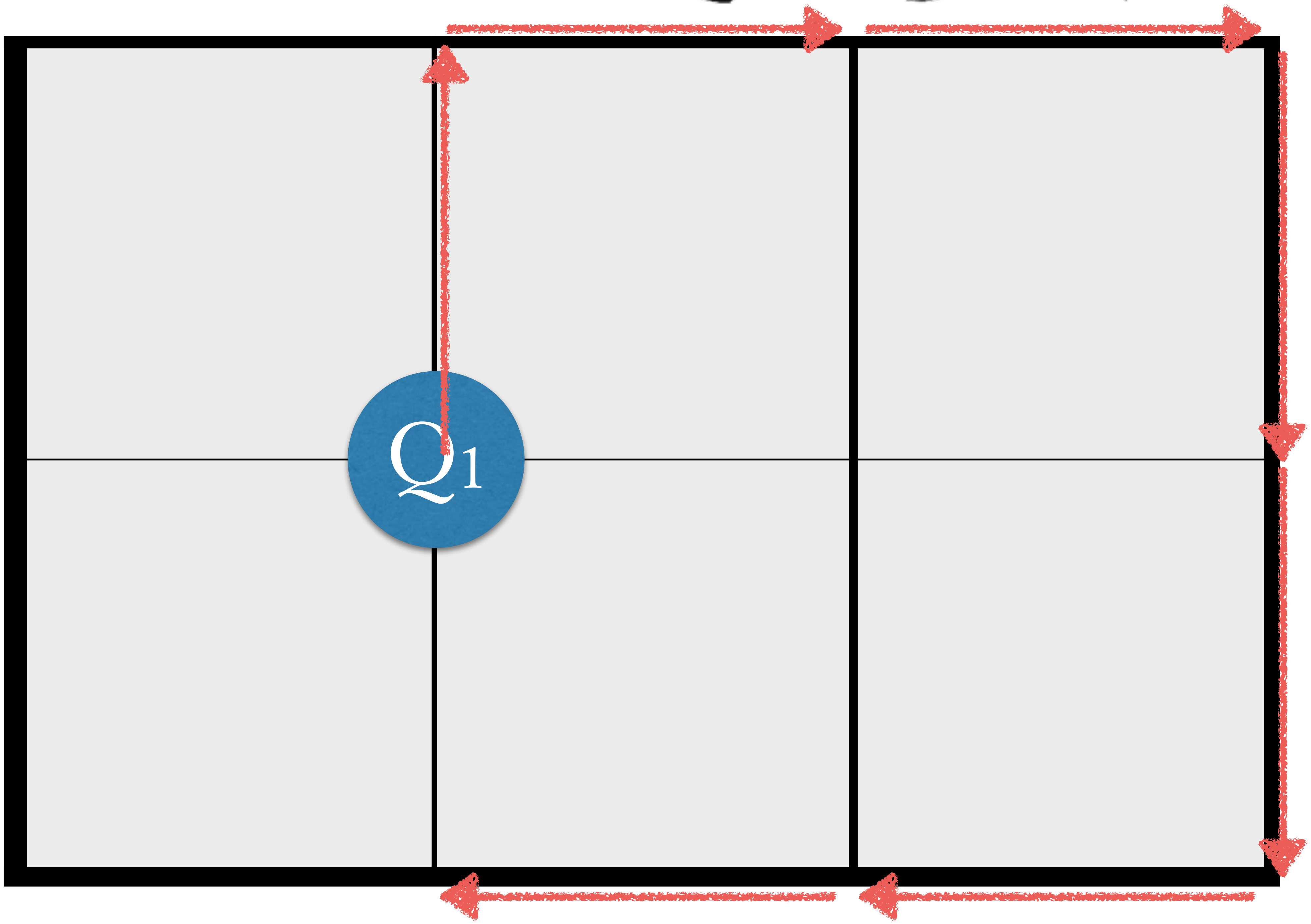
例



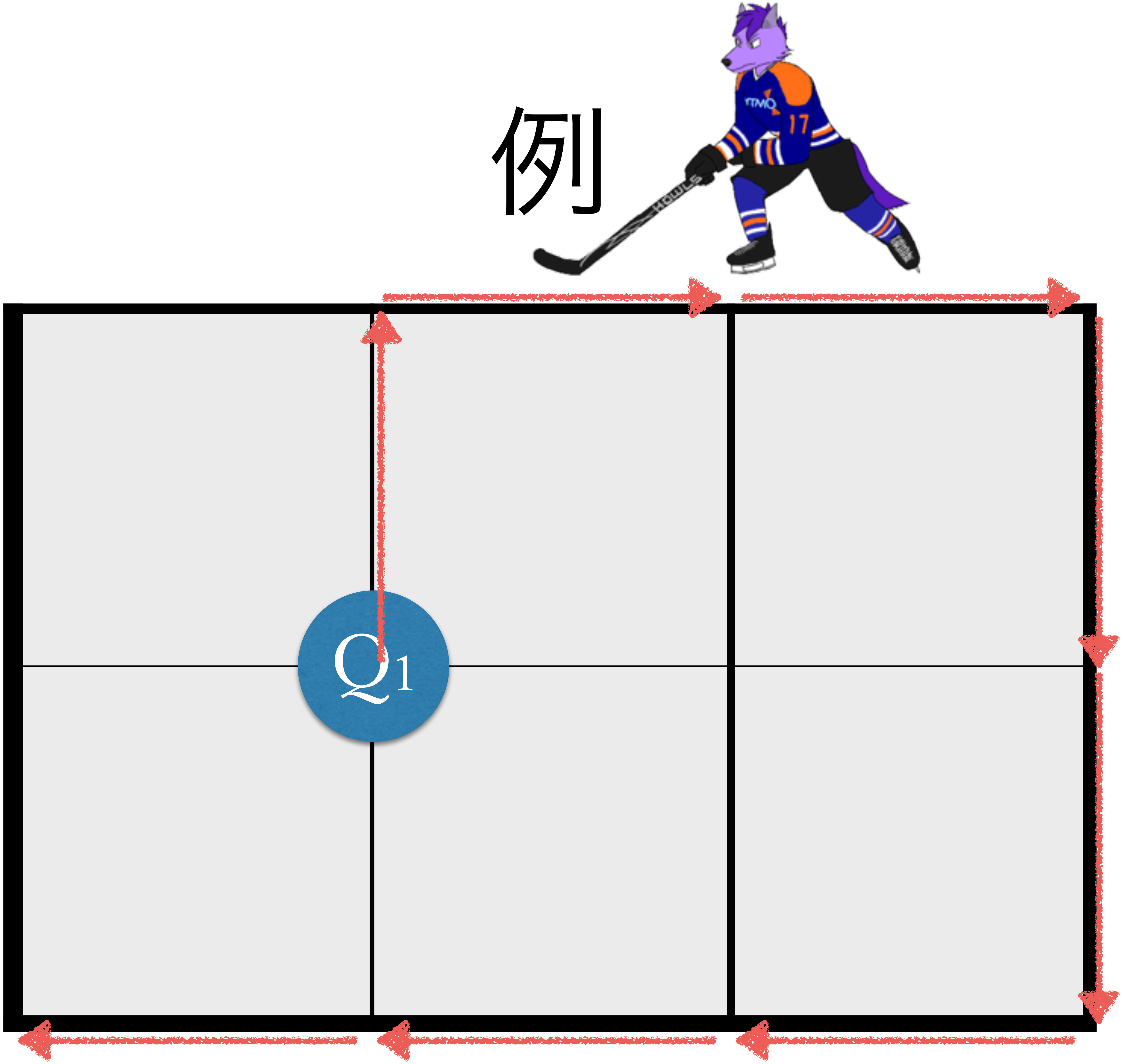
例



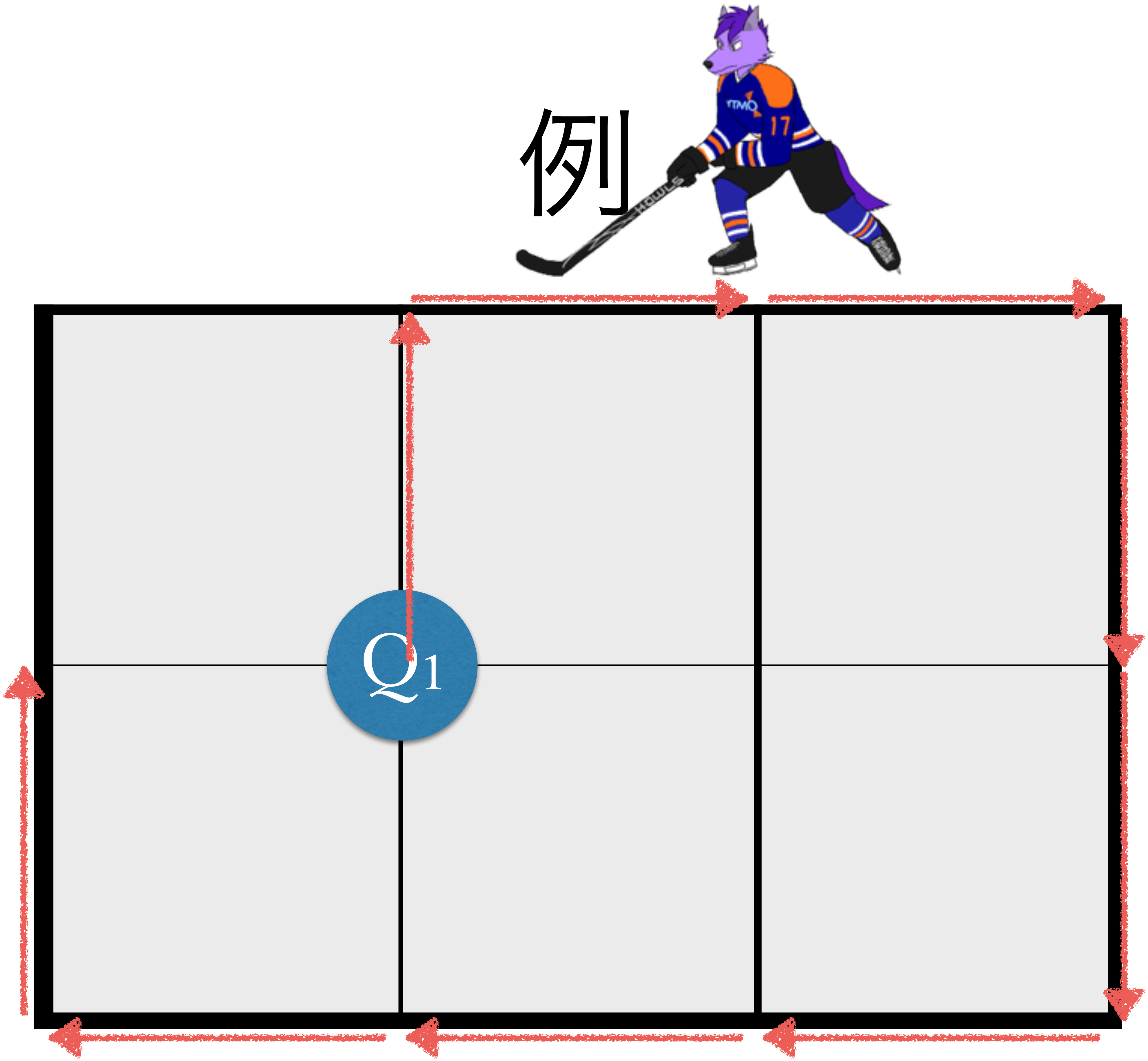
例



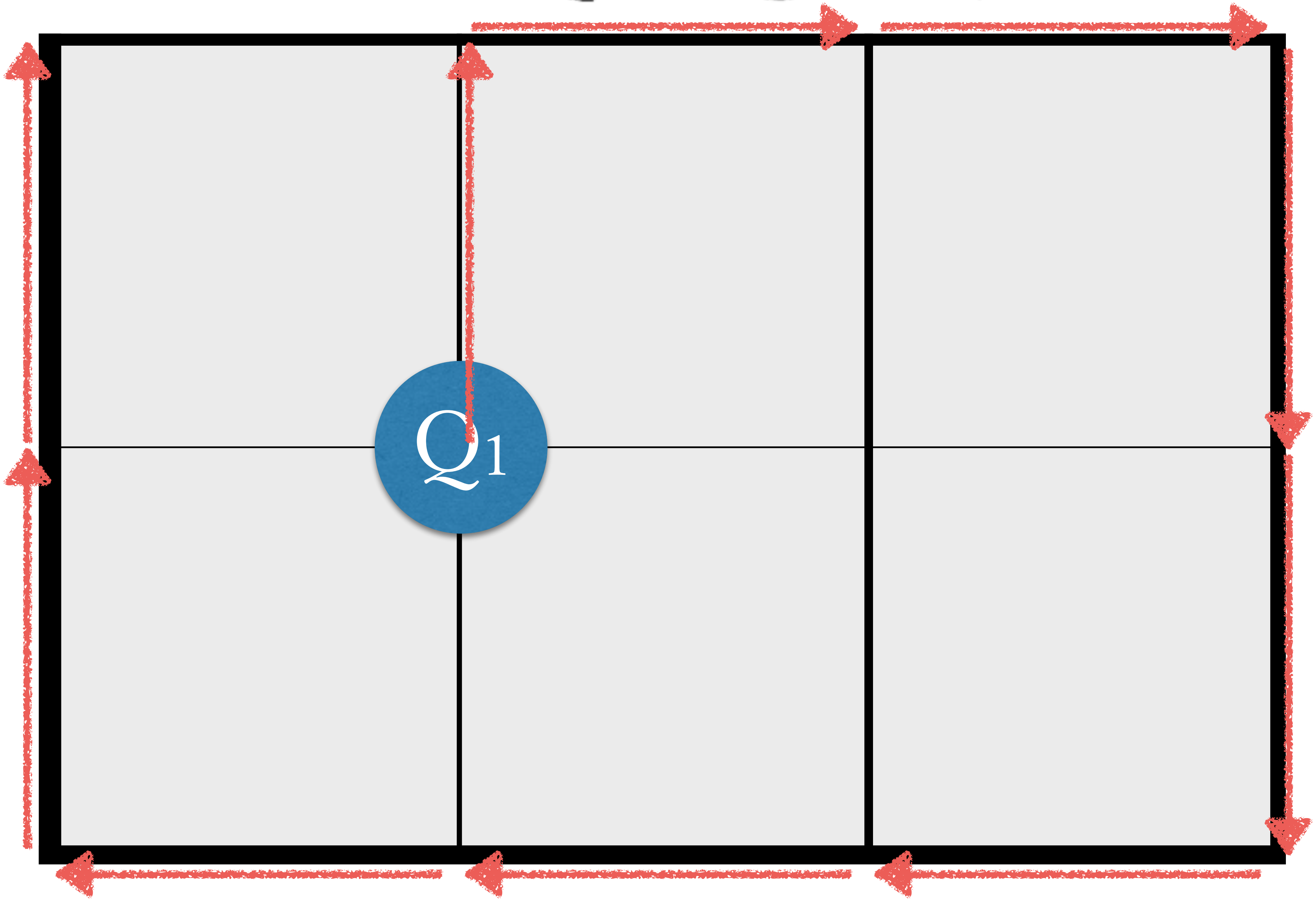
例



例

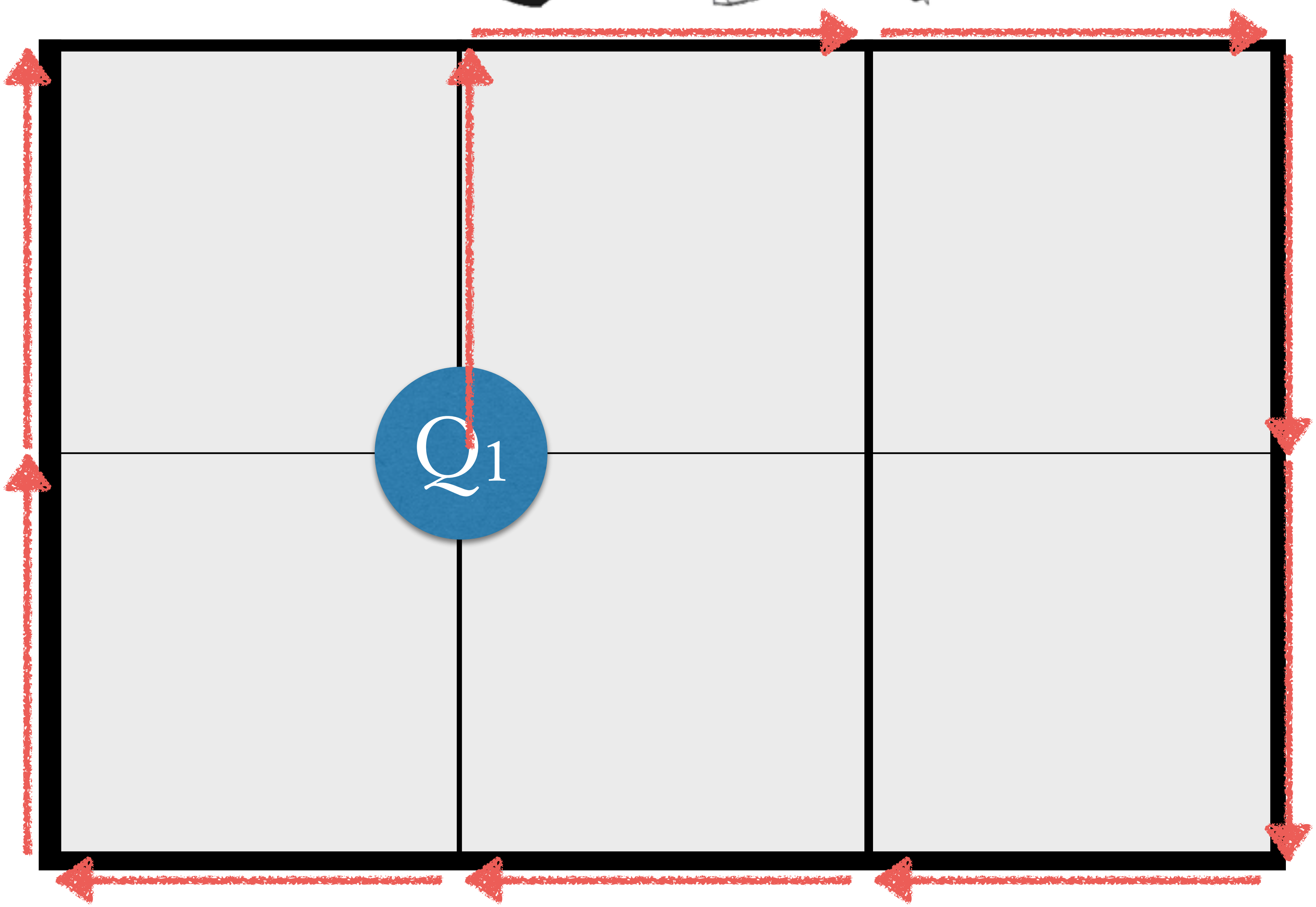


例



答え: 10

例



解法 1

- DFSする
- solve(i,j,方向): 交差点(i,j)からある方向を向いて出発したときの最大値
