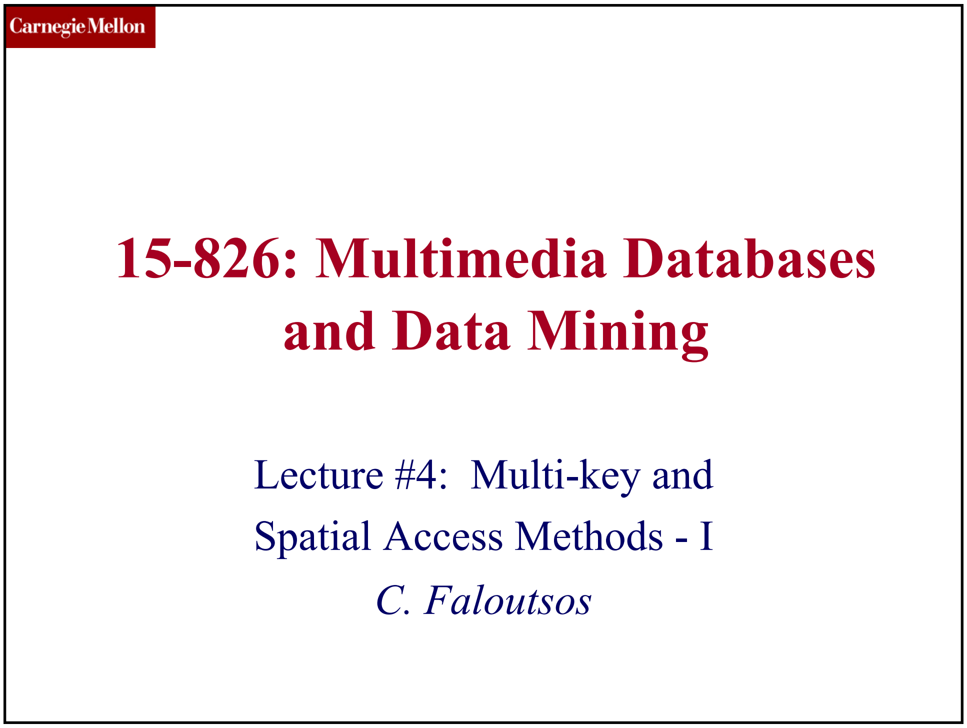
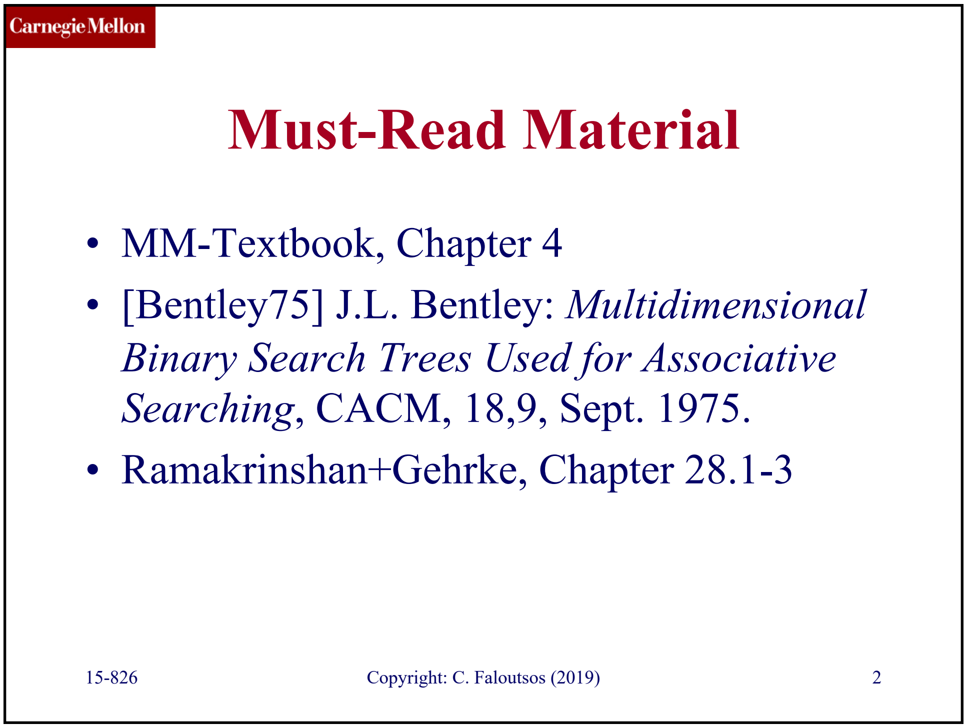


