Hash Tables: Linear Probing

Uri Zwick
Tel Aviv University

Hashing with open addressing "Uniform probing"

Hash table of size m

Assume that $h : U \times [m] \rightarrow [m]$

Insert key k in the first free position among

$$\underbrace{h(k,0)\ ,\ h(k,1)\ ,\ h(k,2)\ ,\ \ldots\ ,\ h(k,m-1)}$$

(Sometimes) assumed to be a permutation

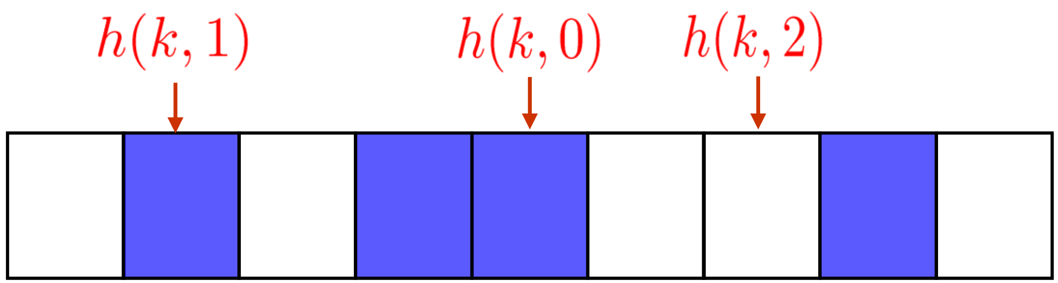


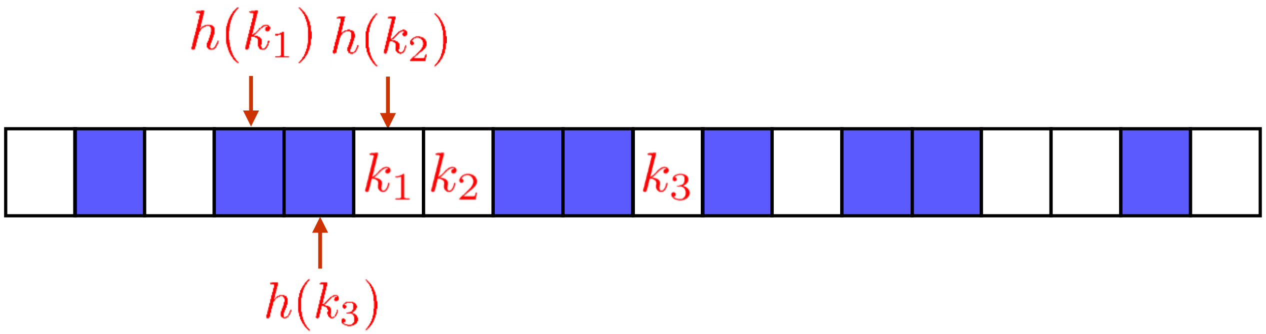
Table is not full → Insertion succeeds

To search, follow the same order

Linear probing

“The most important hashing technique”

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



More *probes* than uniform probing due to *clustering*: long runs tend to get longer and merge with other runs

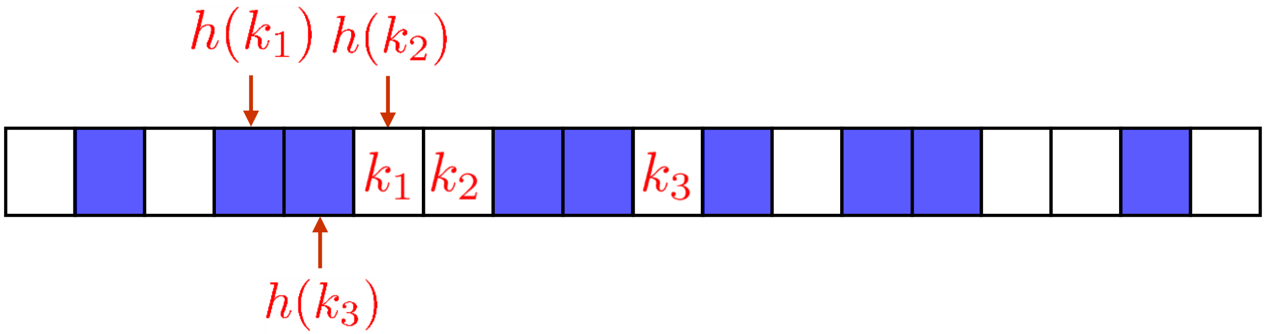
But, many fewer *cache misses*

Extremely efficient in practice

How do we analyze it?

Which hash functions should we use?

Order of insertions



Theorem: The set of occupied cell and the total number of probes done while inserting a set of items into a hash table using linear probing does *not* depend on the *order* in which the items are inserted

Exercise: Prove the theorem

Exercise: Is the same true for uniform probing?

Number of probes

Exercise: Show that if, after inserting n items into a table of size m, the occupied cells in the table form runs of length $\ell_1, \ell_2, \ldots$, where $\sum_i \ell_i = n$, then the expected number of probes in an *unsuccessful* search, assuming the searched key is mapped into a uniformly random location in the table, is

$$1 + \frac{1}{m} \sum_i \frac{\ell_i(\ell_i + 1)}{2}$$

Exercise: What are the smallest and largest possible total number of probes needed to construct a hash table that contain runs of length $\ell_1, \ell_2, \ldots$?

Probabilistic analysis of uniform probing [Petersen (1957)]

n – number of elements in table

m – size of hash table

$\alpha = n/m$ – load factor (Note: $\alpha \leq 1$)

Uniform probing: for every $k \in U$, $h(k,0), \ldots, h(k,m-1)$ is random permutation, independent of all other permutations

Expected no. of probes in an unsuccessful search of a *random* item is at most $\frac{1}{1-\alpha}$

Expected no. of probes in a successful search is at most $\frac{1}{\alpha} \ln \frac{1}{1-\alpha}$

Probabilistic analysis of uniform probing [Petersen (1957)]

Claim: Expected no. of probes in an unsuccessful search is at most: $\frac{1}{1-\alpha}$

The probability that a random cell is occupied is α

The probability that the first i cells probed are all occupied is at most α^i

$$1 + \alpha + \alpha^2 + \ldots = \frac{1}{1-\alpha}$$

Exercise: Do the calculation more carefully and show that the expected no. of probes in an unsuccessful search is exactly $(m+1)/(m-n+1)$

Probabilistic analysis of linear probing

[Knuth (1962)]

$\alpha = n/m$ – load factor ($\alpha \leq 1$)

Random hash function:

for every $k \in U$, $h(k)$ is uniformly distributed, independent of all other $h(k')$, for $k \neq k'$

Expected no. of probes in an unsuccessful search is at most $\frac{1}{2}\left(1+\left(\frac{1}{1-\alpha}\right)^2\right)$

Expected no. of probes in a successful search of a *random* item is at most $\frac{1}{2}\left(1+\frac{1}{1-\alpha}\right)$

