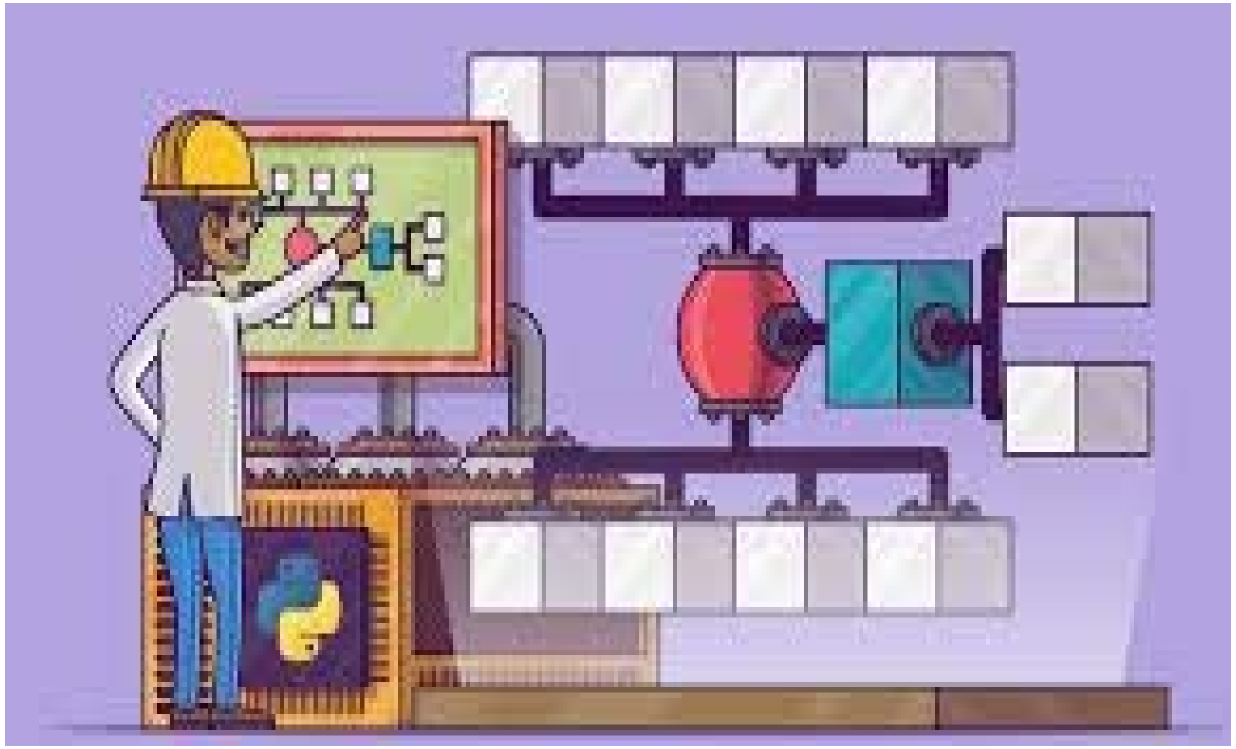
ساختمان داده ها



مدرس:
سمانه حسینی سمنانی

دانشگاه صنعتی اصفهان- دانشکده برق و کامپیوتر

درهم سازی – Hashing

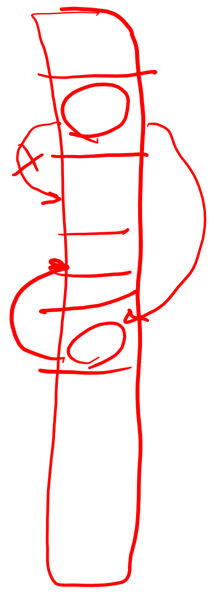
- Direct-address tables
- Hash tables
- Hash functions
- Open addressing

سمانه حسینی سمنانی
هیات علمی دانشکده برق و کامپیوتر- دانشگاه صنعتی اصفهان

Quadratic probing

0, 1, ...