15-826: Multimedia Databases and Data Mining

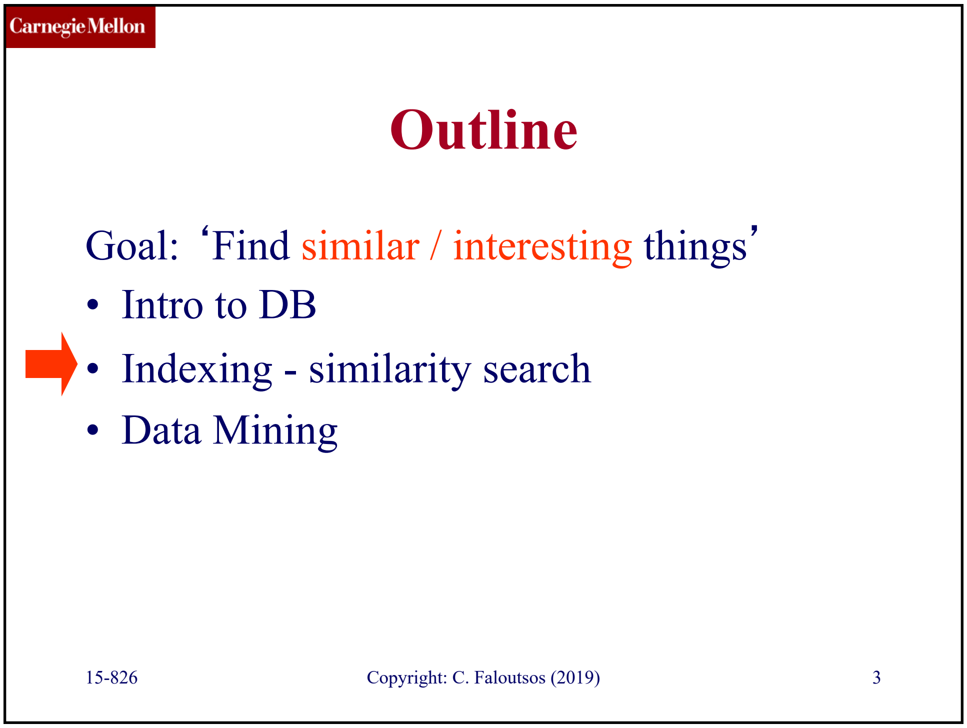
Lecture #4: Multi-key and Spatial Access Methods - I

C. Faloutsos



Must-Read Material

- MM-Textbook, Chapter 4
- [Bentley75] J.L. Bentley: *Multidimensional Binary Search Trees Used for Associative Searching*, CACM, 18,9, Sept. 1975.
- Ramakrinshan+Gehrke, Chapter 28.1-3