- solve(i,j,方向)=max(solve(i’,j’,方向’$_1$),solve(i’,j’,方向’$_2$))+1 的な感じ
- 行き止まりでは return 0;

何点取れるでしょうか？

13点

計算量？

- 13点: H, W ≤ 8, Q = 1
- 分岐する回数は多めに見積もっても 16 回くらい
- いつでも 2 通りに分岐できたとしても 216 通りくらい
- 全通りを試しても余裕で間に合う

解法 2

- 解法 1 をメモ化する
- Q 回クエリがあるが、メモ化した値はクエリによって変わらないので、DP テーブルをクエリごとに初期化する必要はない

何点取れるでしょうか？

27点

計算量？

- 27点: H, W ≤ 2,000
- 状態数が O(HW) の DP となる
- 遷移は O(1) (隣の交差点に移動するだけ)
- 計算量は O(HW) なので間に合う

解法３

- 直進するだけの交差点をスキップ
- dfs(i,j,方向) からは、その方向に進んで最初に曲がる場所を探し、曲がった先に対し dfs(i',j',方向') を呼ぶ
- どうやってスキップしよう？

Sparse Table

- a[i][j]: [i, i+2^j) における A[i] の最大値
- ↑の表は、ダブリングのようにして計算できる
- ある点からどこまでいけるかも、これを j から大きい順に回していけば移動先が見つかる
- segtree上で二分探索するよりも楽
- 計算量は前処理 O(N log N)、クエリ O(log N)

