アルゴリズムとデータ構造

第7回探索のためのデータ構造(3)
その1:ハッシュ表

ハッシュ表

- 今日の内容:
  - 抽象データ型の「辞書」
  - 探索(search)と, 挿入(Insert), (delete)をO(1)平均時間で実現するデータ構造.
  - 実装が簡単で, 実際の性能も良いため, 広く使われる
- ポイント
  - ハッシュ関数のアイディア
  - 平均計算時間量がO(1)のデータ構造

前回の復習

辞書に適したデータ構造とは？

次の3つの基本操作を伴う集合Sを**辞書(dictionary)**という。

1. member(S, x) : x∈Sならばyes, x∉Sならばnoを出力
2. Insert(S, x) : SをS∪{x}に更新
3. delete(S, x) : SをS －{x}に更新

要するに検索と挿入と削除が高速に行えるデータ構造

1．整列された配列
member　　最悪時間計算量 O(log n)
insert, deleate　最悪時間計算量 O(n) ← これは問題！

2．2分探索木
member,insert,delete　最悪時間計算量　O(n)
平均時間計算量　O(log n) ← 挿入の順番がランダムと仮定

3．平衡2分探索木
AVL木（全ての節点において、左部分木と右部分木の高さの差が1以内の2分探索木）

member,insert,delete　最悪時間計算量O(log n)

本日の内容

Q.　検索・挿入・削除の時間計算量は、O(log n)の壁を破れないか？

平均時間計算量なら可能！

- ハッシュ表を用いれば、平均時間計算量をO(1) （定数時間）で行える。
- ただし、ハッシュ表のサイズmを格納要素数nのオーダーO(n)にとった場合に, O(1)を実現可能

ハッシュ表は最も実用的な辞書に適した構造

ハッシング

ハッシュ表(hash table)とは

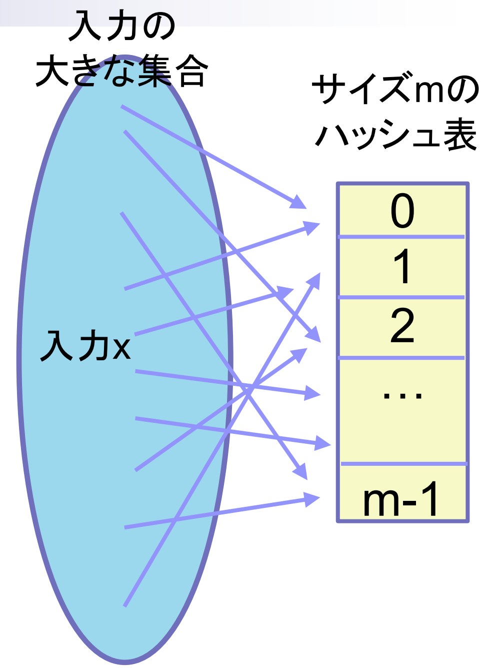
ハッシュ関数hを使って、要素xがあらかじめ準備したm個の場所のうち、位置h(x)に格納される表

hash は英語で「ごた混ぜにする」という意味。

Texas hash with cornbread and green beans
(Wikipedia, Hash (food), United States

ハッシング

ハッシュ表(hash table)とは
ハッシュ関数hを使って、要素xがあらかじめ準備したm個の場所のうち、位置h(x)に格納される表



hash は英語で「ごた混ぜにする」という意味。

[ハッシュ関数 h のもつべき性質]

1. 関数値h(x)の高速な算出が可能である
2. 要素となりうるxに対して、ハッシュ値h(x)がm個の位置にできるだけ偏りなく分布すること

[よく用いられるハッシュ関数]

・xが整数の場合

h(x)=x%m (xをmで割った余り)

・xが文字列の場合

$$h(x)=\left(\sum_{i=0}^{k-1} \mathrm{ord}(x[i])\right)\%m$$

ただし、ord(a)は文字aの整数コード(ASCII, JIS, EUC等)であり、kはxの文字列長

一般にデータはいろいろな周期のものを含んでいるため、mがデータの周期を約数にもつと関数値の衝突が起こりやすくなる。そのためmとしては素数を選ぶことが多い。

しかし、どのようなハッシュ関数を選んでも衝突は避けられない！

衝突対処法

異なる要素x,yに対し、ハッシュ値が等しくなる(h(x)=h(y))ことがある。
そのような場合の対処法として次の2つがある。

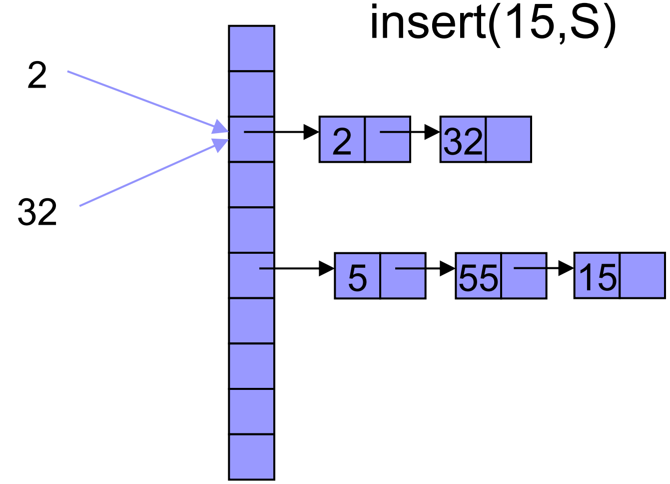
h(x)=x%mの場合の

insert(15,S)