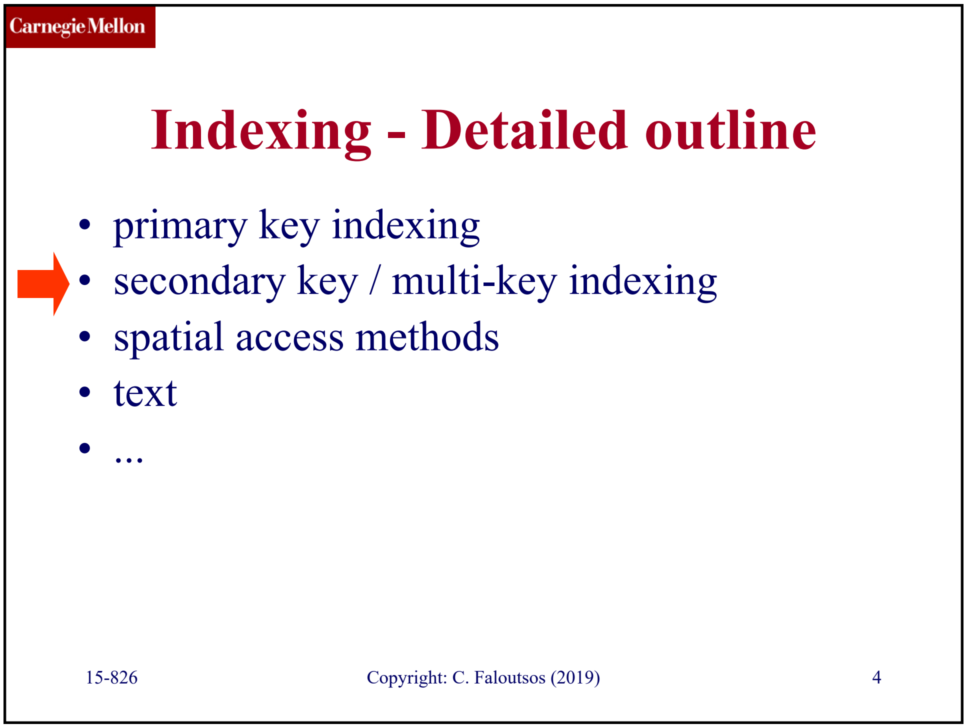


Carnegie Mellon

Outline

Goal: 'Find similar / interesting things'

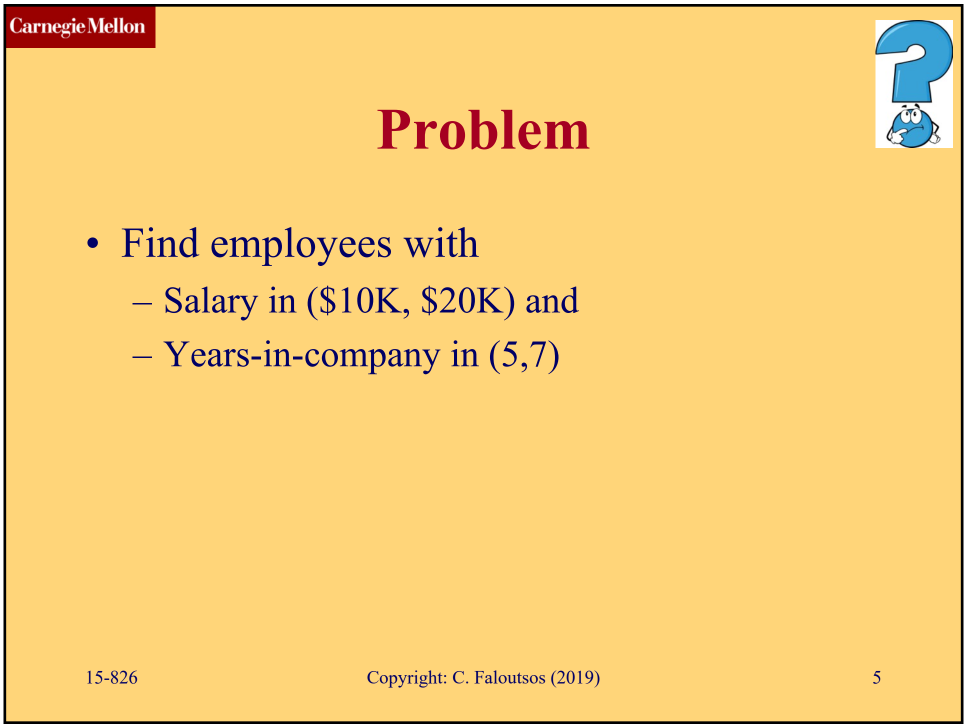
- Intro to DB
- ➡ Indexing - similarity search
- Data Mining