何点取れるでしょうか？

何点取れるでしょうか？

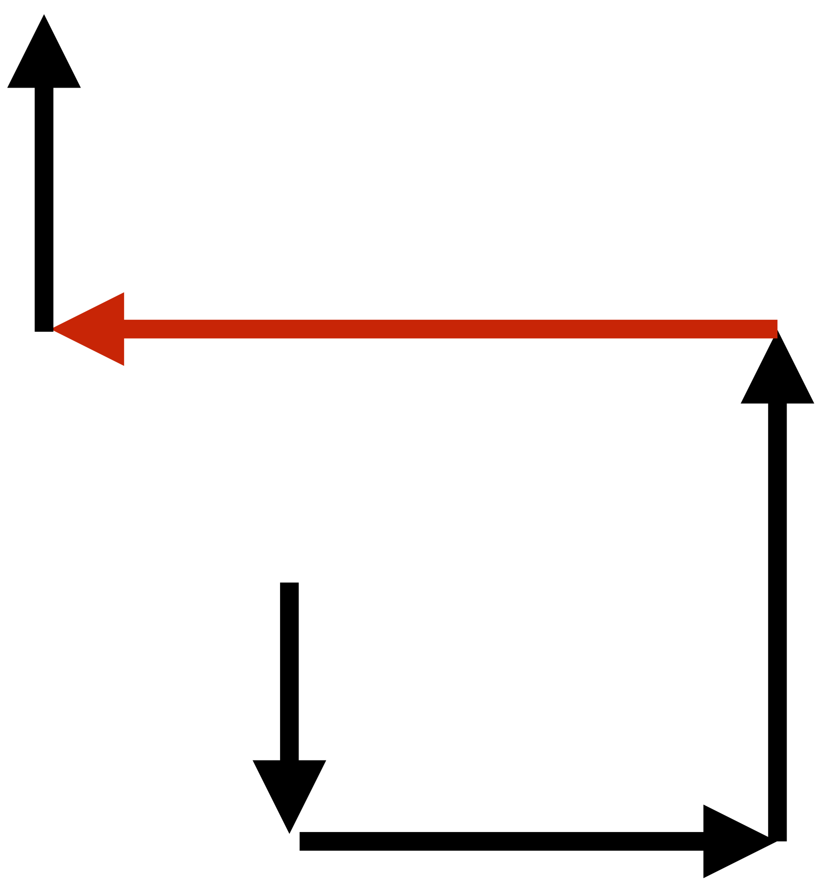
100点

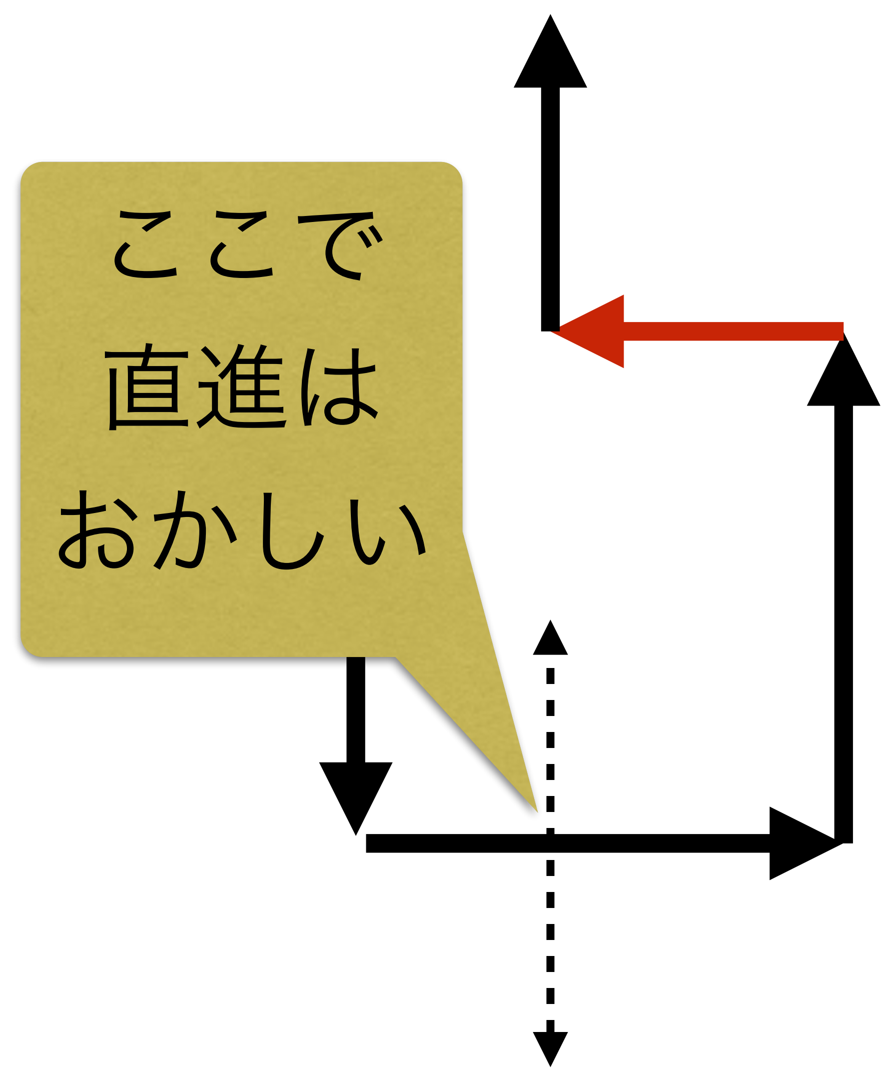
計算量？

- 計算量解析は自明ではない
- とりあえず状態数はいくつだろう？

動き方

- 右のような動き方はありえない

