

第3回のキーワード

1

アルゴリズム関係

- 2分探索 (binary search)
- 再帰
- $O(\log n)$

Java関係

- compareTo

本講義の学修では、演習課題の説明内容もよく読んで理解してください。
この資料は図解が中心なので、細かい部分は説明不足になっています。

もっと速い探索方法はないの？

2

- 前回紹介「線形探索」
 - ▣ 先頭から順番に調べていくしかないのか？
 - ▣ コンピュータなら、1000人の名前がバラバラに並んだ名簿からでも一瞬で検索できるけど...

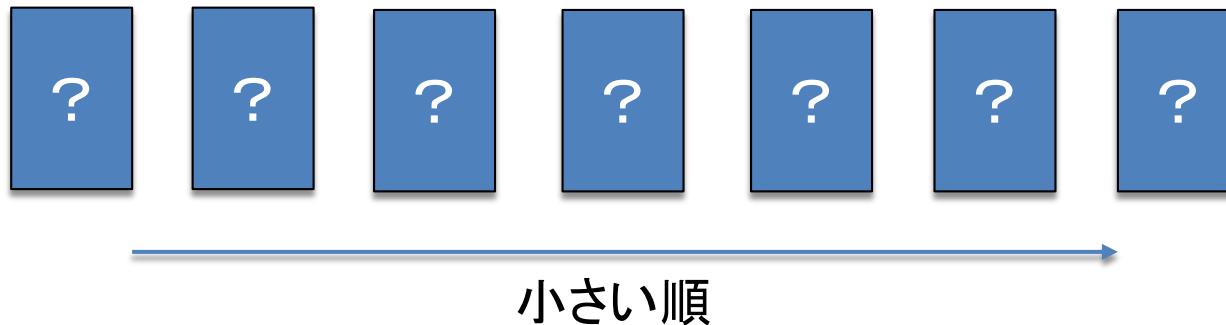
- データ構造を工夫する
 - ▣ もっと効率的に探し出すためには、どうすればいいか？
 - ▣ 世の中の情報は、どのように整理してあるだろうか？