Carnegie Mellon

Indexing - Detailed outline

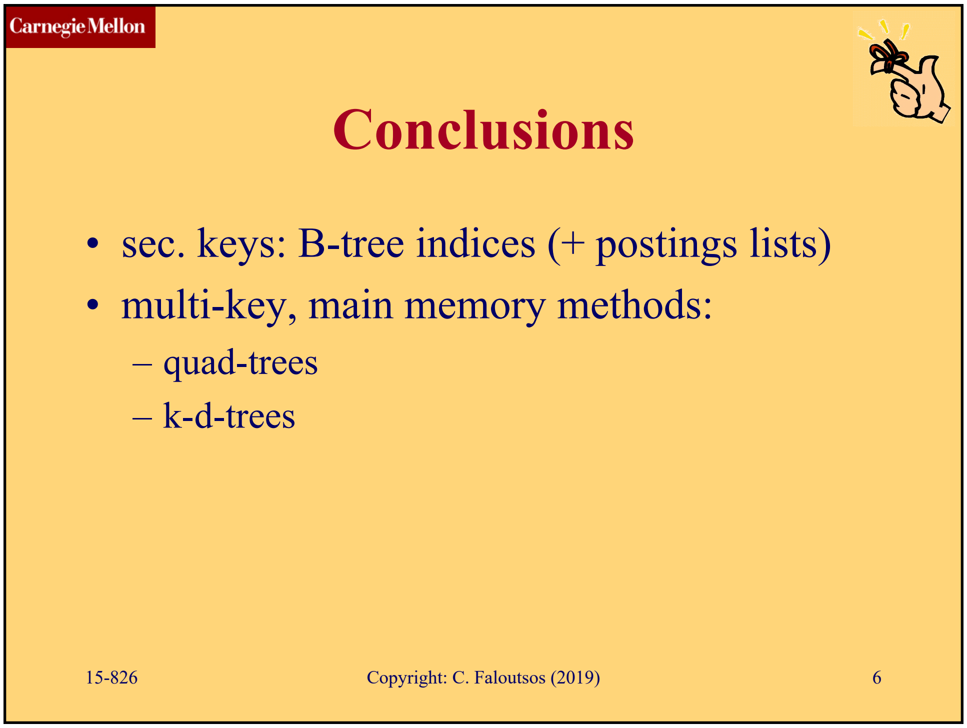
- primary key indexing
- ➡ secondary key / multi-key indexing
- spatial access methods
- text
- ...



Carnegie Mellon

Problem

- Find employees with
  - Salary in ($10K, $20K) and
  - Years-in-company in (5,7)



Carnegie Mellon

Conclusions

- sec. keys: B-tree indices (+ postings lists)
- multi-key, main memory methods:
  - quad-trees
  - k-d-trees