$$h(k,i) = (h'(k) + c_1 i + c_2 i^2) \bmod m$$

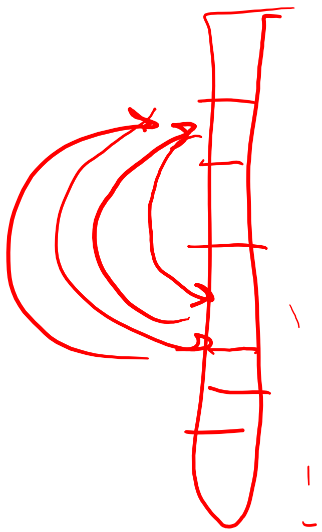


where h' is an auxiliary hash function

c_1 and c_2 are positive auxiliary constants

$$i = 0, 1, \ldots, m-1.$$

Quadratic probing



- This method works much better than linear probing,
- Problems:
- to make full use of the hash table, the values of c1, c2, and m are constrained.
- if two keys have the same initial probe position, then their probe sequences are the same $h(k_1, 0) = h(k_2, 0) \longrightarrow h(k_1, i) = h(k_2, i)$

Double hashing

$h(k,2) = (\cancel{h_1(14)}^{1} + 2 \times \underbrace{\cancel{h_2(14)}}_{4}) \bmod 12 = 9$

$\langle h(k,1), h(k,2), \ldots \rangle$

$\langle 5, 9, \ldots \rangle$

$$h(k,i) = (h_1(k) + i h_2(k)) \bmod m$$

$h_1(k) = k \bmod 13$

$h_2(k) = 1 + (k \bmod 11)$

$k = 14$

$h(14,0) = (1 + 0 \times 4) \bmod 13 = 0$

$h_2(14) = 1 + \underbrace{(14 \bmod 11)}_{3} = 4$

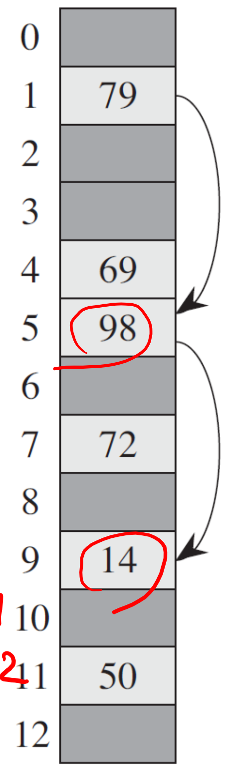
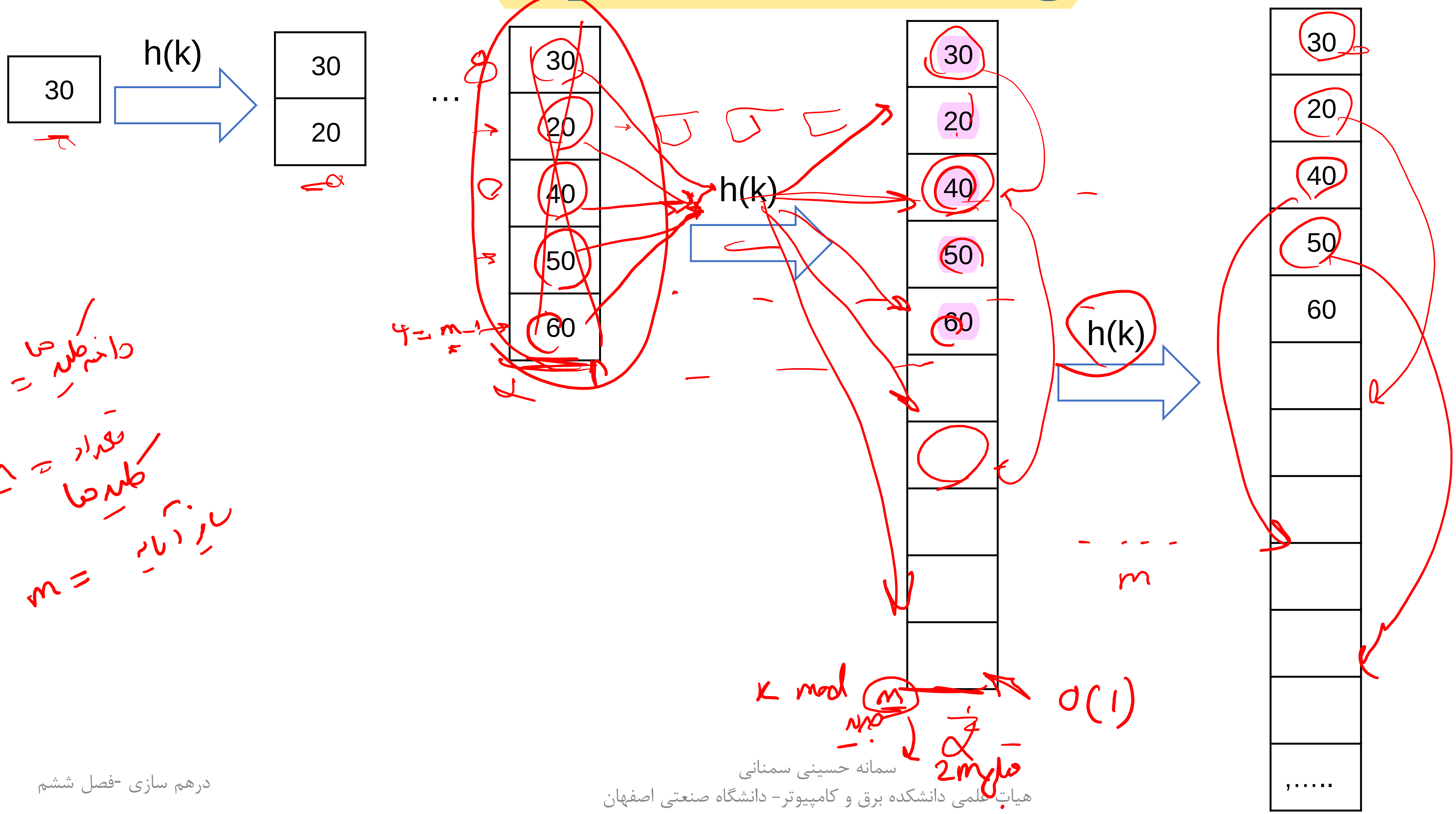