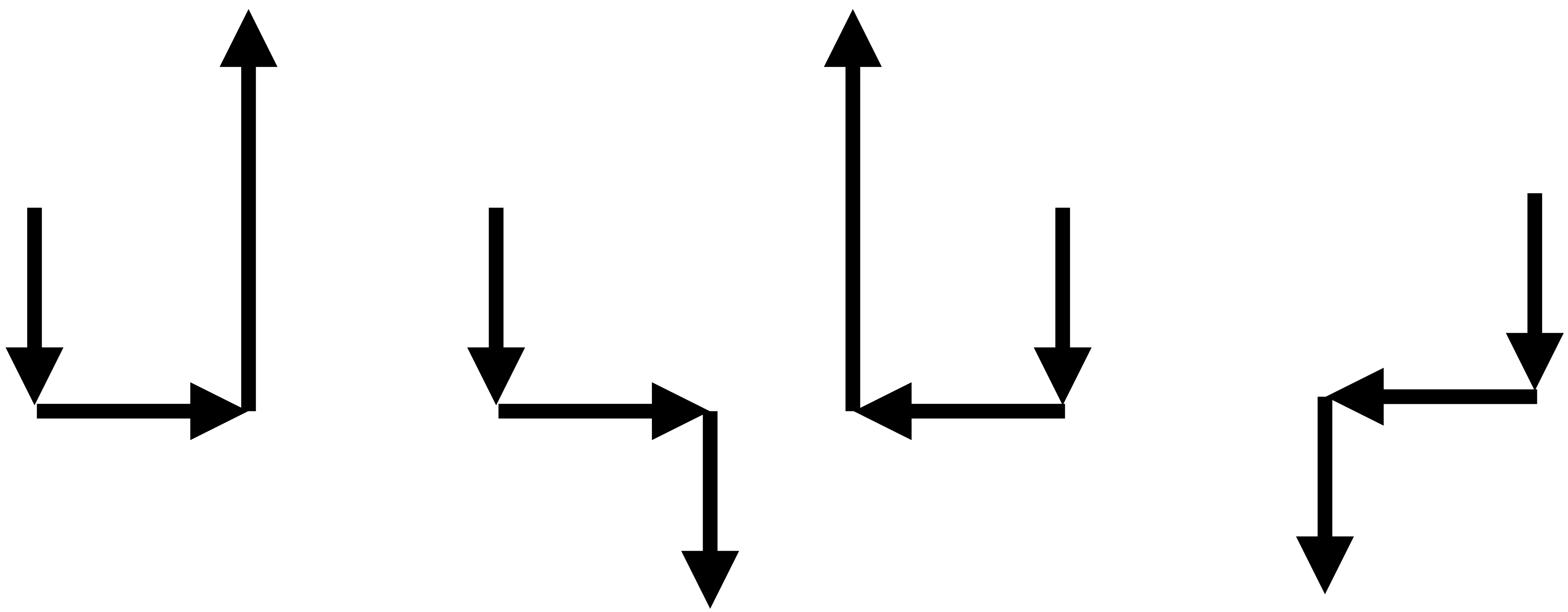




状態数の解析

- DFS で計算していく時、何回 DFS 回数を呼んだ内部での状態かを、「深さ」と言うことにする
- 深さｉまでの状態数は、いくらになるだろう？

例: 深さ 3

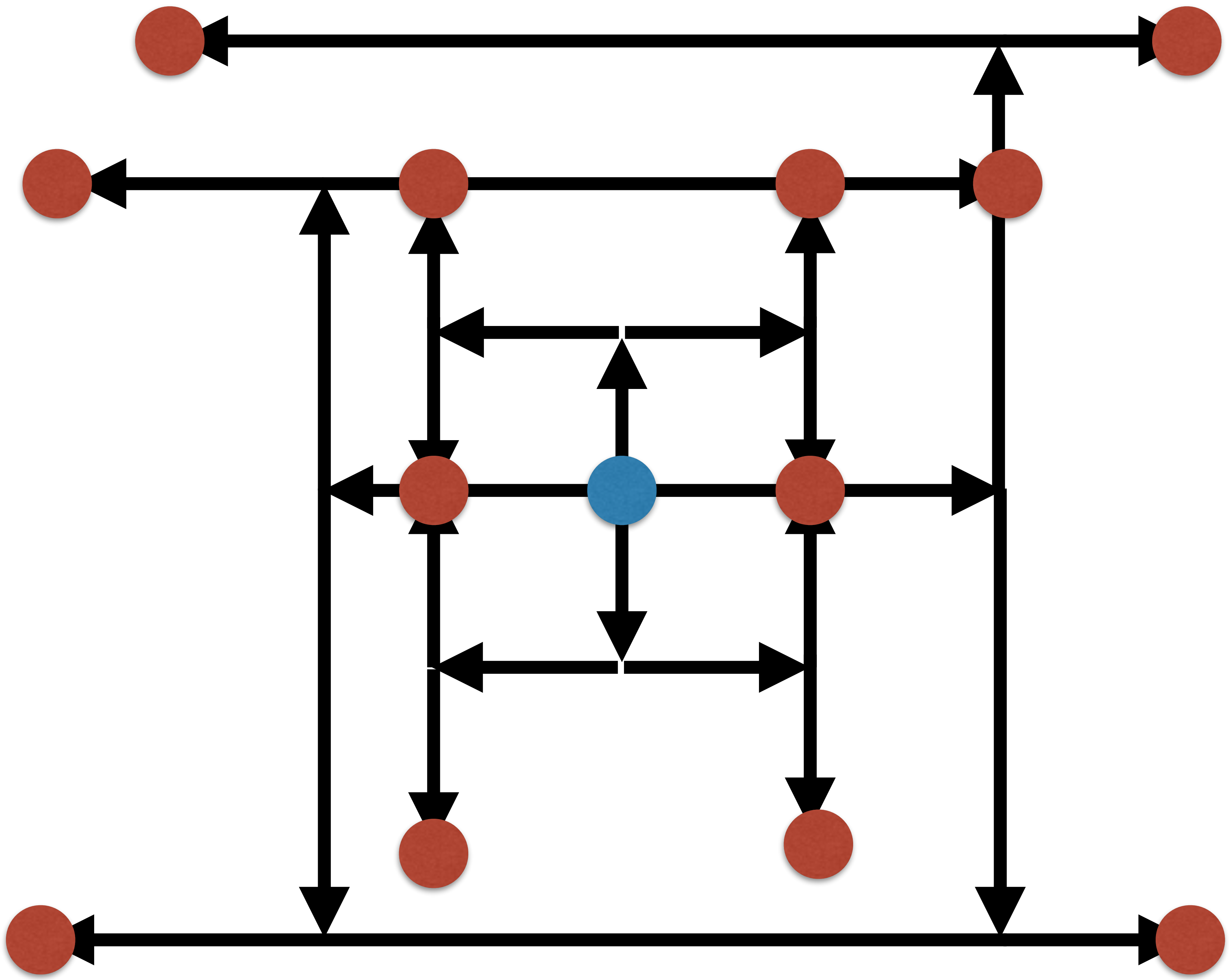


(向きを変えると 4 倍)