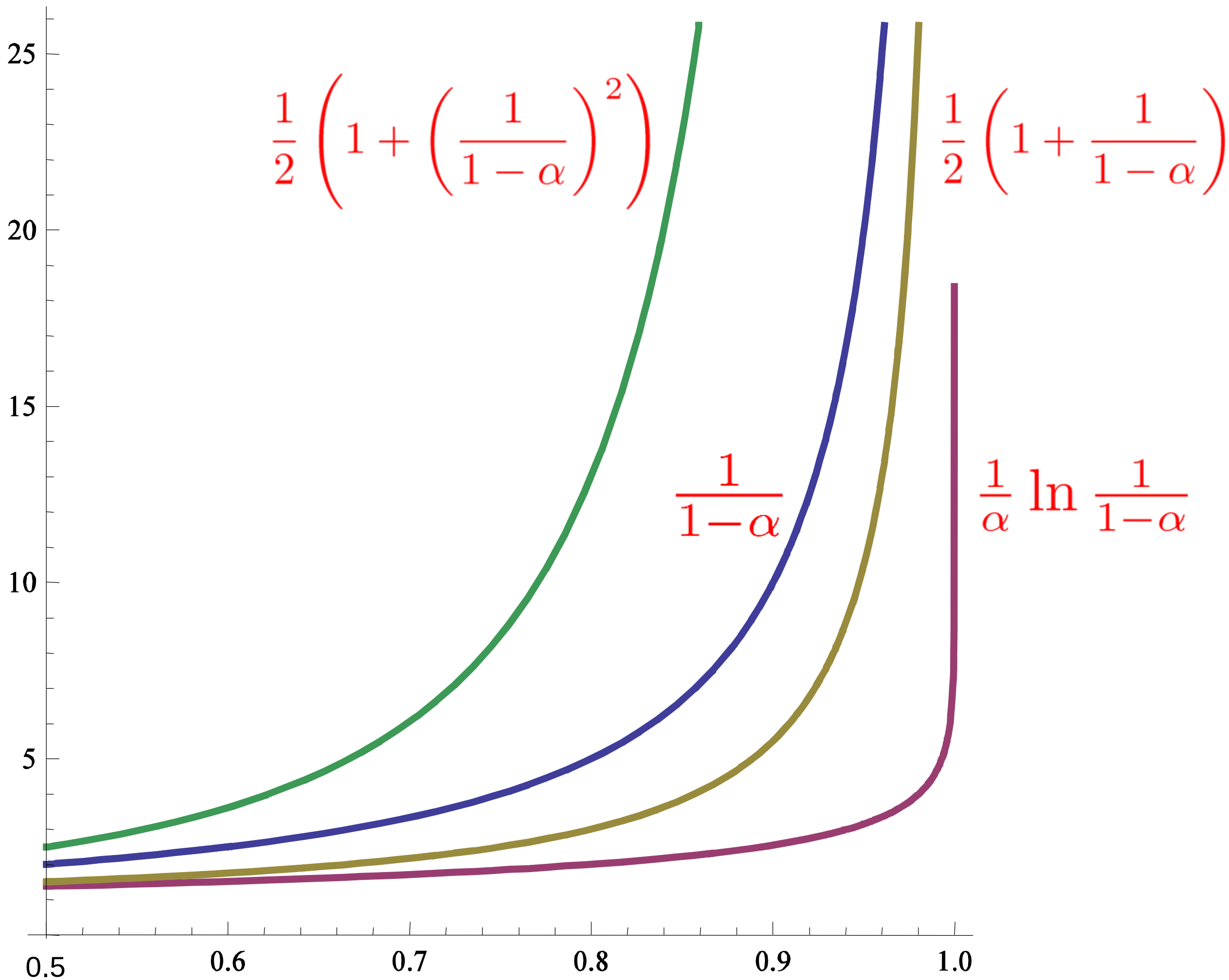
Expected number of probes

Assuming random hash functions

	Unsuccessful Search	Successful Search
Uniform Probing	$\frac{1}{1-\alpha}$	$\frac{1}{\alpha}\ln\frac{1}{1-\alpha}$
Linear Probing	$\frac{1}{2}\left(1+\left(\frac{1}{1-\alpha}\right)^2\right)$	$\frac{1}{2}\left(1+\frac{1}{1-\alpha}\right)$

When, say, $\alpha \leq 0.6$, all small constants

Expected number of probes



Probabilistic analysis of linear probing [Knuth (1962)]

n – number of elements in table

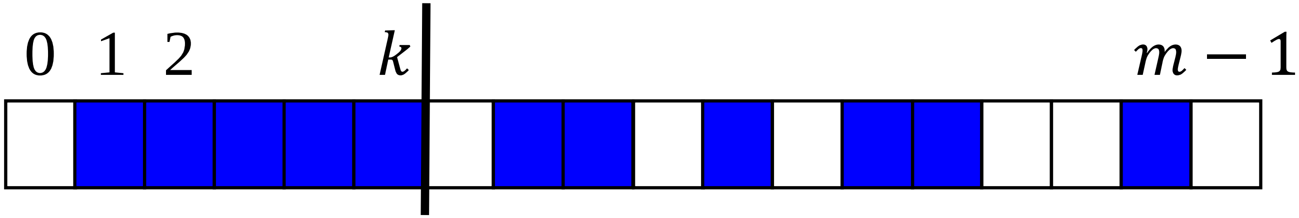
m – size of hash table

What is the probability that $T[0]$ is empty?

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

By symmetry, all cells are equally likely to be empty

What is the probability that $T[0], T[k+1]$ empty, $T[1], \ldots, T[k]$ occupied?



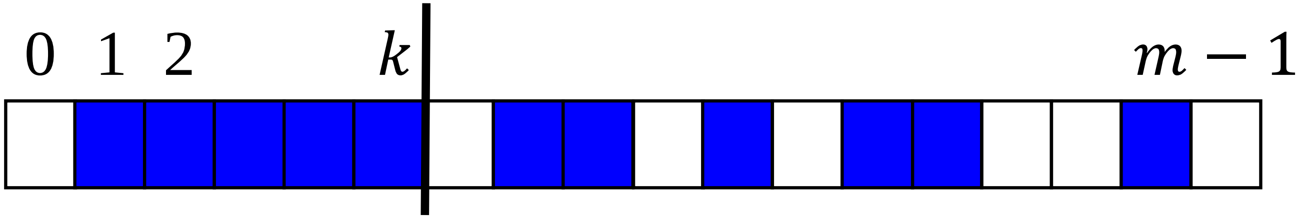
$$\binom{n}{k} \left(\frac{k+1}{m}\right)^k \left(1 - \frac{k}{k+1}\right) \left(\frac{m-k-1}{m}\right)^{n-k} \left(1 - \frac{n-k}{m-k-1}\right)$$

Exactly k items should be mapped to $[0, k]$
and $n - k$ items should be mapped to $[k+1, m-1]$