Figure 11.5 Insertion by double hashing. Here we have a hash table of size 13 with $h_1(k) = k \bmod 13$ and $h_2(k) = 1 + (k \bmod 11)$. Since $14 \equiv 1 \pmod{13}$ and $14 \equiv 3 \pmod{11}$, we insert the key 14 into empty slot 9, after examining slots 1 and 5 and finding them to be occupied.

Open addressing



u = دامنه کلیدها

n = تعداد کلیدها

m = سایز آرایه

سمانه حسینی سمنانی
هیات علمی دانشکده برق و کامپیوتر- دانشگاه صنعتی اصفهان

Open addressing

$$n \rightarrow O(n)$$

$$c_i = \begin{cases} 1 & i \neq 2^{k}+1 \\ 2^{k}+1 & i = 2^{k}+1 \end{cases}$$

هزینه زیاد برای دو برابر کردن سایز آرایه

$$2^{1},\ 2^{2},\ 2^{3},\ \ldots\ldots,\ 2^{\log n}$$

هزینه اونهایی که هزینه زیاد لازم دارند

$$\sum_{k=0}^{\log n} 2^{k} = \frac{2^{\cancel{\log n}+1}-1}{2-1} = O(2n)$$

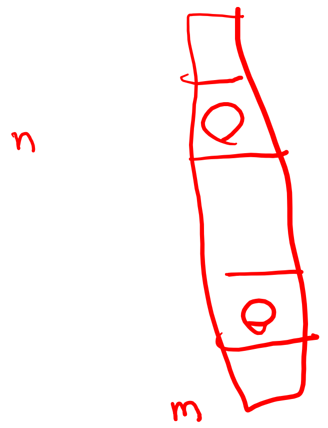
$$O(3) = \frac{O(2n+n)}{n} = \text{هزینه سرشکن}$$

هزینه درج هر عنصر

Analysis of search in open-address hashing

$$\langle h(k,0), h(k,1), \ldots, h(k,m-1)\rangle$$

$$\alpha = n/m \qquad \frac{n}{m} < 1$$



$n \le m$, which implies $\alpha \le 1$.

we assume uniform hashing: the probe sequence of each key is equally likely to be any of the m! permutations of <0, 1, …., m -1>.

⇩

Average #probs in unsuccessful search: $O(\frac{1}{1-\alpha})$

Analysis of search in open-address hashing

Theorem 11.6

Given an open-address hash table with load factor $\alpha = n/m < 1$, the expected number of probes in an unsuccessful search is at most $1/(1-\alpha)$, assuming uniform hashing.