- 探索の高速化戦略
 - ▣ 整列しておく ⇒ 今回説明
 - ▣ 分類しておく ⇒ 似たようなことは次回以降説明

2分探索の考え方

3

- トランプにたとえると...
 - 裏返しのカードが、数の小さい順に並んでいる
 - この中で探したいカードはどこにあるか？

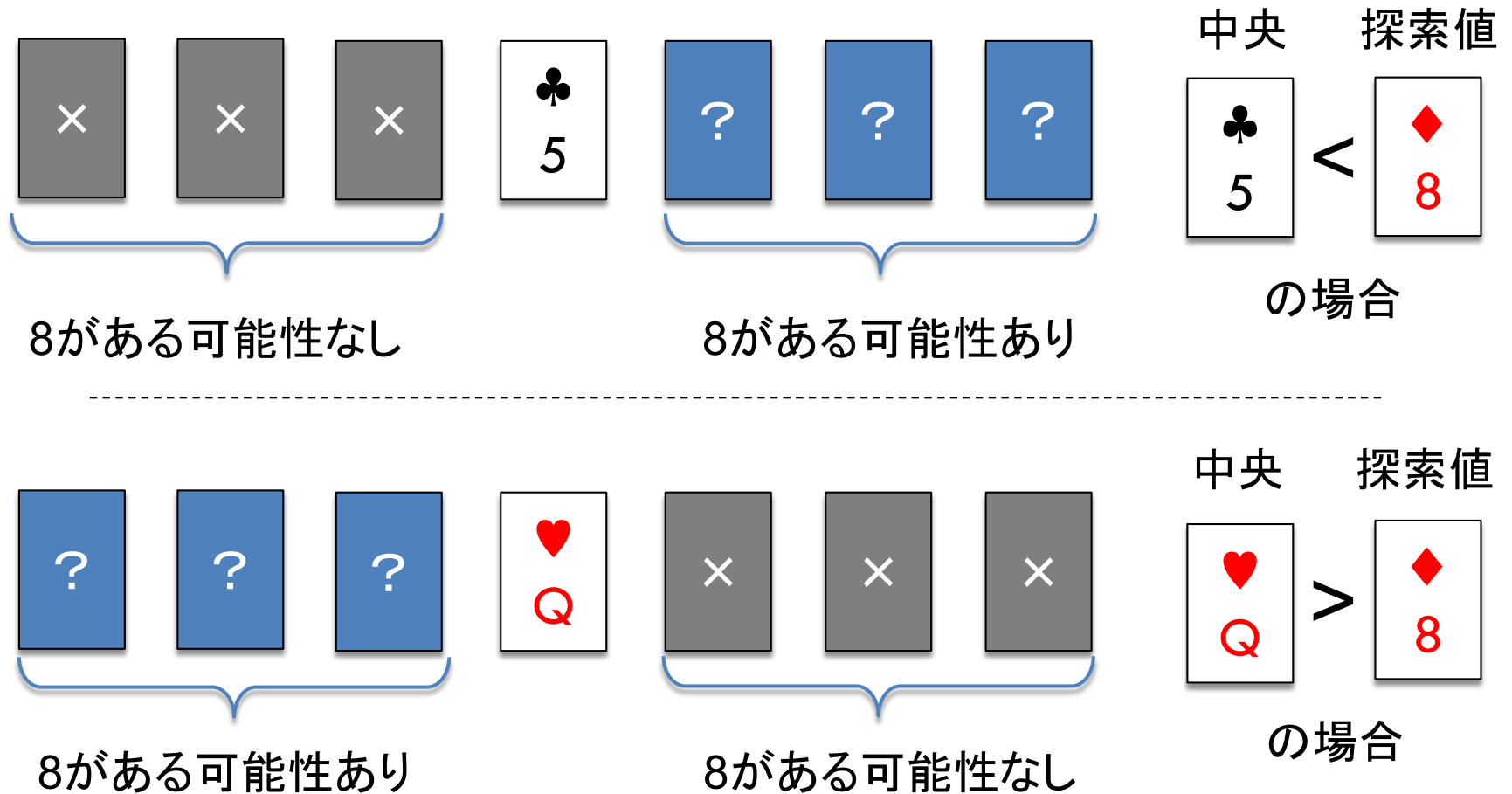


- 戦略を考えてみよう
 - まず、真ん中のカードを開けると何がわかるか？
 - 候補が半分ずつに減っていく(半分→4分の1→8分の1...)

2分探索の考え方

4

- 中央を開けると探索範囲が半分に絞り込まれる



2分探索のアルゴリズム

5

□ データ

- 配列の要素は、小さい順(または大きい順)に並べておく
- 探索する値をkeyとする

これが高速化の
ポイント！

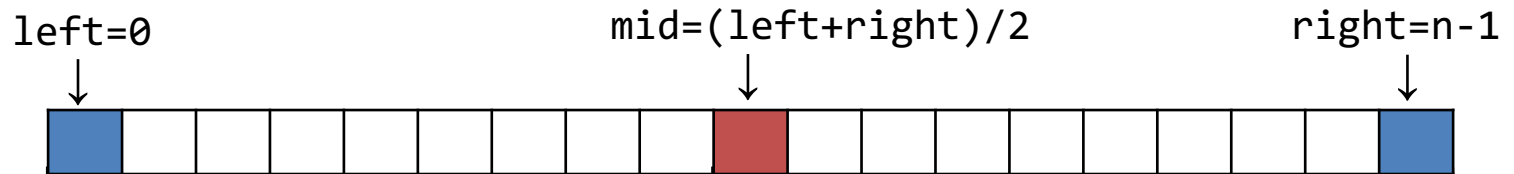
□ アルゴリズム

- 配列(探索範囲)の中央にある値とkeyを比較する
- もし両者が等しければ、発見したのでその位置を返す
- もしkeyの方が小さければ、探索範囲を前半分にせばめる
- もしkeyの方が大きければ、探索範囲を後半分にせばめる
- 以上の手順を、探索範囲に要素がなくなるまで繰り返す
- 探索範囲に要素がなくなったら、keyは含まれていない