Given that k items are mapped to $[0, k]$,
$T[0]$ should remain empty

Given that $n - k$ items are mapped to $[k+1, m-1]$,
$T[k+1]$ should remain empty

What is the probability that $T[0], T[k+1]$ empty, $T[1], \dots, T[k]$ occupied?



$$\binom{n}{k}\left(\frac{k+1}{m}\right)^{k}\left(1-\frac{k}{k+1}\right)\left(\frac{m-k-1}{m}\right)^{n-k}\left(1-\frac{n-k}{m-k-1}\right)$$

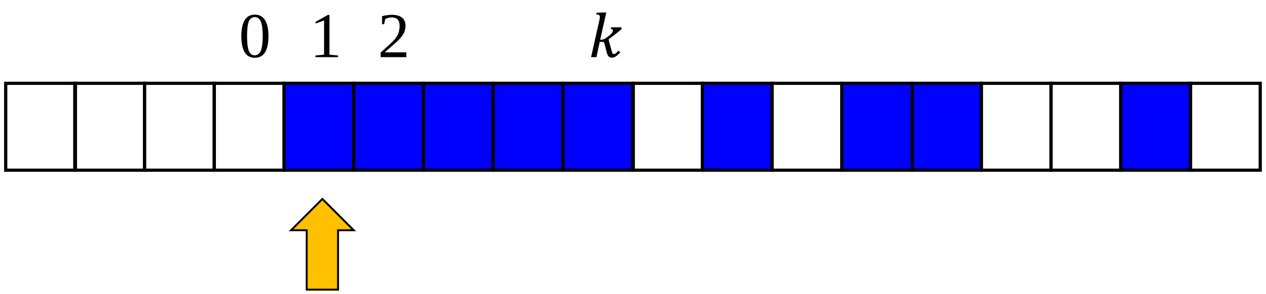
⇩

$$g_k \;=\; m^{-n}\binom{n}{k}\,(k+1)^{k-1}(m-k-1)^{n-k-1}(m-n-1)$$

g_k is the probability that a run of size exactly k starts at a given position

Exercise: $g_0 \to (1-\alpha)\mathrm{e}^{-\alpha}$, $g_1 \to \alpha(1-\alpha)\mathrm{e}^{-2\alpha}$

What is the probability that an unsuccessful search encounters exactly k occupied cells?



$$p_k = \sum_{i=k}^{n} g_i$$

Interesting to note that

$$p_0 = 1 - \alpha$$

$$p_1 = p_0 - g_0 \to (1 - \alpha)(1 - e^{-\alpha})$$

The *expected* no. of probes in an unsuccessful search, which is also the expected no. of probes needed to insert the $(n+1)$-*st* item is

$$C'_n = \sum_{k=0}^{n} (k+1) p_k = \sum_{k=0}^{n} \binom{k+2}{2} g_k$$

$$C_n' = \sum_{k=0}^{n}(k+1)p_k = \sum_{k=0}^{n}\binom{k+2}{2} g_k$$

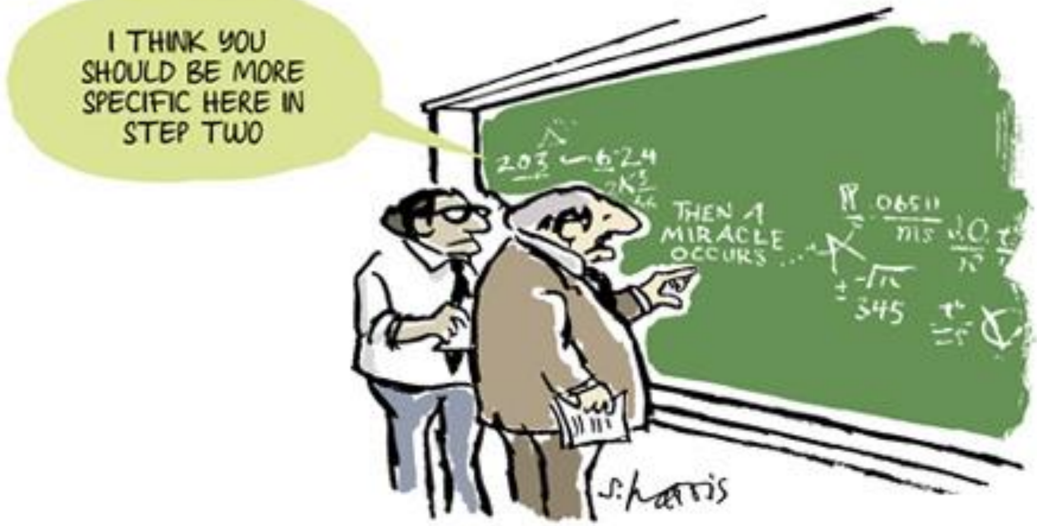
$$= \frac{1}{2}\left(\underbrace{\sum_{k=0}^{n}(k+1)g_k}_{\sum_{k=0}^{n} p_k = 1} + \underbrace{\sum_{k=0}^{n}(k+1)^2 g_k}_{Q_1(m,n)}\right)$$

$$Q_1(m,n) = m^{-n}\sum_{k=0}^{n}\binom{n}{k}(k+1)^{k+1}(m-k-1)^{n-k-1}(m-n-1)$$

$$= \sum_{k=0}^{n}(k+1)\frac{(n)_k}{m^k}$$

Ex. 6.4.27

Knuth, Vol. 3



Abel's binomial theorem
(see Knuth Eq. 1.2.6-(16))

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x(x-kz)^{k-1}(y+kz)^{n-k}$$

Unsuccessful search

$$C'_n = \frac{1}{2}(1 + Q_1(m,n))$$

$$Q_1(m,n) = \sum_{k=0}^{n}(k+1)\frac{(n)_k}{m^k} \le \sum_{k=0}^{n}(k+1)\left(\frac{n}{m}\right)^k < \left(\frac{1}{1-\alpha}\right)^2$$

$$(n)_k = n(n-1)\ldots(n-k+1) \le n^k$$

$$\sum_{k\ge 0}(k+1)\alpha^k = \left(\frac{1}{1-\alpha}\right)^2$$

The birth of Knuth's style *Analysis of Algorithms*…

Successful search / Construction time

The expected number of probes in a search of randomly selected item is

$$C_n = \frac{1}{n}\sum_{k=0}^{n-1} C'_k < \frac{1}{2}\left(1 + \frac{1}{1-\alpha}\right)$$

The expected number of probes in the construction of the table is

$$n\, C_n = \sum_{k=0}^{n-1} C'_k$$

The "parking problem"
[Knuth (1962)] [Konheim-Weiss (1966)]

A one-way street contains m parking spots

n cars arrive, one after the other

The i-*th* car chooses a random number h_i between 1 and m and parks in the first free spot at or after location h_i, if there is one

Exercise: What is the probability that all cars find a parking spot?

Linear Probing: Theory vs. Practice

In practice, we *cannot* use
a truly random hash function

Does linear probing still have a
constant expected time per operation
when more realistic hash functions are used?

