

アルゴリズムとデータ構造⑥

～ 探索問題（ハッシュ） ～

鹿島久嗣

探索問題：

データ集合から所望の要素を見つける

- 探索問題は、データ集合から特定のデータを見つける問題
 - データは「キー」と「内容」からなる
 - 与えられたキーに一致するキーをもったデータを見つけ、その内容を返す
- これは、**ハッシュ**や**二分探索木**等によって実現可能

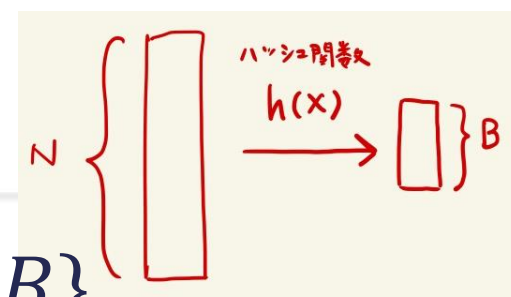
今回の話題

ハッシュ表:

$O(1)$ で探索するためのデータ構造

- データ集合から、あるキーをもつデータを $O(1)$ で発見する
- 単純な実現：
 - キーに対して自然数を割り当てる($1 \sim N$)
 - キーがアルファベット6文字なら $N = 26^6 \simeq 3 \times 10^8$
 - サイズの配列 N を準備する
 - キーを自然数に変換して、配列のその位置に格納する
- 問題点：
 - 長さ N の配列を使うと大きすぎる
 - M 個のデータを格納するのに、 $M \ll N$ なのでムダが多い

ハッシュ関数: ハッシュ表を省スペースで実現する



- ハッシュ関数 $h(x): \{1, 2, \dots, N\} \rightarrow \{1, 2, \dots, B\}$
- キー x を持つデータを $h(x)$ の位置に格納する
 - 異なる x が同じ位置に格納されうるので、衝突したら（例えば）次の場所に格納
- $h(x)$ のデザインは色々考えられるが、なるべく均等に格納されるものがよい
 - 例: $x = a_1 a_2 \dots a_6$ (アルファベット6文字) に対するハッシュ関数 $h(x) = \sum_{i=1,2,\dots,6} c(a_i) \bmod B$: c は文字コード

ハッシュの衝突：

内部ハッシュと外部ハッシュによって衝突を回避

- 異なるキー x と y に対して $h(x) = h(y)$ となることがある
 - $x = \text{すし}, y = \text{しす}$
- 衝突の回避法：内部ハッシュと外部ハッシュ
 - 外部ハッシュ：衝突回避のためにリストを格納
 - 内部ハッシュ：衝突時に別のハッシュ関数を用いる

外部ハッシュ： 衝突回避のためにリストを格納する

- 配列にデータを直接格納せず、 $h(x)$ の位置にリスト（へのポインタ）を格納
 - $M > B$ でもよい
- 計算量：ハッシュが一樣ならば平均的に $O(1)$
 - ハッシュ関数を用いて配列にアクセスするところまで $O(1)$
 - そこから先の計算量はリストの長さ ℓ に依存
 - ハッシュが一樣ならば $\ell \approx \frac{M}{B}$,
これを定数とみなせば平均的に $O(1)$

内部ハッシュ：

衝突したときのために複数のハッシュ関数を用意

- 配列にデータを直接格納する
 - $M \leq B$ である必要がある
- ハッシュ関数の列 h_0, h_1, h_2, \dots を用意して、 h_i が衝突したら、次の h_{i+1} を調べる
 - 例： $h_i(x) = h(x) + i \bmod B$

内部ハッシュの計算手間： 挿入は平均的に $O(1)$

- 新しい要素を挿入する手間： 空きがみつかるまでの期待ステップ数は、現在 M 個のデータがランダムに入っていて、かつハッシュもランダムだとすると、 $\frac{B}{B-M} = \frac{1}{1-\alpha}$

- $\alpha = \frac{M}{B}$ (占有率)

- ハッシュ表を最初からつくる (M 個の要素を登録する) と

$$-\sum_{m=0}^{M-1} \frac{B}{B-m} \approx \int_0^{M-1} \frac{B}{B-m} dm = B \log \frac{B}{B-M+1}$$

$$\text{—要素あたり平均 } \frac{B}{M} \log \frac{B}{B-M+1} \approx -\frac{1}{\alpha} \log(1 - \alpha)$$

内部ハッシュの計算手間： 探索と削除

■ 探索・削除の手間

— x が表にないならば挿入と同じ

— x が表にあるならば作成の一要素あたり手間と同じ

- x を含む M 個の要素をもつハッシュ表をつくることを考える

- x が m 番目に挿入された場合、それは $\frac{B}{B-m}$ ステップで見つかる位置に置かれる

■ 占有率 $\alpha = \frac{1}{2}$ にしておけば $-\frac{1}{\alpha} \log(1 - \alpha) \approx 1.39$