1. **外部ハッシュ法、チェイニング**
   (open hashing, chaining)

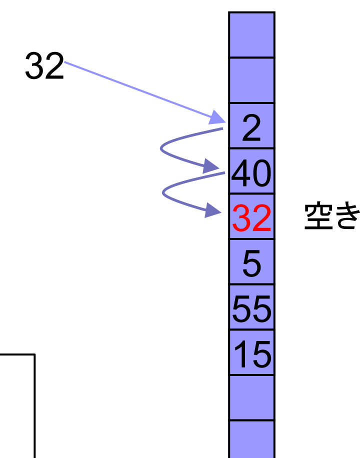
   同じハッシュ値をもつ要素を、
   その値に対応するバケットに格納する。
   バケットは連結リストで実現できる。

32



2. **内部ハッシュ法、オープンアドレッシング**
   (closed hasing, open addressing)

   xを格納しようとしたときに、位置h(x)がすでに
   使われている場合、新しいハッシュ値$h_i(x)$ (i=1,2,…)を
   次々と求め、最初に見つかった空いている位置
   $h_i(x)$に格納する。h_iとしては、以下の関数などが使われる。



$$h_i(x)=(h(x)+i)\%m$$

コメント： この関数は，位置h(x)が使われていたら，添字が一つ大きな隣を順に見ていき，最初の空き位置に入れることと同じ.

外部ハッシュ法の基本操作

member(x,S) h(x)に対応するバケット内で値がxのものを探し、見つかればyes、見つからなければnoを返す。

Insert(x,S) h(x)に対応するバケット内で値がxのものを探し、見つかれば何もしない、見つからなければバケットに追加。

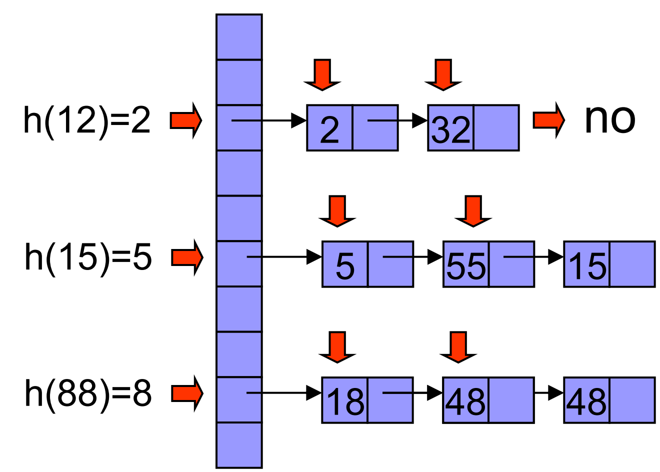
delete(x,S) h(x)に対応するバケット内で値がxのものを探し、見つかれば削除、見つからなければ何もしない。

基本操作の時間計算量

α=n/m : 占有率 (n: 格納要素数, m: バケット数)

最悪時間計算量 O(n)
(全ての要素が同じハッシュ値をもつ場合)

平均時間計算量 O(α)
(n=O(m)であればO(1))



内部ハッシュ法の基本操作

member(x,S) 位置h(x)から探し始め、h(x),h_1(x),h_2(x),…の順でその位置の要素がxと等しいかをチェックする。等しいものがみつかったらyesを返す。位置h_i(x)が空き(“empty”)となるまでチェックし、見つからなければnoを返す。

Insert(x,S) 位置h(x)から探し始め、h(x),h_1(x),h_2(x),…の順でその位置の要素がxと等しいかをチェックする。位置h_i(x)が空き(“empty”)となるまでチェックし、xと等しい要素が見つかれば何もしない。みつからなければ、検索途中で見つけた空き(“empty” or “deleted”)にxを格納する。

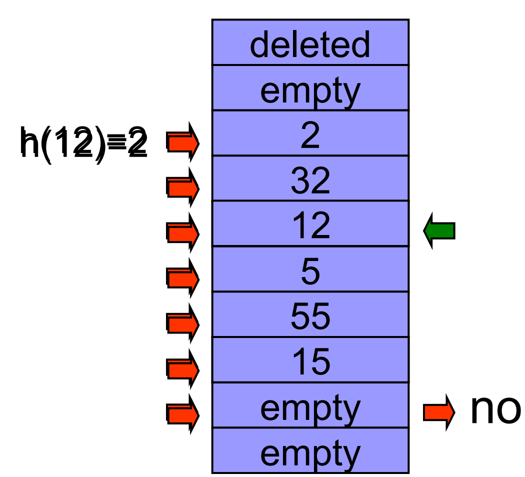
delete(x,S) 位置h(x)から探し始め、h(x),h_1(x),h_2(x),…の順でその位置の要素がxと等しいかをチェックする。位置h_i(x)が空き(“empty”)となるまでチェックし、xと等しい要素が見つかれば削除し”deleted”のフラグを立てる。みつからなければ何もしない。

