KYOTO UNIVERSITY

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

DEPARTMENT OF INTELLIGENCE SCIENCE AND TECHNOLOGY

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

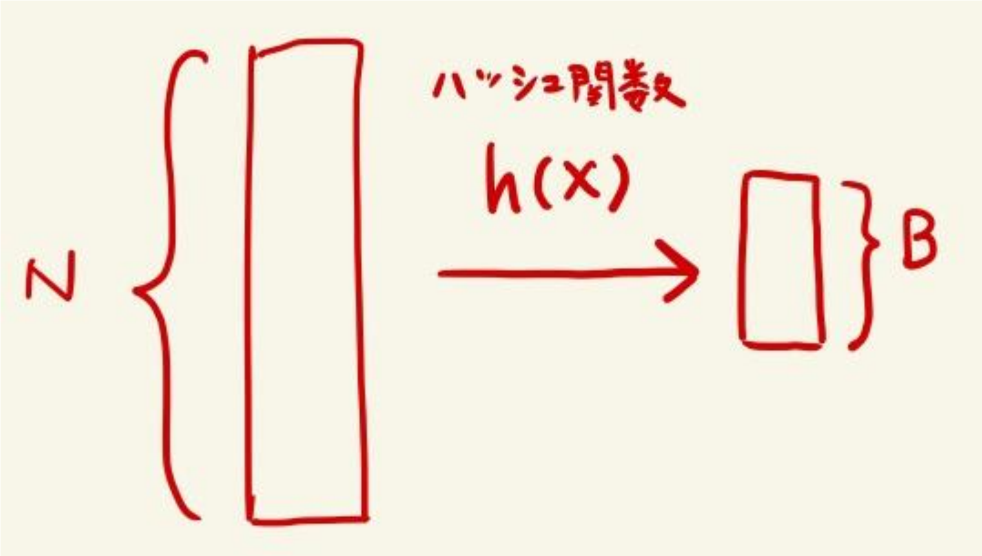
今回の話題

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 $h(x)$: $\{1,2,\ldots,N\} \to \{0,1,\ldots,B-1\}$
- キー x を持つデータを $h(x) \in \{0,1,\ldots,B-1\}$ の位置に格納する
- N種類の値を、B個の格納箇所に詰め込む



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 $h(x)$: $\{1,2,\dots,N\} \rightarrow \{0,1,\dots,B-1\}$
  - キー x を持つデータを$h(x)$の位置に格納する
- $h(x)$の設計は任意だがなるべく均等に格納されるのがよい
  - 例：$x = a_1 a_2 \dots a_6$ （アルファベット6文字）
    - $h(x) = \Sigma_{i=1,2,\dots,6}\, c(a_i) \bmod B$: cは文字コード
- ハッシュの衝突：キー$x \neq y$に対して$h(x) = h(y)$となりうる
  - 例：$x =$ すし, $y =$ しす $\rightarrow h($すし$) = h($しす$) = 5$
  - ハッシュ値が衝突した場合の解消方法は後程

ハッシュの衝突解決：
衝突が起きたとき内部ハッシュと外部ハッシュで解決する

- ハッシュの衝突：
  異なるキーxとyに対して$h(x) = h(y)$となることがある
  – 衝突しない（単射）ハッシュは完全ハッシュと呼ばれる
- 衝突の解決法：内部ハッシュと外部ハッシュ
  – 外部ハッシュ（チェイン法）：
    衝突してもよいようにデータはリストに格納
  – 内部ハッシュ（オープンアドレス法）：
    衝突時に別のハッシュ関数を用いて格納場所を再計算

外部ハッシュ：
衝突してもよいようにデータをリストに格納する

- 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：ハッシュが一様ならば平均的に $O(1)$
  - 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
    - ハッシュが一様ならば$\ell \approx \frac{M}{B}$,
      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$h_0, h_1, h_2, \ldots$を用意して、h_iが衝突したら次のh_{i+1}を調べる、…
  - 例：
    - 線形探索：$h_i(x) = h(x) + i \bmod B$
      - キャッシュ値が固まりやすい
    - 二次探索：$h_i(x) = h(x) + i^2 \bmod B$

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
  - $\sum_{m=0}^{M-1}\frac{B}{B-m}\approx\int_0^{M-1}\frac{B}{B-m}\mathrm{d}m=B\log\frac{B}{B-M+1}$
  - 一要素あたり平均 $\frac{B}{M}\log\frac{B}{B-M+1}\approx-\frac{1}{\alpha}\log(1-\alpha)$

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の作成を考えると、xがm番目に挿入された場合、$\frac{B}{B-m}$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 $-\frac{1}{\alpha}\log(1-\alpha) \approx 1.39$
- 削除は面倒：削除する要素の位置に「墓標」を置いて探索経路が絶たれないようにする。挿入時には再利用可能。