File access

2 standard methods for accessing data:

- sequential access
- random access: access via index or ID (key) for data element

Questions

1. What can be done using only Sequential access?
2. How can one implement Random access?

Merge Sort

procedure MergeSort(A, f, l):
{ Input: Array A with first index f and last index l }
{ Output: Sorted array, A, with same entries as input A }

if ($f < l$) **then**
  $m \leftarrow (f + l)$ div 2
  MergeSort(A, f, m)
  MergeSort(A, $m + 1$, l)
  MergeArrays($A[f..m]$, $A[m + 1..l]$, C)
  Copy C to A

MergeSort(A, 1, length(A));

Analysis of Merge Sort

Let $T(n)$ be the maximum number of comparisons MergeSort uses if length$(A)= n$.

$$\begin{aligned}
T(n) &\leq T\left(\left\lceil\frac{n}{2}\right\rceil\right)+T\left(\left\lfloor\frac{n}{2}\right\rfloor\right)+M\left(\left\lceil\frac{n}{2}\right\rceil,\left\lfloor\frac{n}{2}\right\rfloor\right)\\
&\leq T\left(\left\lceil\frac{n}{2}\right\rceil\right)+T\left(\left\lfloor\frac{n}{2}\right\rfloor\right)+\left(\left\lceil\frac{n}{2}\right\rceil+\left\lfloor\frac{n}{2}\right\rfloor-1\right)\\
&\leq T\left(\left\lceil\frac{n}{2}\right\rceil\right)+T\left(\left\lfloor\frac{n}{2}\right\rfloor\right)+n-1
\end{aligned}$$

$T(n) \in \Theta(n \log n)$.

Analysis of Merge Sort

$$T(n) \leq T\left(\left\lceil \frac{n}{2} \right\rceil\right) + T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n - 1$$

Prove by induction: $T(n) \leq n \log_2(n)$, if $n = 2^j$ for some integer j

Base case: $n = 1$. $1 \cdot \log_2(1) = 0 = T(1)$.

Induction hypothesis: For all $k < n$, where $k = 2^i$, $T(k) \leq k \log_2(k)$.

Induction step (prove for n):

$$\begin{aligned}
T(n) &\leq T\left(\left\lceil \frac{n}{2} \right\rceil\right) + T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n - 1 \\
&\leq 2T\left(\frac{n}{2}\right) + n - 1 \\
&\leq 2\frac{n}{2} \log_2\left(\frac{n}{2}\right) + n - 1 \\
&\leq n(\log_2 n - 1) + n - 1 \\
&\leq n \log_2 n
\end{aligned}$$

Analysis of Merge Sort

$T(n) \leq n \log_2(n)$, if $n = 2^j$ for some integer j.

If $n \neq 2^j$ for any integer j, $T(n) \leq T(n')$ where n' is the next power of 2 larger than n.

In general $T(n) \leq (2n) \log_2(2n) \leq 2n \log_2 n + 2n$.
So $T(n) \in \Theta(n \log n)$.

Merging more than 2 lists

Problem:

Input: 3 lists, A, B and C are sorted.
Output: 1 sorted list, D, containing the entries of $A \cap B \cap C$.

Intersecting 3 lists

Input: 3 lists, A, B and C are sorted.
Output: 1 sorted list, D, containing the entries of $A \cap B \cap C$.

Merge Step:

- Compare current records of A, B and C.
- If all the same, put record in D. Advance to next record in all of A, B and C.
- If current in A is smaller than current in either B or C, advance to next record in A. (Do same for B and C.)

Merging 2 lists — recall


```
procedure MergeFiles(A, B, C):
    open(A); open(B); open(C); fA,fB,fC ← false;
    if (isEndOfFile(A) and isEndOfFile(B)) then Stop with C empty
    if (not isEndOfFile(A)) then currentA ← readNext(A); fA← true;
    if (not isEndOfFile(B)) then currentB ← readNext(B); fB ← true;
    while (fA and fB) do
        if (currentA ≤ currentB) then
            writeNext(currentA,C)
            if (not isEndOfFile(A)) then currentA ← readNext(A)
            else fA ← false
        else
            writeNext(currentB,C)
            if (not isEndOfFile(B)) then currentB ← readNext(B)
            else fB ← false
    Starting with the current record in the input file which is not at EOF,
        copy the remaining records to C
    close(A); close(B); close(C)
```

Random access API

random access: access via ID (key) for data element

Operations:
findElm(ID)
insertElm(ID,elementData)
deleteElm(ID)
open()
close()

Examples:

- dictionaries in Python
- arrays in Java — with ID = index in array

Random access API

How do you implement random access?