Carnegie Mellon

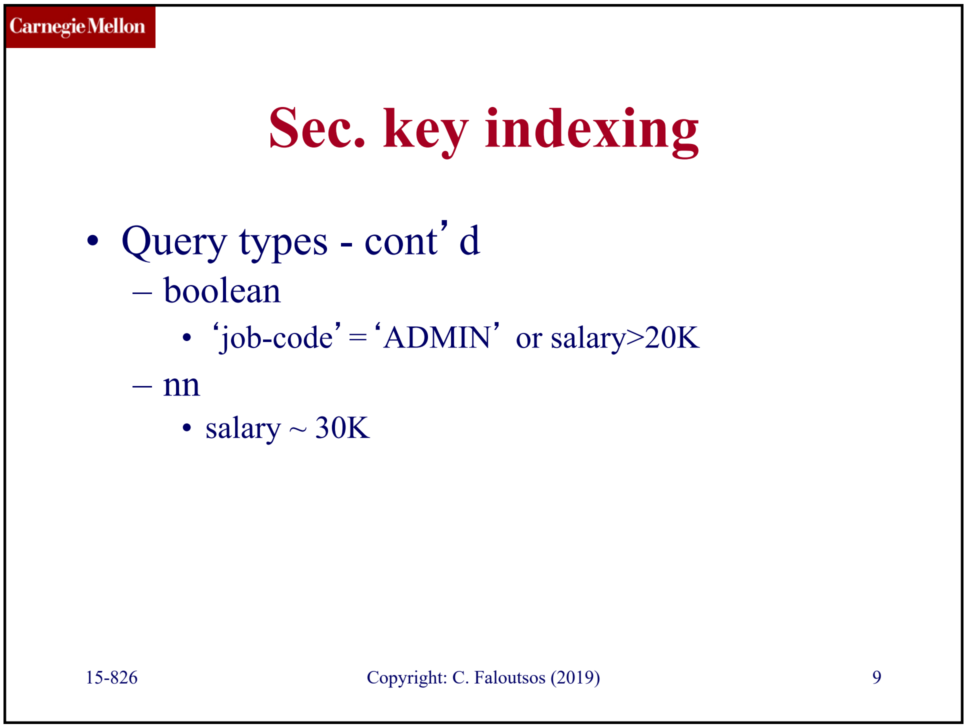
Sec. key indexing

- attributes w/ duplicates (eg., EMPLOYEES, with ‘job-code’, ‘salary’, ‘dept’)
- Query types:

Carnegie Mellon

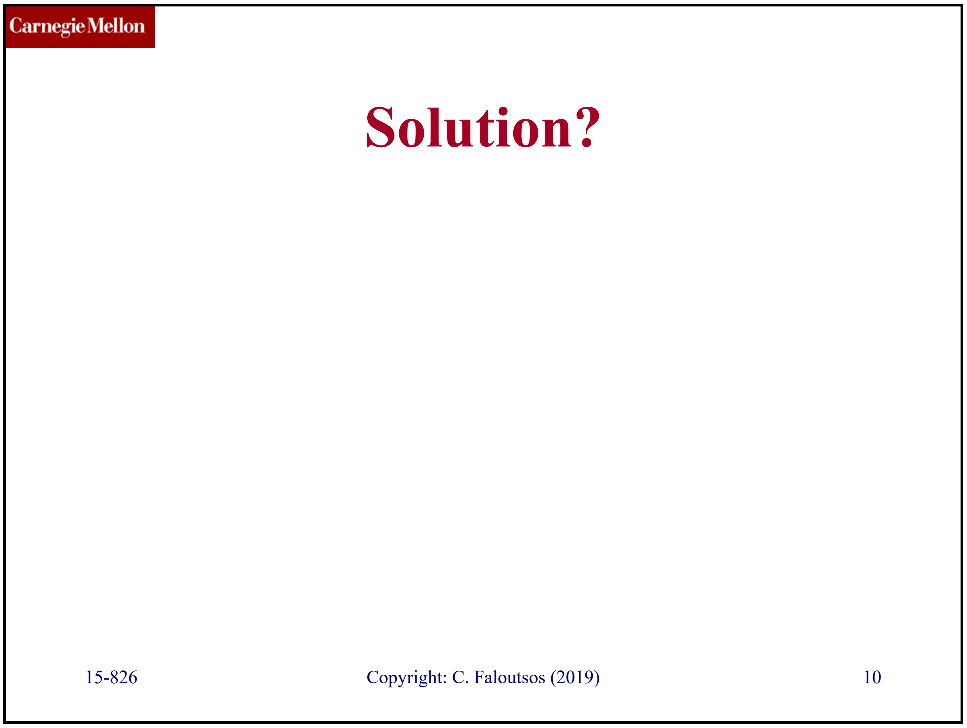
Sec. key indexing

- attributes w/ duplicates (eg., EMPLOYEES, with ‘job-code’, ‘salary’, ‘dept’)
- Query types:
  - exact match
  - partial match
    - ‘job-code’ = ‘PGM’ and ‘dept’ = ‘R&D’
  - range queries
    - ‘job-code’ = ‘ADMIN’ and salary < 50K