例: 深さ 3

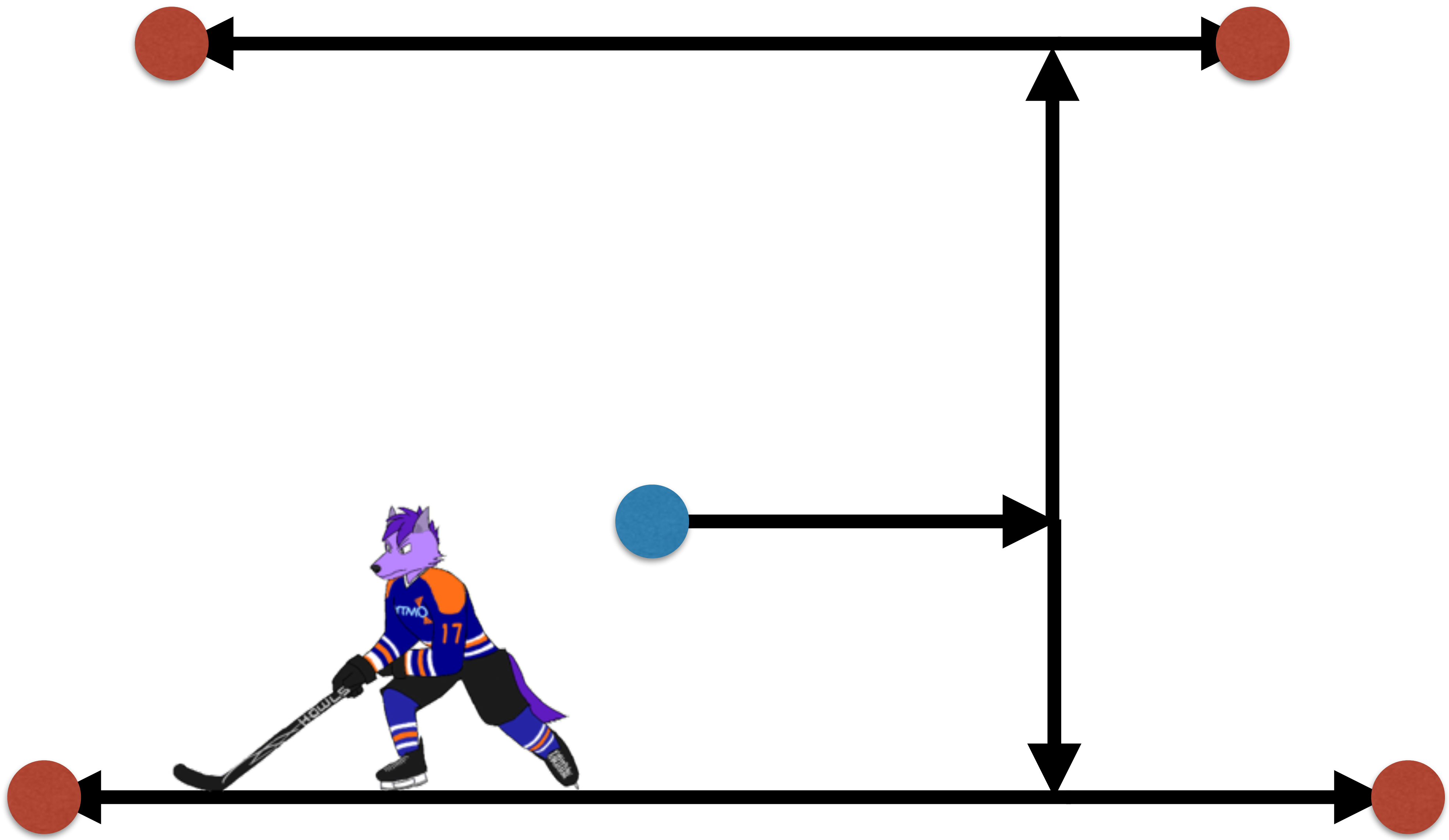
- しかし、 前述のようにメモ化して計算しているので、途中の経路はどうでもよくて、最後何通りの場所に到達できるかが問題
- 行き先は全部で何通り？