For chaining, 2-independence,
or just "universality", was enough

How much independence is
needed for linear probing?

Linear Probing: Theory vs. Practice

5-independence suffices for linear probing!
[Pagh-Pagh-Rŭzíc (2009)]

4-independence does not suffice!
[Pătraşcu-Thorup (2010)]

k-independence

Definition:

$X_1, X_2, \dots, X_k$ are independent iff
for every $x_1, x_2, \dots, x_k$, we have

$$\Pr[X_1 = x_1, X_2 = x_2, \dots, X_k = x_k] = \Pr[X_1 = x_1] \Pr[X_2 = x_2] \dots \Pr[X_k = x_k]$$

Definition:

$X_1, X_2, \dots, X_n$ are k-independent iff for every distinct $i_1, i_2, \dots, i_k$, $X_{i_1}, X_{i_2}, \dots, X_{i_k}$ are independent

Families of k-independent hash functions

Let H be a family of hash functions from U to V.
H is k-independent iff for every *distinct* $x_1, x_2, \ldots, x_k \in U$, $h(x_1), h(x_2), \ldots, h(x_k)$ are independent, when h is chosen at random from H

We usually require that for every $x \in U$, $h(x)$ is (almost) uniformly distributed on V

If H is k-independent and $H' = \{ f(h(x)) \mid h \in H \}$, for some function f, then H' is also k-independent

Polynomial hash functions

Lemma: If F is a field, then

$$H = \{ \sum_{i=0}^{k-1} a_i x^i \mid a_0, a_1, \ldots, a_k \in F \}$$

is a k-independent family of hash functions

Corollary: If p is a prime, and m is arbitrary, then

$$H = \{ ((\sum_{i=0}^{k-1} a_i x^i) \bmod p) \bmod m \mid a_0, a_1, \ldots, a_k \in F \}$$

is a k-independent family of hash functions

When $p \gg m$, $h(x)$ is almost uniformly distributed on $[m] = \{0,1, \ldots, m-1\}$

Polynomial hash functions

$$h(x) = \sum_{i=0}^{k-1} a_i x^i$$

$x_1, x_2, \ldots, x_k \in F$ distinct

$y_1, y_2, \ldots, y_k \in F$ (not necessarily distinct)

$$h(x_1) = y_1, h(x_2) = y_2, \ldots, h(x_k) = y_k$$

$$\begin{pmatrix} 1 & x_1 & \ldots & x_1^{k-1} \\ 1 & x_2 & \ldots & x_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & \ldots & x_k^{k-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix}$$

Unique solution!

Vandermonde Determinant

$$\det\begin{pmatrix} 1 & x_1 & \dots & x_1^{k-1} \\ 1 & x_2 & \dots & x_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & \dots & x_k^{k-1} \end{pmatrix} = \prod_{i<j}(x_j - x_i)$$

Tabulation-based hash functions

[Carter-Wegman (1979)]
[Pătraşcu-Thorup (2010)]

$$h(x_1, x_2, \ldots, x_c) \;=\; h_1(x_1) \oplus h_2(x_2) \oplus \cdots \oplus h_c(x_c)$$

$$h_1, h_2, \ldots, h_c : \left[u^{1/c}\right] \to [2^k]$$

$$h : [u] \to [2^k]$$

$$[u] = \{0,1,\ldots,u-1\}$$

$h_1, h_2, \ldots, h_c$ may be implemented using small look-up tables

Very efficient in practice

Tabulation-based hash functions

[Carter-Wegman (1979)]
[Pătraşcu-Thorup (2010)]

$$h(x_1, x_2, \dots, x_c) \;=\; h_1(x_1) \oplus h_2(x_2) \oplus \cdots \oplus h_c(x_c)$$

If $h_1, h_2, \dots, h_c$ are independently chosen from a uniform 2-independent family, then h is 2-independent

If $h_1, h_2, \dots, h_c$ are independently chosen from a uniform 3-independent family, then h is 3-independent

Not 4-independent!

$$h(x_1, y_1) \oplus h(x_1, y_2) \oplus h(x_2, y_1) \oplus h(x_2, y_2) = 0$$

Tabulation-based hash functions

[Thorup-Zhang (2012)]

$$h(x,y) \;=\; h_1(x) \oplus h_2(y) \oplus h_3(x+y)$$

If h_1, h_2, h_3 are independently chosen from a 5-independent family, then h is 5-independent

Higher independence possible at the cost of more table look-ups

Linear probing with bounded independence

[Pagh-Pagh-Rŭzíc (2009)]

[Pătraşcu-Thorup (2010)]

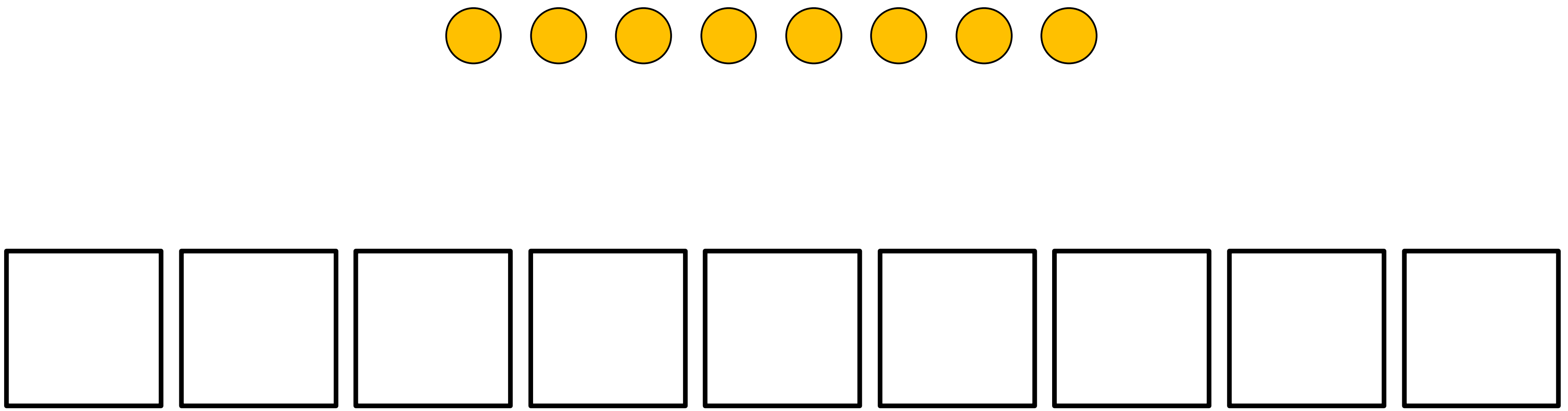
<table>
<tr><th>Independence</th><th>2</th><th>3</th><th>4</th><th>5</th></tr>
<tr><td>Search time</td><td>$\Theta(\sqrt{n})$</td><td colspan="2">$\Theta(\log n)$</td><td>$\Theta(1)$</td></tr>
<tr><td>Construction time</td><td colspan="2">$\Theta(n \log n)$</td><td colspan="2">$\Theta(n)$</td></tr>
</table>