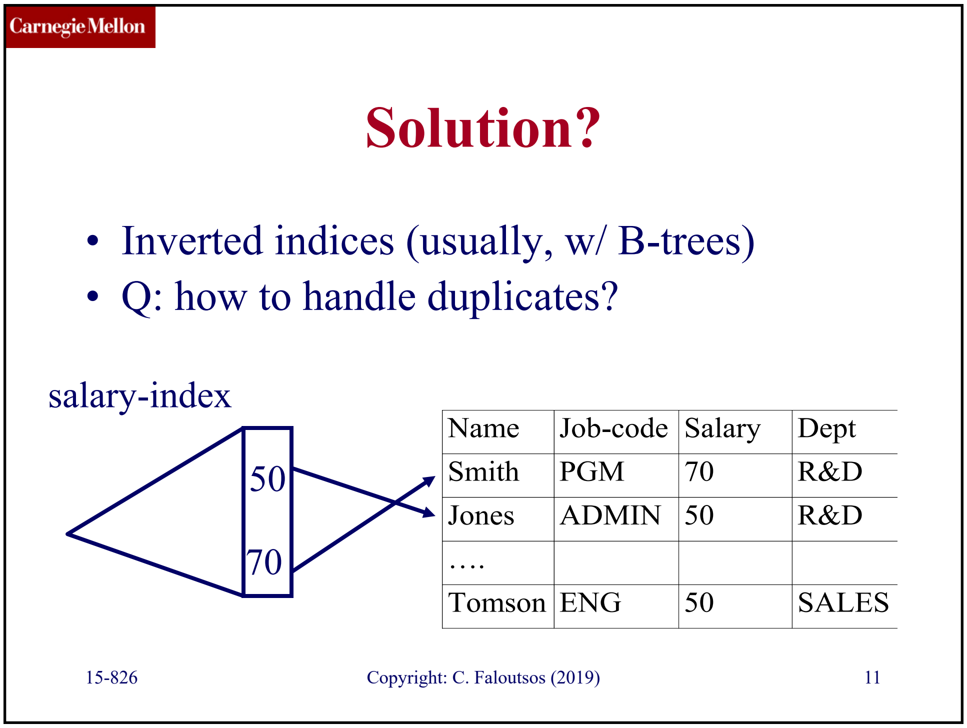


Sec. key indexing

- Query types - cont’ d
  - boolean
    - ‘job-code’ = ‘ADMIN’ or salary>20K
  - nn
    - salary ~ 30K



Solution?

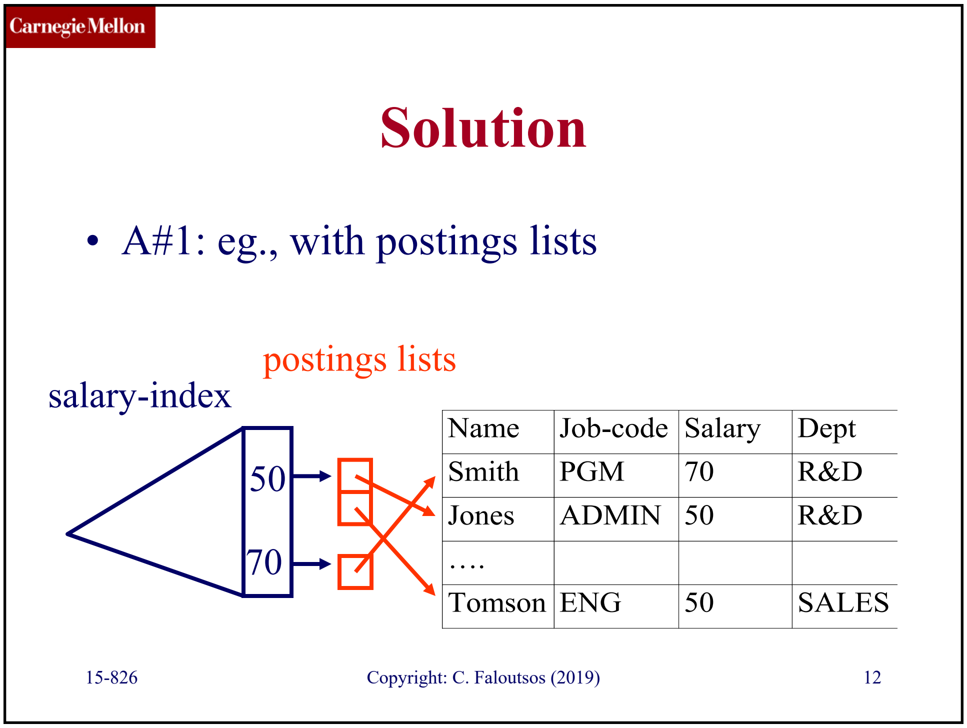


Solution?