例: 深さ 3

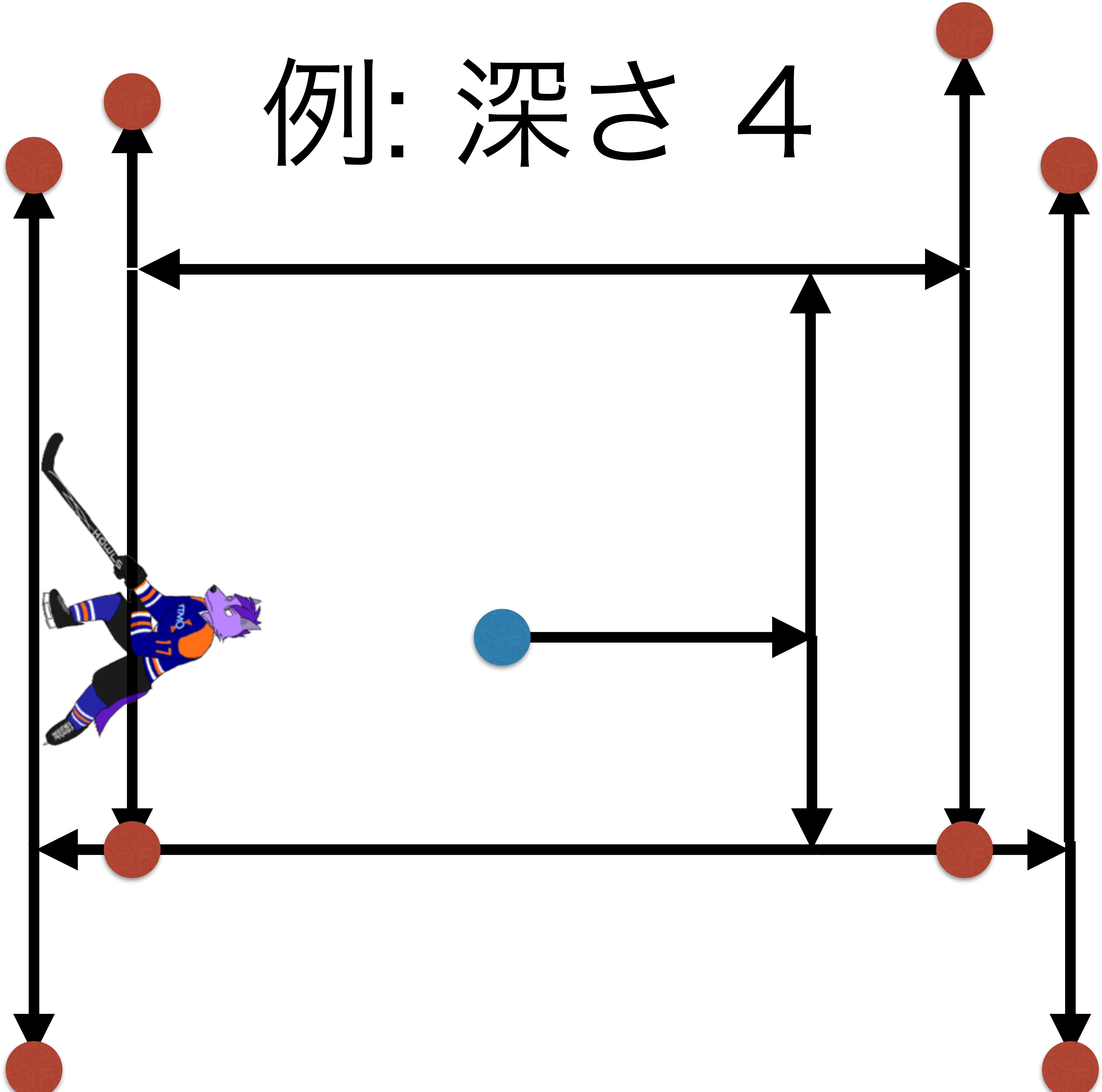


わけがわからないので

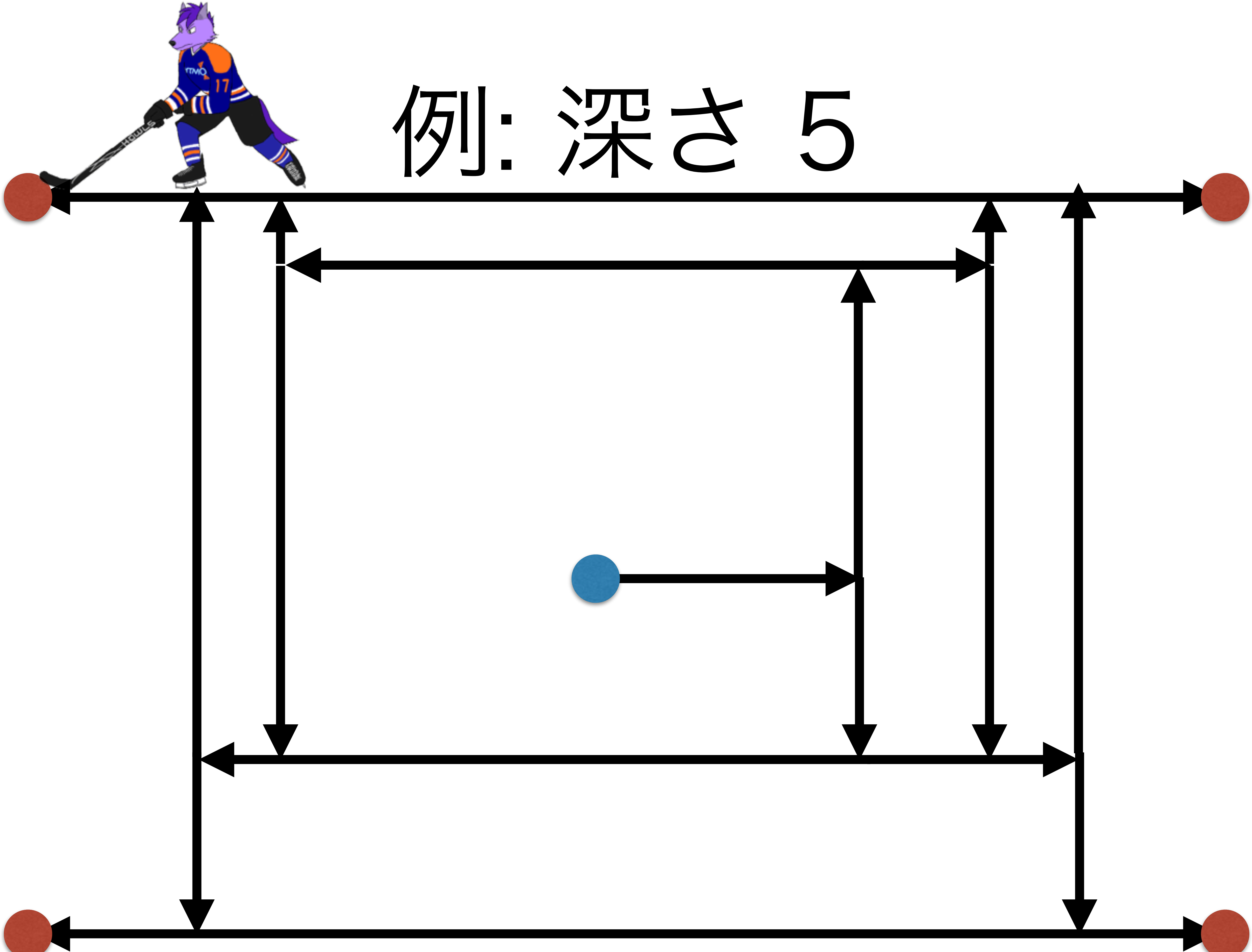
例: 深さ 3



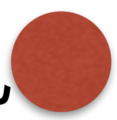
例: 深さ 4



例: 深さ 5



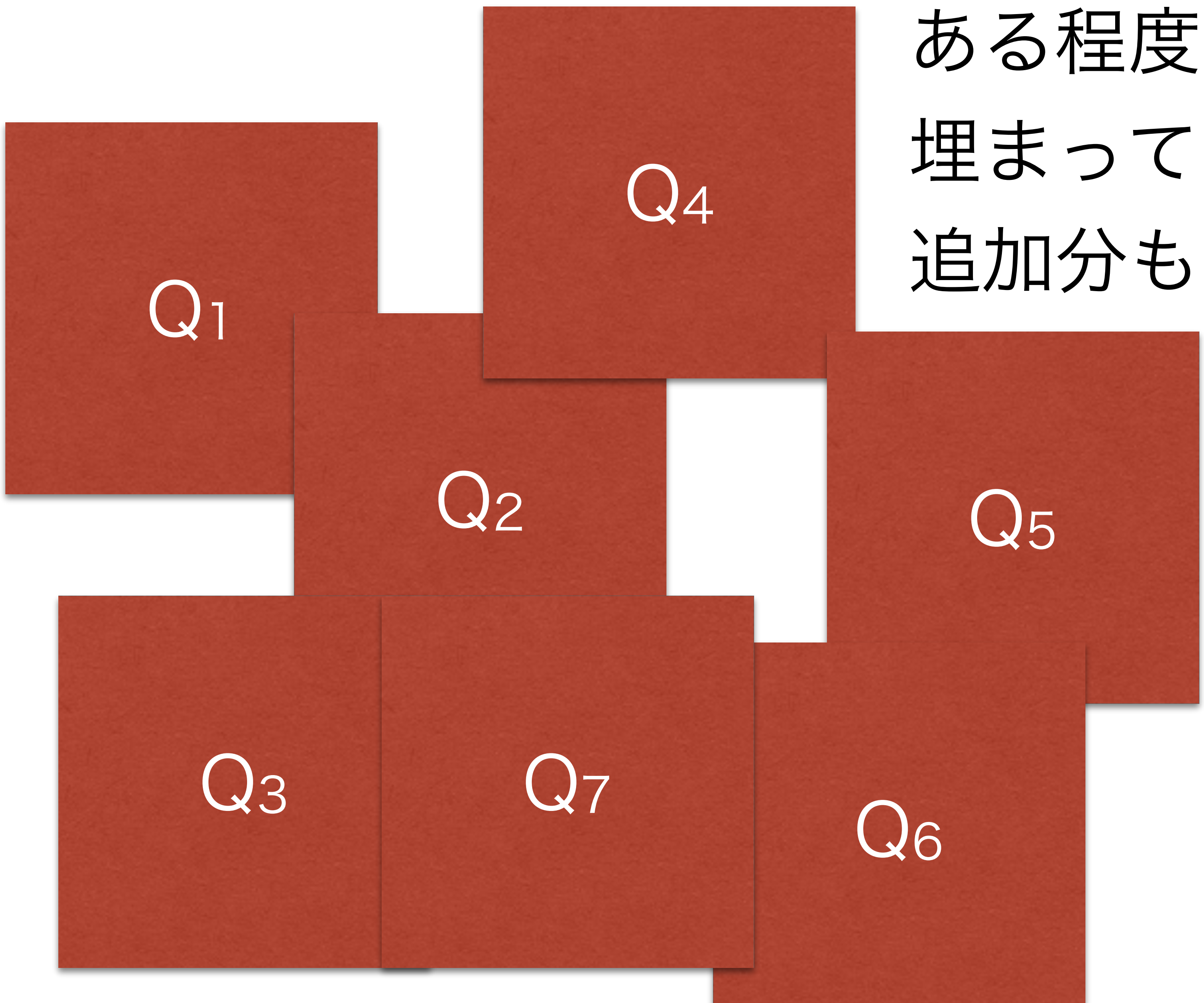
手で試して見ると？

- 一見は増えたりするように見えるが、結局 DP の中に埋もれて回収される
- 全体的に見ると、ある点から移動し始めたときの、深さ O(N) まで進めても状態数は O(N)