Upper bounds hold for *any* set of keys and *any* family with the specified independence

Lower bounds hold for *some* sets of keys and *some* families with the specified independence

Balls in Bins

Throw n balls randomly into m bins



All throws are uniform and (partially-)independent

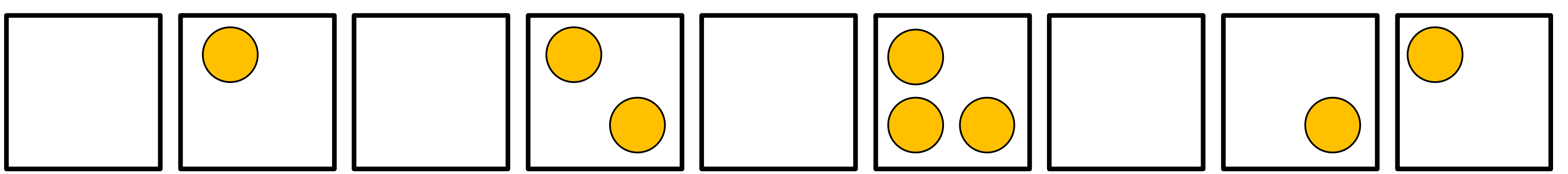
Balls in Bins

Throw n balls randomly into m bins

Let X be the number of balls that fall into a specific bin, e.g., the first

Let X_i be 1 if the i-th ball falls into the specific bin, and 0 otherwise

We want to bound the probability that X is large



Tail bounds

Markov's inequality:

$$\text{If } X \geq 0,\ \Pr[X \geq b\mu] \leq \frac{1}{b}$$

Chebyshev's inequality:

$$\Pr[|\mathrm{X} - \mu| \geq b\mu] = \Pr[(X-\mu)^2 \geq b^2\mu^2]$$

$$\leq \frac{E[(X-\mu)^2]}{b^2\mu^2} = \frac{Var[X]}{b^2\mu^2}$$

Higher (even) moments:

$$\Pr[|\mathrm{X} - \mu| \geq b\mu] = \Pr[(X-\mu)^k \geq b^k\mu^k]$$

$$\leq \frac{E[(X-\mu)^k]}{b^k\mu^k} = \frac{M_k[X-\mu]}{b^k\mu^k}$$

Tail bounds

Chernoff bound:

If $X_1, X_2, \ldots, X_n$ are *independent* indicators, $X = \sum_{i=1}^{n} X_i$, $\mu = E[X]$, and $\delta > 0$, then

$$\Pr[X \geq (1+\delta)\mu] < \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}$$

Proof: Apply Markov's inequality to e^{tX} and choose $t = \ln(1+\delta)$

Chernoff bound is stronger.

But it requires *complete* *independence*.

Computing moments

$$X = \sum_{i=1}^{n} X_i \qquad X_{\boldsymbol{i}} = \begin{cases} 1 & \text{w.p.} & p \\ 0 & \text{w.p.} & 1-p \end{cases}$$

$$\mu = E[X] = np$$

$$X - \mu = \sum_{i=1}^{n} Y_i \qquad Y_i = X_i - p = \begin{cases} 1-p & \text{w.p.} & p \\ -p & \text{w.p.} & 1-p \end{cases}$$

$$E[Y_i] = 0$$

$$E[(X-\mu)^k] = E[(\sum_{i=1}^{n} Y_i)^k]$$

$$= E[\sum_{i_1,i_2,\ldots,i_k} Y_{i_1} Y_{i_2} \ldots Y_{i_k}]$$

$$= \sum_{i_1,i_2,\ldots,i_k} E[Y_{i_1} Y_{i_2} \ldots Y_{i_k}]$$

$$E[Y_{i_1} Y_{i_2} \ldots Y_{i_k}] \stackrel{?}{=} E[Y_{i_1}] E[Y_{i_2}] \ldots E[Y_{i_k}]$$

Computing moments

If $X_1, X_2, \dots, X_n$ are k-independent,
then so are $Y_1, Y_2, \dots, Y_n$

If $i_1, i_2, \dots, i_k$ are distinct, then

$$E[Y_{i_1} Y_{i_2} \dots Y_{i_k}] = E[Y_{i_1}]\, E[Y_{i_2}] \dots E[Y_{i_k}] = 0$$

If i_1 differs from $i_2, \dots, i_k$, then

$$E[Y_{i_1} Y_{i_2} \dots Y_{i_k}] = E[Y_{i_1}]\, E[Y_{i_2} \dots Y_{i_k}] = 0$$

If $i \neq j$, then

$$E[Y_i\, Y_i\, Y_j\, Y_j] = E[Y_i^2]\, E[Y_j^2]$$

Computing moments

$$Y_{\boldsymbol{i}} = \begin{cases} 1-p & \text{w.p.} \quad p \\ -p & \text{w.p.} \quad 1-p \end{cases}$$

$$E\left[Y_i^k\right] = p(1-p)^k + (1-p)(-p)^k$$
$$= p(1-p)\left((1-p)^{k-1} - (-p)^{k-1}\right)$$
$$\leq \ p(1-p) \ \leq \ p$$

If $X_1, X_2, \ldots, X_n$ are 2-independent

$$E[(X-\mu)^2] = E[(\sum_{i=1}^n Y_i)^2]$$
$$= \ \sum_{i=1}^n E\left[Y_i^2\right] = np(1-p) < \mu$$

Computing moments

If $X_1, X_2, \ldots, X_n$ are 4-independent

$$E[(X-\mu)^4] = E[(\textstyle\sum_{i=1}^{n} Y_i)^4]$$

$$= 3\sum_{i \neq j} E[Y_i^2]E[Y_j^2] + \sum_{i} E[Y_i^4]$$

Why?