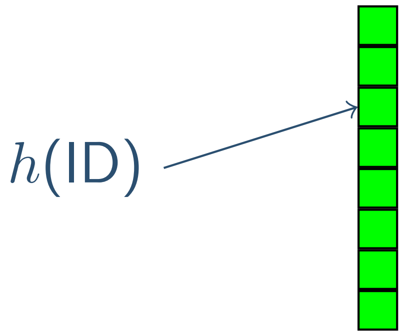
One solution: hashing.

Hashing

Idea:

- store values in an array A
- ID determines index where stored

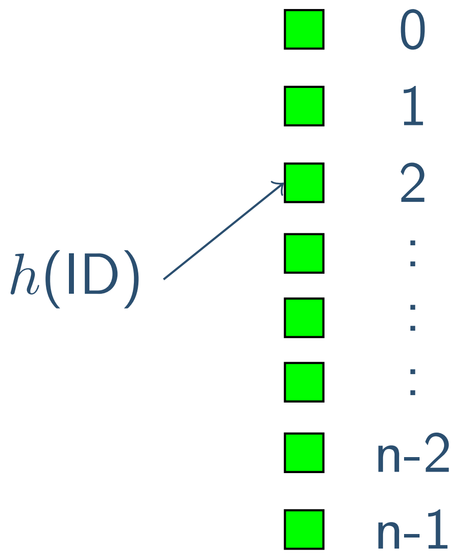
Hash function: h



$h(\text{ID}) = \text{index in } A$

Hashing

Hash function: h



$h(\text{ID}) = \text{index in } A$

Example: Assume ID is an integer, $|A| = n$.

$$h(\text{ID}) \;=\; \text{ID} \pmod{n}$$

Note: $h(\text{ID}) \in \{0, 1, 2, ..., n-1\}$, so legal index.

Hashing

Hash function: h

Example: Assume ID is an integer, $|A| = k$.

$$h(\text{ID}) = \text{ID} \pmod{k}$$

Note: $h(\text{ID}) \in \{0, 1, 2, ..., k-1\}$, so legal index.

Let $k = 41$.

$$\begin{aligned} h(46) &= 5 && \text{since } 1 \cdot 41 + 5 = 46 \\ h(12) &= 12 && \text{since } 0 \cdot 41 + 12 = 12 \\ h(100) &= 18 && \text{since } 2 \cdot 41 + 18 = 100 \\ h(479869) &= 5 && \text{since } 11704 \cdot 41 + 5 = 479869 \end{aligned}$$

Hashing

Why not let $h(x) = x$?

Hashing

Why not let $h(x) = x$?

Example: IDs (keys) are CPR-numbers.

CPR-number: $180782\text{-}2345 \in \{0, 1, 2, ..., 10^{10} - 1\}$
10^{10} bytes > 9 GB (just to store one byte per key).

Hashing

Why not let $h(x) = x$?

Example: IDs (keys) are CPR-numbers.

CPR-number: $180782\text{-}2345 \in \{0, 1, 2, ..., 10^{10} - 1\}$
10^{10} bytes > 9 GB (just to store one byte per key).

Huge waste of space!

Suppose you only need to store 6 million records.
You need to allocate space for 10^{10} records.
You are allocating more than 1000 records for every one used.

Hashing

Why not let $h(x) = x$?

Example: IDs (keys) are CPR-numbers.

CPR-number: $180782\text{-}2345 \in \{0, 1, 2, ..., 10^{10} - 1\}$
10^{10} bytes > 9 GB (just to store one byte per key).

Huge waste of space!

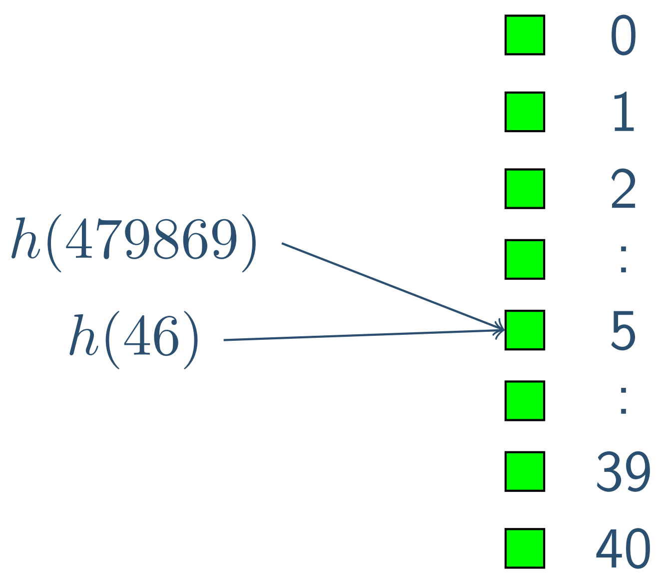
Suppose you only need to store 6 million records.
You need to allocate space for 10^{10} records.
You are allocating more than 1000 records for every one used.