$h(k,0)\,,\ h(k,1)\,,\ \ldots\ h(k,i)$

آدرس خانه خالی NIL

$\boxed{\text{تعداد prob های چک شده} = i+1}$

$X = i+1$

$p(x=i) = P(x \geq i) - P(x \geq i+1)$

Analysis of search in open-address hashing

متغیر تصادفی

هزینه سرچ — X = the number of probes made in an unsuccessful search

$\boxed{E(X)} = \sum i \times P($ادرس خالی را ببیند$)$ ($x=i$) $= \sum i \times P(X \geq i) - P(X \geq i+1)) = \sum P(x \geq i)$

event A_i : the event that an i_{th} probe occurs and it is to an occupied slot

اشغال نبودن خانه

for $i = 1, 2, \ldots,$ probe sequence

$$\text{event } \{X \geq i\} = A_1 \cap A_2 \cap \cdots \cap A_{i-1}$$

$$\Pr\{A_1 \cap A_2 \cap \cdots \cap A_{i-1}\} = \Pr\{A_1\} \cdot \Pr\{A_2 \mid A_1\} \cdot \Pr\{A_3 \mid A_1 \cap A_2\} \cdots \Pr\{A_{i-1} \mid A_1 \cap A_2 \cap \cdots \cap A_{i-2}\} .$$

Analysis of search in open-address hashing

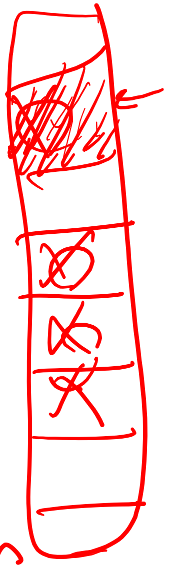
Since there are n elements and m slots, $\Pr\{A_1\} = n/m$.

$$\Pr\{A_1 \cap A_2 \cap \cdots \cap A_{i-1}\} = \Pr\{A_1\} \cdot \Pr\{A_2 \mid A_1\} \cdot \Pr\{A_3 \mid A_1 \cap A_2\} \cdots \Pr\{A_{i-1} \mid A_1 \cap A_2 \cap \cdots \cap A_{i-2}\} .$$

$$\begin{aligned}\Pr\{X \geq i\} &= \frac{n}{m} \cdot \frac{n-1}{m-1} \cdot \frac{n-2}{m-2} \cdots \frac{n-i+2}{m-i+2} \\ &\leq \left(\frac{n}{m}\right)^{i-1} \\ &= \alpha^{i-1} .\end{aligned}$$

$\left(\frac{n}{m}\right)$ $\left(\frac{n}{m}\right)$

$\frac{n}{m}$



m

$\frac{n-1}{m-1}$

$\frac{n-1}{m-1} < \frac{n}{m}$

$(n-1)m < n(m-1)$

$\cancel{nm} - m < \cancel{nm} - n$

$n < m$

Analysis of search in open-address hashing

$$
\begin{aligned}
\mathrm{E}[X] &= \sum_{i=1}^{\infty} \Pr\{X \geq i\} \\
&\leq \sum_{i=1}^{\infty} \alpha^{i-1} \\
&= \sum_{i=0}^{\infty} \alpha^{i} \\
&= \frac{1}{1-\alpha}.
\end{aligned}
$$

$\to \alpha^0 + \alpha^1 + \alpha^2 \cdots$

$$1/(1-\alpha) = 1 + \alpha + \alpha^2 + \alpha^3 + \cdots$$

$\frac{n}{m} \times \frac{n-1}{m-1}$

- We always make the first probe.
- With probability approximately α, the first probe finds an occupied slot, so that we need to probe a second time.
- With probability approximately α^2, the first two slots are occupied so that we make a third probe, and so on.

Analysis of search in open-address hashing

$$\frac{1}{1-\alpha}$$

If α is a constant, Theorem 11.6 predicts that an unsuccessful search runs in $O(1)$ time. For example, if the hash table is half full, the average number of probes in an unsuccessful search is at most $1/(1-.5) = 2$. If it is 90 percent full, the average number of probes is at most $1/(1-.9) = 10$.

$n = m/2$

$\alpha = n/m = 1/2 = 0.5$

$$\frac{1}{1-\underset{0.5}{\alpha}} = O(2)$$

Analysis of search in open-address hashing

Given an open-address hash table with load factor $\alpha < 1$, the expected number of probes in a successful search is at most

$$\frac{1}{\alpha} \ln \frac{1}{1-\alpha} ,$$