単純計算

- クエリ(始点)は全部で Q 個
- 深さは O(N) (=O(H+W)) くらい
- 全部で状態数は O(NQ) ？

さらに拡大すると？



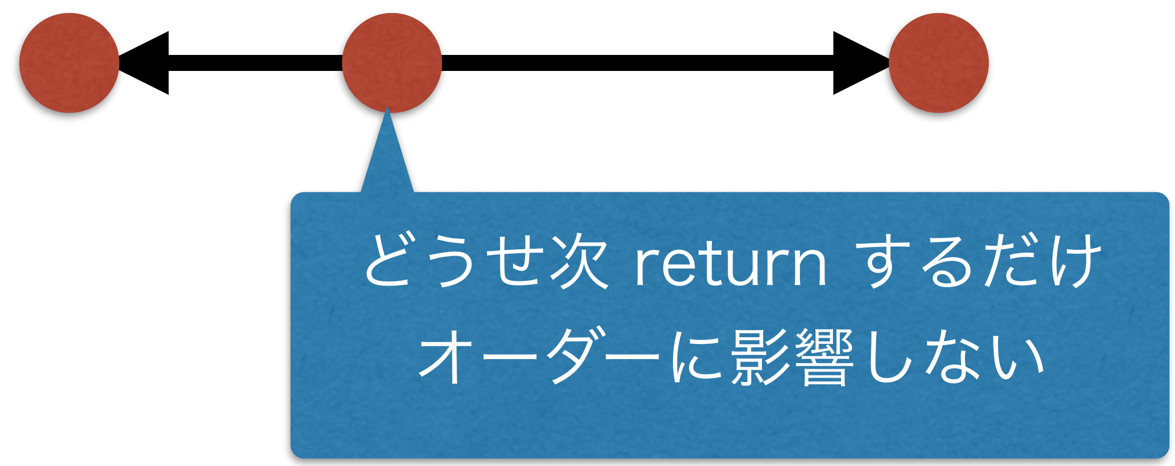
ある程度DPテーブルが
埋まっているので
追加分も少なくなる

後半の解析方法を変えよう

- 途中からは見ていない面積がどう減っていくかで解析しよう
- 深さ N/√Q まで進めた後は、1ステップで面積 N/√Q 以上減る (1 x N/√Q の長方形領域)
- 残った領域が分断されすぎていないことにも注意

後半の状態数

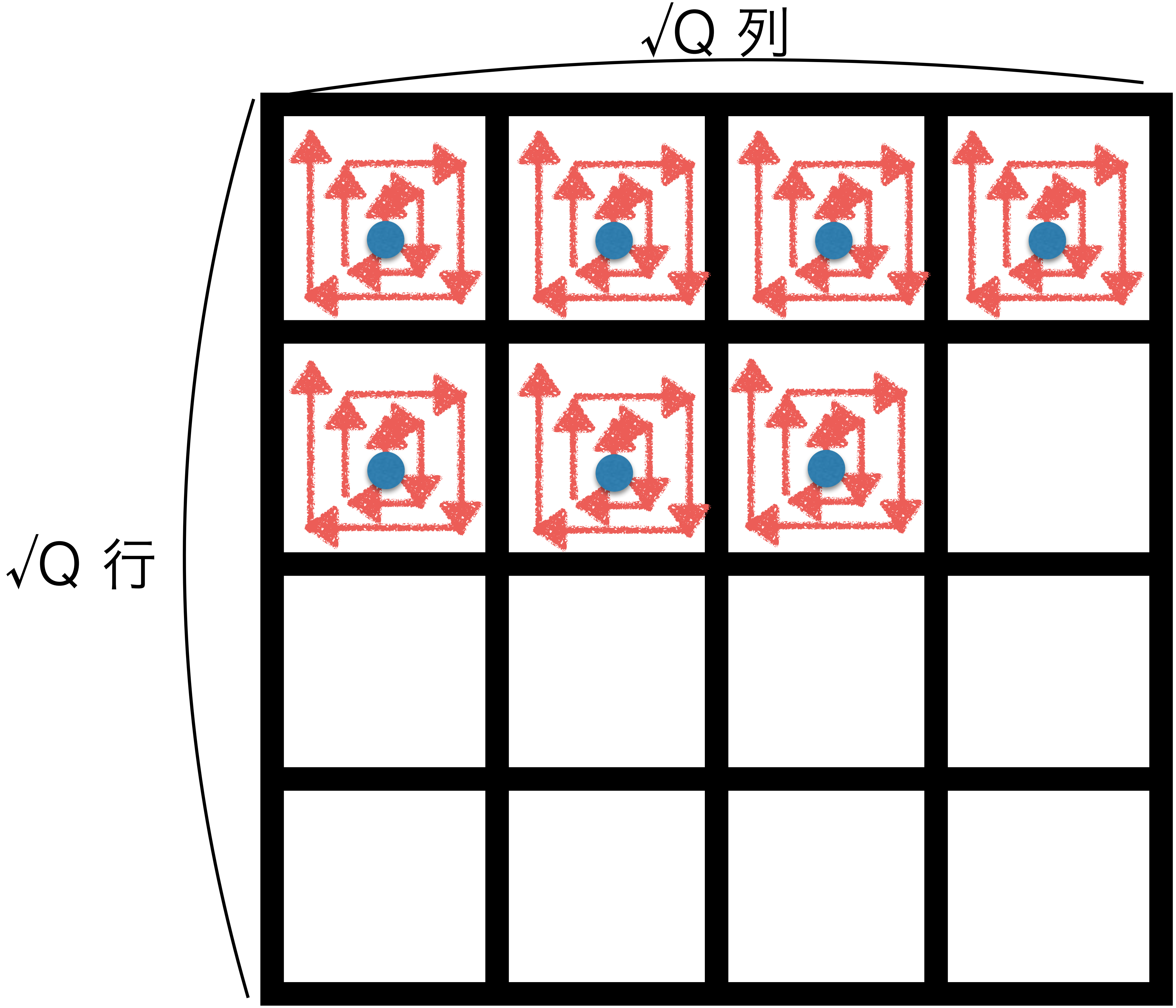
- 面積は全部で N^2
- 1ステップで $N/\sqrt{Q}$ 減る
- 必要なステップ数は $O(N\sqrt{Q})$
- 同じ領域を変なところから何回も見るのでは？
  → どうせ次のステップで埋もれる



計算量まとめ

- 状態数
  - 前半: 各 N/√Q ステップ × Q クエリ
    → O(N√Q)
  - 後半: 合計 N^2 / (N/√Q) すなわち O(N√Q)
- それぞれの状態
  - メモ化(mapとか), Sparse Table: O(log N)

状態数 O(N√Q) かかるケース



余談

- Sparse Tableの代わりにSegtreeを使う
  - 工夫してO(log N) → 大丈夫そう
  - 工夫せずO($\log^2$ N) → 怪しい

- Sparse Tableの代わりにループを回す
  - 曲がる場所に到達したらループを抜ける
  - こうするとかなり軽い (**合計** O(N^2) 激軽)

    → 実はこれは大丈夫 (?)

得点分布