If the keys are 64-bit integers...

Collisions

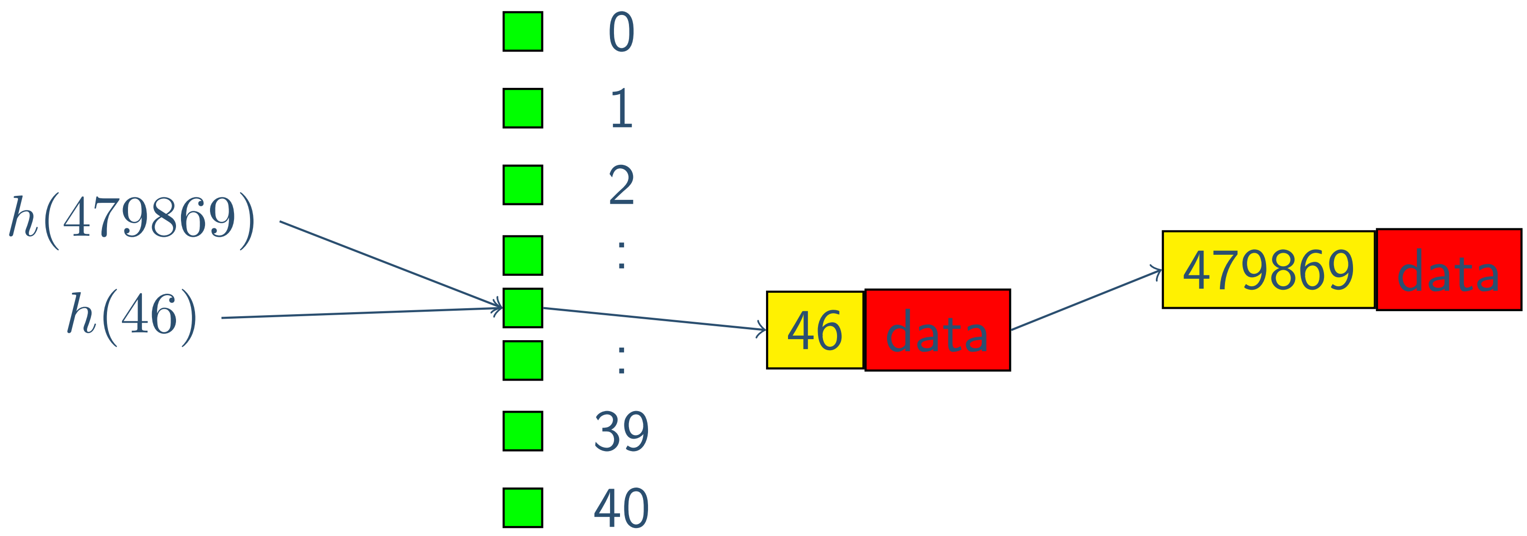
Let $k = 41$.

$$
\begin{aligned}
h(46) &= 5 && \text{since } 1 \cdot 41 + 5 = 46 \\
h(12) &= 12 && \text{since } 0 \cdot 41 + 12 = 12 \\
h(100) &= 18 && \text{since } 2 \cdot 41 + 18 = 100 \\
h(479869) &= 5 && \text{since } 11704 \cdot 41 + 5 = 479869
\end{aligned}
$$



1st solution: Chaining

For each cell in array, have a linked list for elements stored there.



1st solution: Chaining

Assume computing $h(x)$ takes constant time.
Worst case: How long does it take to find a record from a key if there are no collisions?
How long does it take if there are at most s collisions for any cell?

A. $\Theta(1)$, $\Theta(1)$.

B. $\Theta(1)$, $\Theta(k)$.

C. $\Theta(1)$, $\Theta(s)$.

D. $\Theta(k)$, $\Theta(s)$.

E. $\Theta(k)$, $\Theta(k \cdot s)$.

Vote at m.socrative.com. Room number 415439.

1st solution: Chaining

Assume computing $h(x)$ takes constant time.
Worst case: How long does it take to find a record from a key if there are no collisions?
How long does it take if there are at most s collisions for any cell?

C. $\Theta(1)$, $\Theta(s)$.

So we want short lists, few collisions.

Can collisions be avoided?

Suppose the hash function h is fixed.
Suppose the total number of items in the domain of h is d and the array has size k.

Can collisions be avoided?

Suppose the hash function h is fixed.
Suppose the total number of items in the domain of h is n and the array has size k.