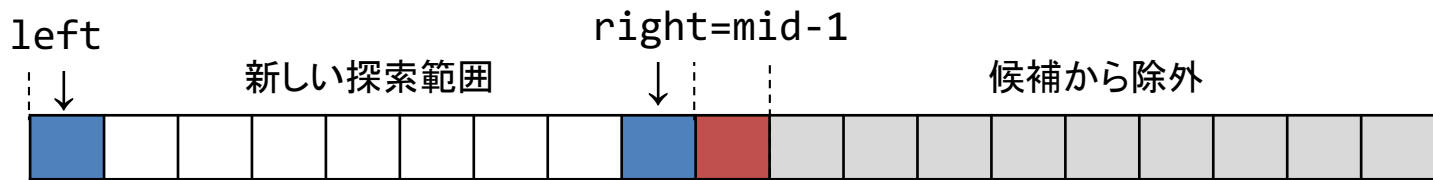
探索範囲のせばめ方

6

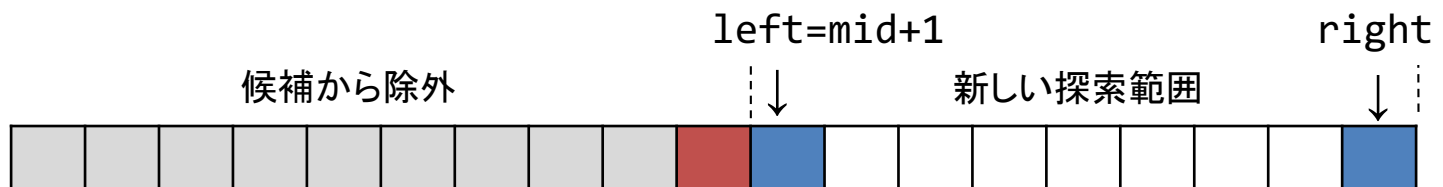
探索範囲の左端と右端を表す変数を使う



左半分にせばめるときは、右端を動かす



右半分にせばめるときは、左端を動かす



2分探索の平均計算量

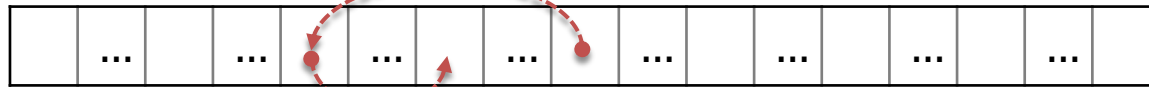
8

簡単のため
 $n = 2^k - 1$ とする

中点→“四分”点→
“八分”点と比較

data[0] data[mid] data[n-1]

配列



そこにkeyが
ある確率

$\frac{1}{n}$ $\frac{1}{n}$ $\frac{1}{n}$ $\frac{1}{n}$... $\frac{1}{n}$... $\frac{1}{n}$...
× × × × ... × ... × ...

比較回数の期待値

1回目で発見 1個

1

$$= \frac{1}{n} \times 1 \times 1$$

2回目で発見 2個

2

2

$$= \frac{1}{n} \times 2 \times 2$$

3回目で発見 4個

3

3

3

3

$$= \frac{1}{n} \times 3 \times 4$$

...

k回目で発見

2^{k-1} 個

k k k k k k k k ...

$$= \frac{1}{n} \times k \times 2^{k-1} \left(+ \right.$$

さらに $n = 2^k - 1$ を
使って n の式にする

$$\frac{1}{n} (1 \cdot 1 + 2 \cdot 2 + 3 \cdot 4 + \dots + k \cdot 2^{k-1})$$

数列の和の復習

9

□ 等差数列の和

□ 考え方：順番をひっくり返したものを足す

$$\begin{array}{r}
 S = 1 + 2 + \dots + (n-1) + n \\
 +) S = n + (n-1) + \dots + 2 + 1 \\
 \hline
 2S = (n+1) \times n
 \end{array}
 \quad \therefore S = \frac{n(n+1)}{2}$$

□ 等比数列の和

□ 考え方：公比をかけたものと差をとる

$$\begin{array}{r}
 S = 2 + 4 + 8 + \dots + 2^{n-1} + 2^n \\
 -) 2S = \quad 4 + 8 + 16 + \dots + 2^n + 2^{n+1} \\
 \hline
 (1-2)S = 2 - 2^{n+1}
 \end{array}
 \quad \therefore S = \frac{2 - 2^{n+1}}{1 - 2}$$

クラス型の比較

10

- Javaにおけるクラス型の比較
 - ▣ 演算子は使えない (Javaは演算子を多重定義できない)
 - ▣ 代わりに, 以下のメソッドを使う

- `x.equals(y)`
 - ▣ `x`と`y`が等しければ `true`, そうでなければ `false` を返す
 - ▣ Javaの全てのクラスがObjectクラスから継承するメソッド

- `x.compareTo(y)`
 - ▣ $x < y$ ならば負, $x = y$ ならば 0, $x > y$ ならば正を返す
 - ▣ StringやIntegerなど, Comparableインタフェースを実装するクラスが持つメソッド