- Inverted indices (usually, w/ B-trees)
- Q: how to handle duplicates?

salary-index
50
70

Name	Job-code	Salary	Dept
Smith	PGM	70	R&D
Jones	ADMIN	50	R&D
….			
Tomson	ENG	50	SALES

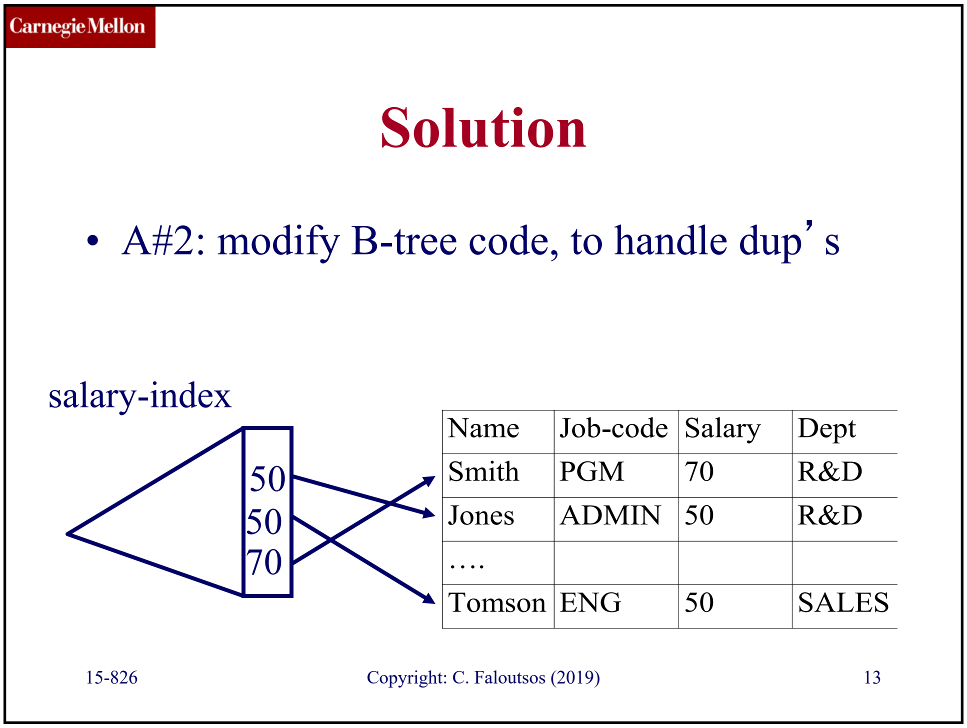


Solution

- A#1: eg., with postings lists

postings lists
salary-index
50
70

Name	Job-code	Salary	Dept
Smith	PGM	70	R&D
Jones	ADMIN	50	R&D
….			
Tomson	ENG	50	SALES