Then d/k elements could hash to some cell.
It depends on the relation between the hash function and the data set.

Can collisions be avoided?

Suppose the hash function h is fixed.
Suppose the total number of items in the domain of h is d and the array has size k.

Then d/k elements could hash to some cell.
It depends on the relation between the hash function and the data set.

In the worst case all n elements being hashed has to the same cell.
Time: $\Theta(n)$.

Can collisions be avoided?

If n (number of elements hashed) $> k$ (size of array), there is at least one collision (Pigeon Hole Principle).

The best hash functions "appear" to hash numbers to random cells.

The birthday paradox

Situation: There are n random people in a room.
Question: Are there two that have the same birthday? (Ignore year.)

The birthday paradox

Situation: There are n random people in a room.
Question: Are there two that have the same birthday? (Ignore year.)

n	Probability for 2 with same birthday
0	0
1	0
2	$1/365$
.	
.	?
.	
366	1

Question: For which n is the probability $\geq 1/2$?

The birthday paradox

Let s_n = probability none of n have same birthday.

$$s_n = s_{n-1} \cdot \frac{365 - (n-1)}{365}$$

Note: $s_1 = 1$.

n	s_n
1	1
2	$1 \cdot \frac{364}{365}$
3	$1 \cdot \frac{364}{365} \cdot \frac{363}{365}$
4	$1 \cdot \frac{364}{365} \cdot \frac{363}{365} \cdot \frac{362}{365}$
.	.
.	.
.	.

The birthday paradox

Computing these s_n gives:

$$\begin{aligned} s_{22} &= 0.5243... \\ s_{23} &= 0.4972... \end{aligned}$$

So when is the probability $\geq 1/2$ that 2 have the same birthday?

The birthday paradox

Computing these s_n gives:

$$\begin{aligned} s_{22} &= 0.5243... \\ s_{23} &= 0.4972... \end{aligned}$$

So when is the probability $\geq 1/2$ that 2 have the same birthday?

$1 - s_{23} > 1 - 0.4973 = 0.5027 > 1/2$

Data mining

Data mining — techniques for finding patterns in collections of data.
Examples?

Data mining

Data mining — techniques for finding patterns in collections of data.

Examples:

- marketing
- investment analysis
- quality control
- loan risk management
- fraud detection
- identifying functions of particular genes (from DNA)

Data mining

Done on static data collections — data warehouses.
Use a snapshot of the database.

Data mining

Common forms:

- class description — identifying properties that characterize a given group of data items (who buys small cars)

Data mining

Common forms:

- class description — identifying properties that characterize a given group of data items (who buys small cars)
- class discrimination — identifying techniques that could be used to distinguish between groups (tell if current customer would buy a large car or a small)

Data mining

Common forms:

- class description — identifying properties that characterize a given group of data items (who buys small cars)
- class discrimination — identifying techniques that could be used to distinguish between groups (tell if current customer would buy a large car or a small)
- cluster analysis — finding groupings (people who see children's films have ages 4–10 and 25–40?)

Data mining

Common forms:

- class description — identifying properties that characterize a given group of data items (who buys small cars)
- class discrimination — identifying techniques that could be used to distinguish between groups (tell if current customer would buy a large car or a small)
- cluster analysis — finding groupings (people who see children's films have ages 4–10 and 25–40?)
- association analysis — finding links between data groups (people who buy pasta sauce also buy pasta)

Data mining

Common forms:

- class description — identifying properties that characterize a given group of data items (who buys small cars)
- class discrimination — identifying techniques that could be used to distinguish between groups (tell if current customer would buy a large car or a small)
- cluster analysis — finding groupings (people who see children's films have ages 4–10 and 25–40?)
- association analysis — finding links between data groups (people who buy pasta sauce also buy pasta)
- outlier analysis — finding data points which look wrong (credit card fraud)

Data mining

Common forms:

- class description — identifying properties that characterize a given group of data items (who buys small cars)
- class discrimination — identifying techniques that could be used to distinguish between groups (tell if current customer would buy a large car or a small)
- cluster analysis — finding groupings (people who see children's films have ages 4–10 and 25–40?)
- association analysis — finding links between data groups (people who buy pasta sauce also buy pasta)
- outlier analysis — finding data points which look wrong (credit card fraud)
- sequential pattern analysis — identifying patterns over time (climate patterns)

Data mining

Techniques:

- statistics
- database technology — giving data warehouses capability of presenting data as data cubes
  (data viewed from multiple perspectives — dimensions)

Data mining

Ethical and societal questions:

Is it OK that a store finds out that people who buy candy also buy chips and put them far apart?

Is it OK to find out and make public characteristics of people who commit crimes?