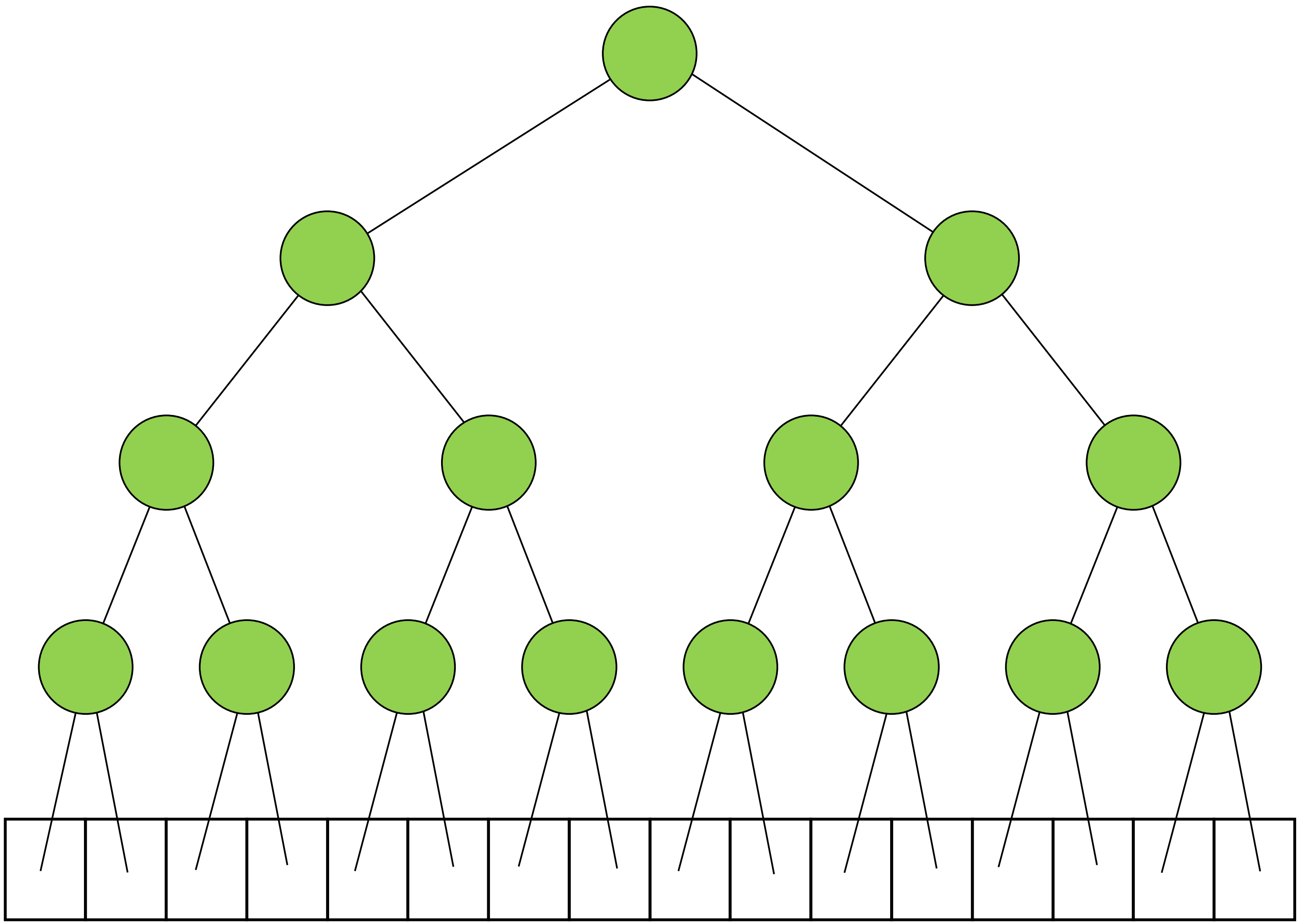
$$\leq \; 3n^2p^2 + np \; = \; 3\mu^2 + \mu$$

If $X_1, X_2, \ldots, X_n$ are k-independent, where $k = O(1)$ and $\mu = \Omega(1)$, then

$$E[(X-\mu)^k] = O(\mu^{k/2})$$

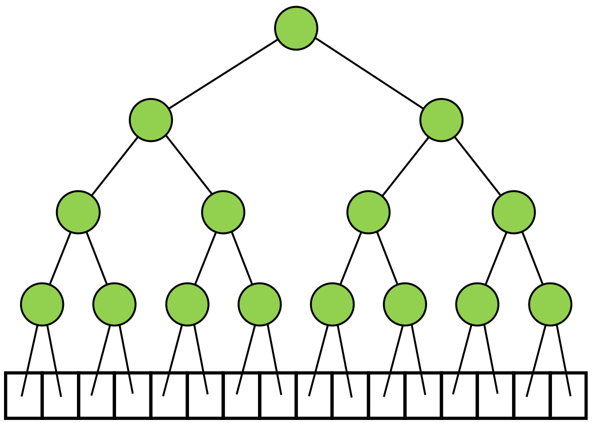
(We only need 4-*th* moments)

Planting a binary tree



Crowded nodes

[Pătraşcu-Thorup (2010)]



Simplifying assumptions:

m is a power of 2

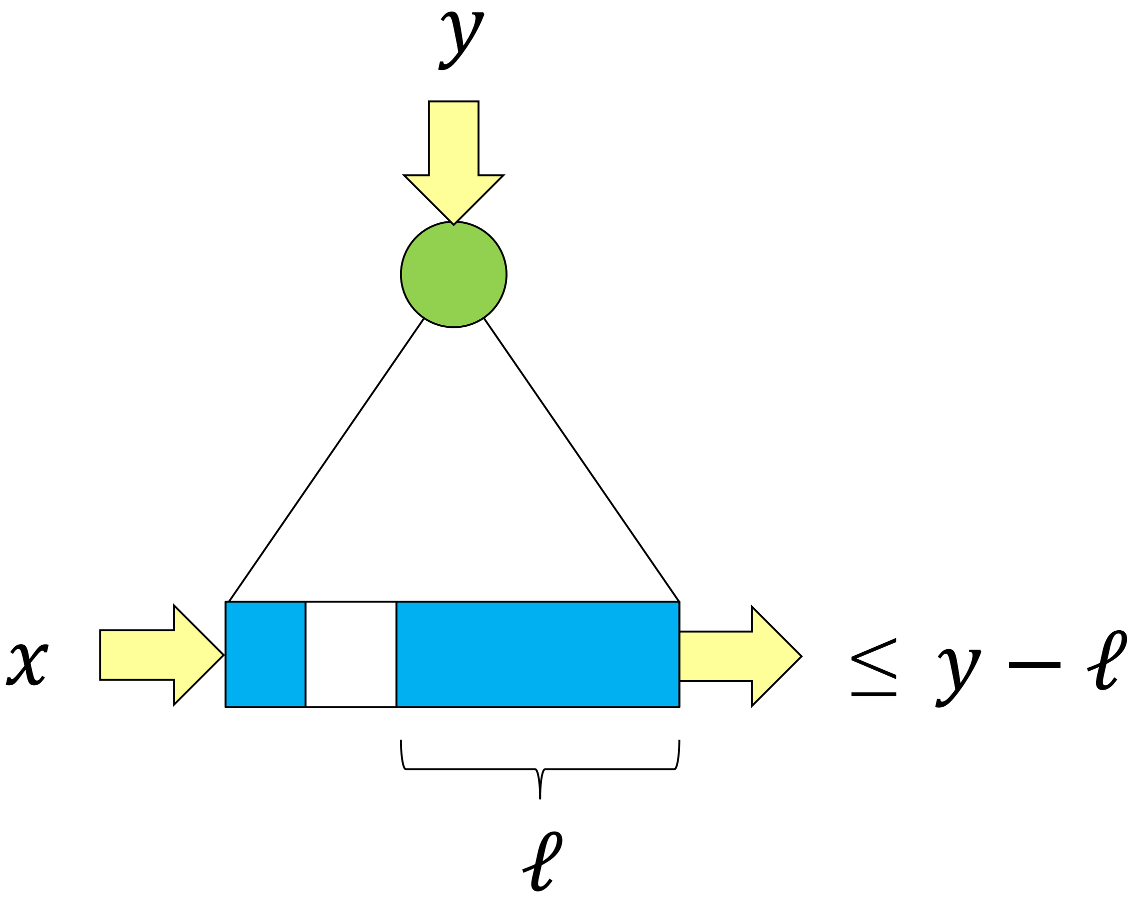
$\alpha = n/m \leq 2/3$

A node at height i corresponds to 2^i consecutive cells in the table

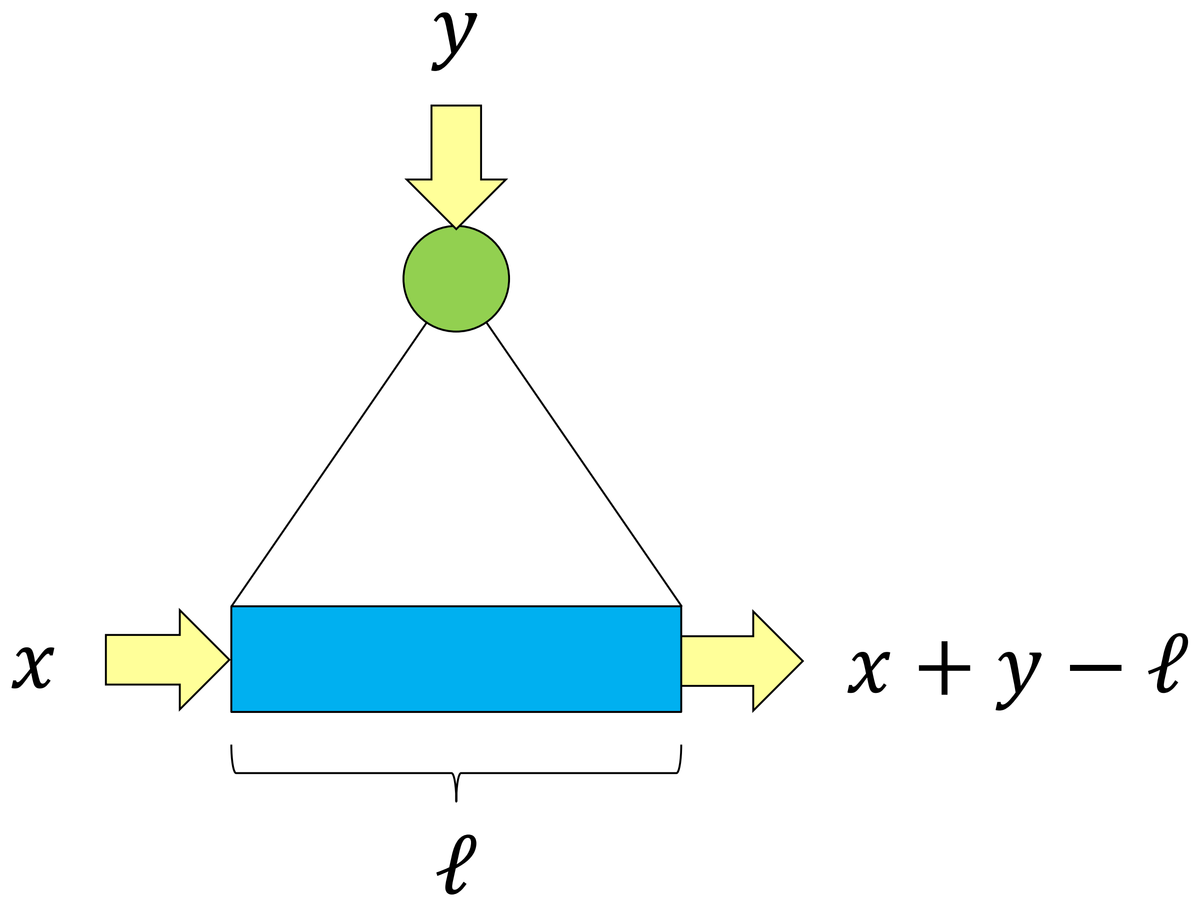
A node at height i is *crowded*, if at least $(3/4)2^i$ items are mapped into its interval

The final locations of items mapped into an interval may be outside the interval

Simple observation I



Simple observation II

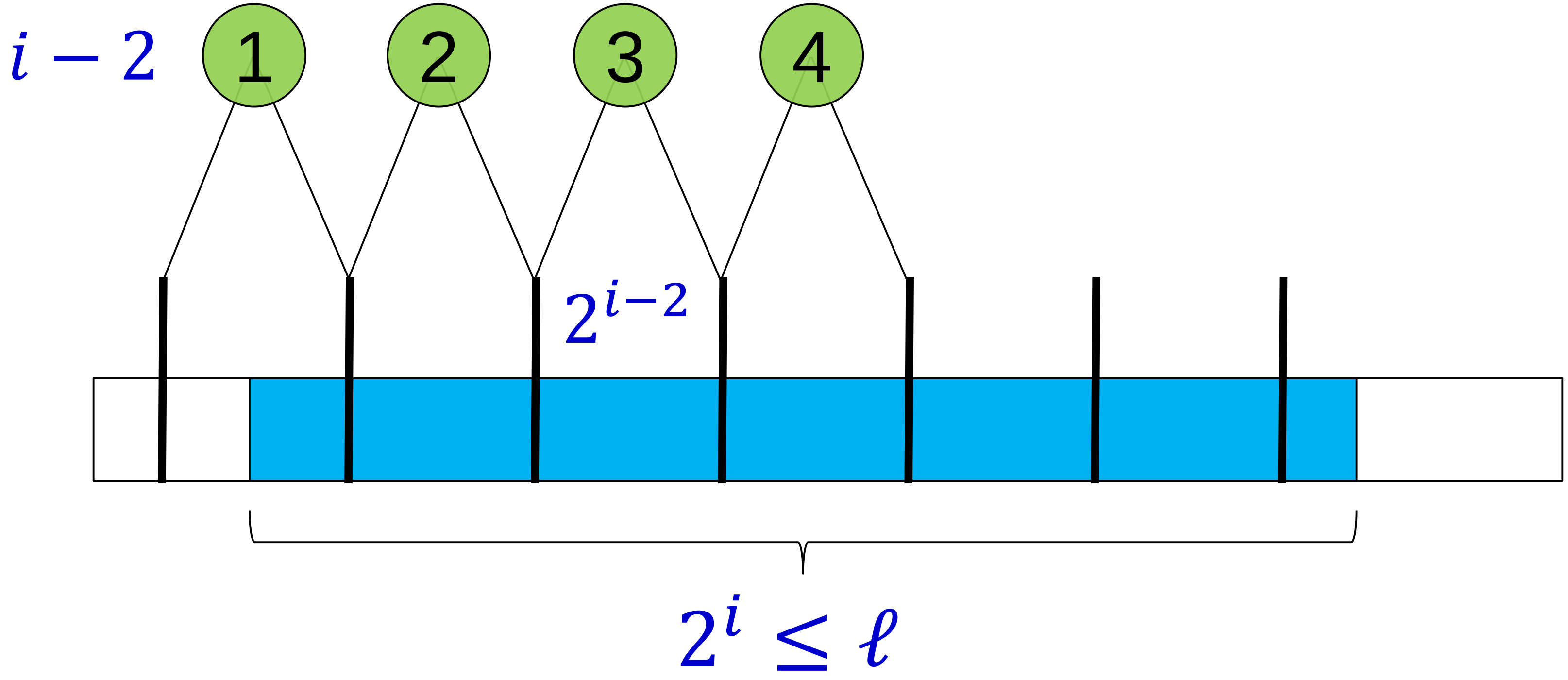


Main observation

[Pătraşcu-Thorup (2010)]