基本操作の時間計算量

α=n/m : 占有率　(n: 格納要素数, m: バケット数)

最悪時間計算量　O(n)
(全ての要素が同じハッシュ値をもつ場合)

平均時間計算量　O(1/(1-α))　(xが表にない場合)
O(-(1/α)log(1-α))　(xが表にある場合)
どちらの場合もn=O(m)であれば、O(1)



内部ハッシュ法の平均時間計算量の証明

・xが表にない場合

$h_0(x)(=h(x)),h_1(x),\ldots$の順にハッシュ値の計算を行うものとする。
位置$h_i(x)$で初めて空を見つける確率は

$$\frac{n(n-1)\cdots(n-i+1)(m-n)}{m(m-1)\cdots(m-i+1)(m-i)}$$

である。(ただし、i=0のとき、上式は(m-n)/mをあらわすものとする。)
したがって、要素の比較回数の期待値は、

$$\begin{aligned}
\sum_{i=0}^{n}(i+1)\frac{n(n-1)\cdots(n-i+1)(m-n)}{m(m-1)\cdots(m-i+1)(m-i)} &= \sum_{i=0}^{n}(i+1)\frac{n(n-1)\cdots(n-i+1)}{m(m-1)\cdots(m-i+1)}\left(1-\frac{n-i}{m-i}\right) \\
&= \sum_{i=0}^{n}(i+1)\frac{n(n-1)\cdots(n-i+1)}{m(m-1)\cdots(m-i+1)} - \sum_{i=1}^{n+1} i\frac{n(n-1)\cdots(n-i+1)}{m(m-1)\cdots(m-i+1)} \\
&\le \sum_{i=0}^{n}\frac{n(n-1)\cdots(n-i+1)}{m(m-1)\cdots(m-i+1)} \\
&\le \sum_{i=0}^{\infty}\left(\frac{n}{m}\right)^{i} = \frac{1}{1-\dfrac{n}{m}} = \frac{1}{1-\alpha}
\end{aligned}$$

比較回数の定数倍の時間で計算できるので平均時間計算量はO(1/(1-α))

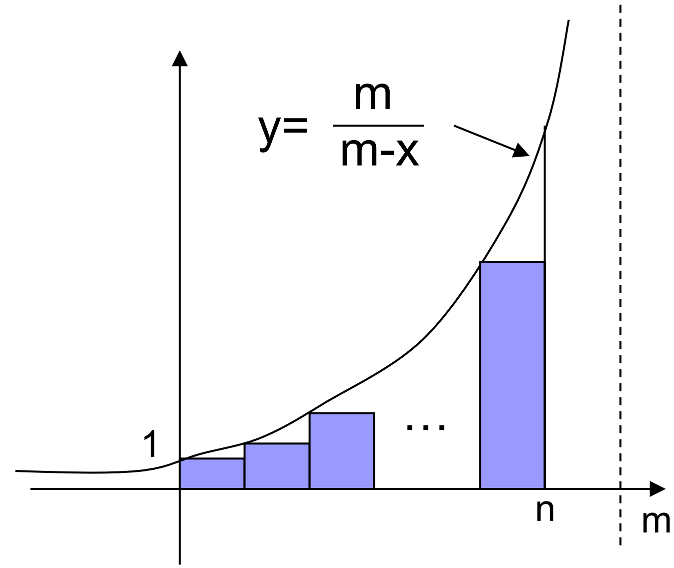
内部ハッシュ法の平均時間計算量の証明(続き)

・xが表にある場合

比較回数の期待値は、n個の異なる要素を空の表に格納するのに必要な比較回数の平均に等しい。したがって

$$\frac{1}{n}\sum_{i=0}^{n-1}\frac{m}{m-i} \le \frac{1}{n}\int_0^n \frac{m}{m-x}dx$$

$$= \frac{m}{n}\ln\frac{m}{m-n} = -\frac{1}{\alpha}\ln(1-\alpha)$$



比較回数の定数倍で計算できるので、平均時間計算量はO($\frac{1}{\alpha}$ log(1-α))

(証明終わり)

確率pで生起する事象がk回目で初めて生起するとした場合、kの期待値は1/pであるという事実を使っている。つまり、確率(m-i)/mで生起する事象では、kの期待値はm/(m-i)である。

ハッシュ表

- 今日の内容:
  - 抽象データ型の「辞書」
  - ハッシュ関数を用い，n個の入力要素を長さmの配列に格納するデータ構造
  - （m=O(n)のとき），探索（search）と，挿入（Insert），（delete）をO(1)平均時間で実現するデータ構造.
  - 最悪時間はO(n).
  - 実装が簡単で，実際の性能も良いため，広く使われる
- ポイント
  - ハッシュ関数のもつべき性質と作り方
  - 平均計算時間量がO(1)のデータ構造