Consider a run of length $2^i \leq \ell$, where $i > 2$

At least one of the first four nodes at level $i - 2$ whose last cell belongs to the run is crowded

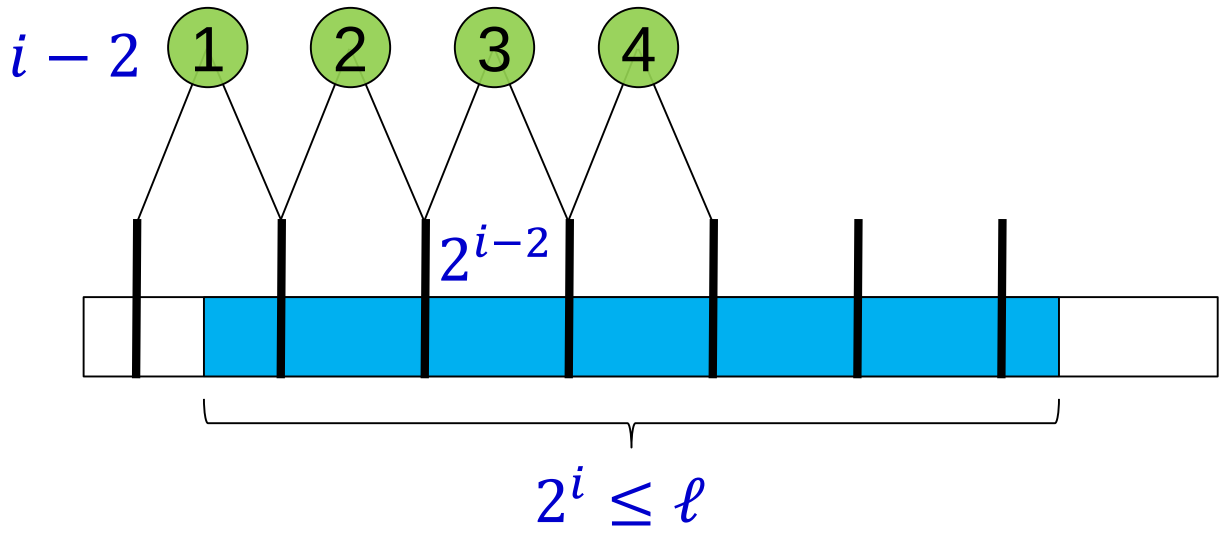


Proof of main observation

Just before the run, there is an empty cell.
Thus, if 1 is not crowded, it contributes
less than $(3/4)2^{i-2}$ items to the run

If 2,3,4 are not crowded, then each of their
intervals can absorb at least $(1/4)2^{i-2}$ items

Thus, if none of 1,2,3,4 is crowded, the run ends at or
before the interval of 4 and its length is less than $4 \cdot 2^{i-2} = 2^i$



Probability of being crowded

Assume that $\alpha = \frac{n}{m} \leq \frac{2}{3}$

Consider a node at height i

Throwing n balls into $m/2^i$ bins

$$\mu = n/(m/2^i) = \alpha\, 2^i \leq (2/3)2^i$$

$$\Pr\left[\mathrm{X} \geq \frac{3}{4}2^i\right] \leq \Pr[|\mathrm{X} - \mu| \geq b\mu]$$

$$\leq \frac{E\left[(X-\mu)^k\right]}{b^k \mu^k} = O\left(\frac{1}{b^k \mu^{k/2}}\right) = O\left(2^{-ik/2}\right)$$

$$b \geq (3/4 - \alpha)/\alpha \geq 3/24$$

Construction time

[Pătraşcu-Thorup (2010)]

Let $\ell_1, \ell_2, \ldots$, where $\sum_i \ell_i = n$, be the length of the consecutive runs in the table after inserting the n items

The cost of the construction is at most $\sum_i \ell_i^2$

Runs of length $\ell_i < 4$ contribute only $O(n)$

By the main observation, if $2^i \le \ell_i < 2^{i+1}$, then at least one of the first four nodes at level $i-2$ whose last cell is in the run is crowded.

Each node corresponds to at most one run.

$$\sum_i \ell_i^2 = O\left(\sum_v 2^{\,2\cdot\text{height}(v)}\,[v \text{ crowded}]\right)$$

Construction time

[Pătraşcu-Thorup (2010)]

$$E\left[\sum_i \ell_i^2\right] = O\left(\sum_v 2^{\,2\cdot\text{height}(v)}\ \Pr[v \text{ crowded}]\right)$$

$$= O\left(\sum_{i=0}^{\log_2 m} \frac{m}{2^i}\, 2^{2i}\, 2^{-\frac{ki}{2}}\right) = O\left(n \sum_{i=0}^{\log_2 m} 2^i\, 2^{-\frac{ki}{2}}\right)$$

If $k = 2$, we get $O(n \log n)$

If $k = 4$, we get $O(n)$

Query time (successful/unsuccessful)

[Pătraşcu-Thorup (2010)]

If $h(k)$ is in a run of length ℓ, then the search time is $O(\ell)$

If $h(k)$ is in a run of length $2^i \leq \ell < 2^{i+1}$, then at least one of 12 nodes at height $i-2$ associated with $h(k)$ is crowded

$$p(i) = \Pr[v \text{ crowded}] = O\left(2^{-ik'/2}\right) \;,\; \text{height}(v) = i$$

$$E[\ell] \leq 3 + 12 \sum_{i \geq 2} p(i-2) \cdot 2^{i+1}$$

Query time (successful/unsuccessful)

[Pătraşcu-Thorup (2010)]

$$E[\ell] \le 3 + 12 \sum_{i \ge 2} p(i-2) \cdot 2^{i+1} = O\left(\sum_{i=0}^{\log_2 m} 2^i \, 2^{-k' i/2} \right)$$

k' - The independence after *conditioning* on the hash value of the key searched

$$k' = k - 1$$

If $k = 2$, we get $O(\sqrt{n})$

If $k = 3$, we get $O(\log n)$

If $k = 5$, we get $O(1)$

Why 12?

The constant 12 itself, of course, if *not* too important.
The important thing is that it *is* a constant