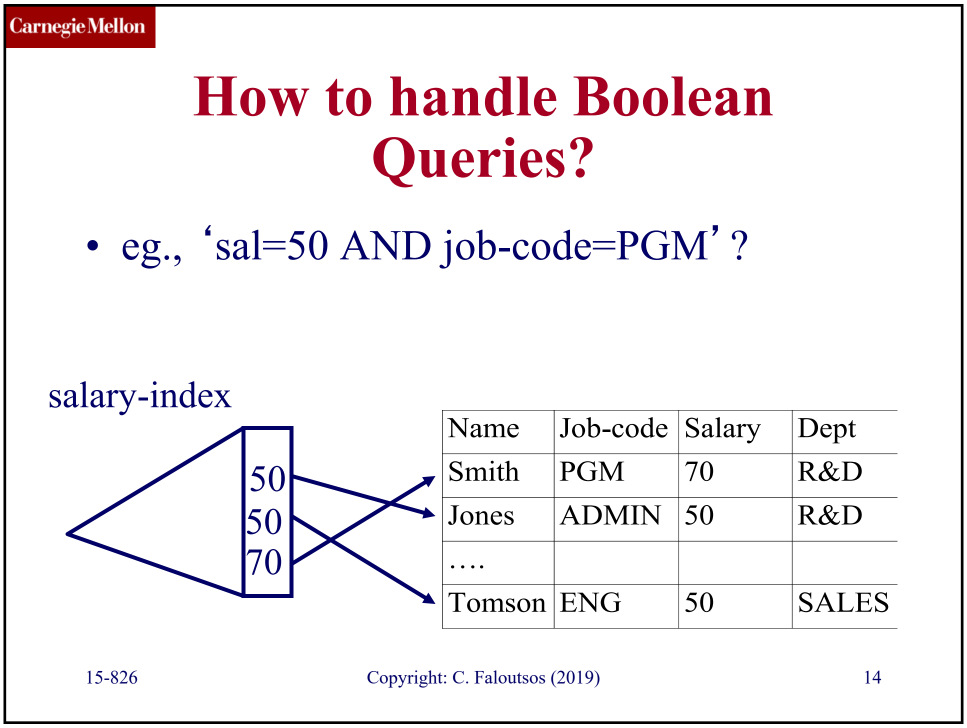
Solution

- A#2: modify B-tree code, to handle dup’s

salary-index

50
50
70

Name	Job-code	Salary	Dept
Smith	PGM	70	R&D
Jones	ADMIN	50	R&D
….			
Tomson	ENG	50	SALES



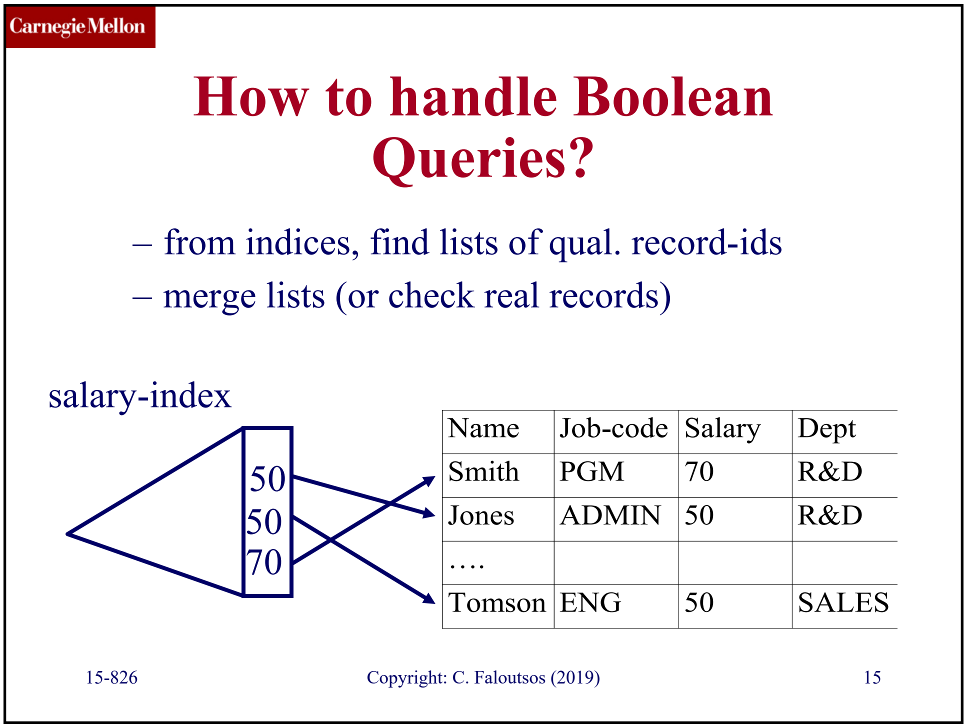
How to handle Boolean Queries?

- eg., ‘sal=50 AND job-code=PGM’?

salary-index

50
50
70

Name	Job-code	Salary	Dept
Smith	PGM	70	R&D
Jones	ADMIN	50	R&D
….			
Tomson	ENG	50	SALES



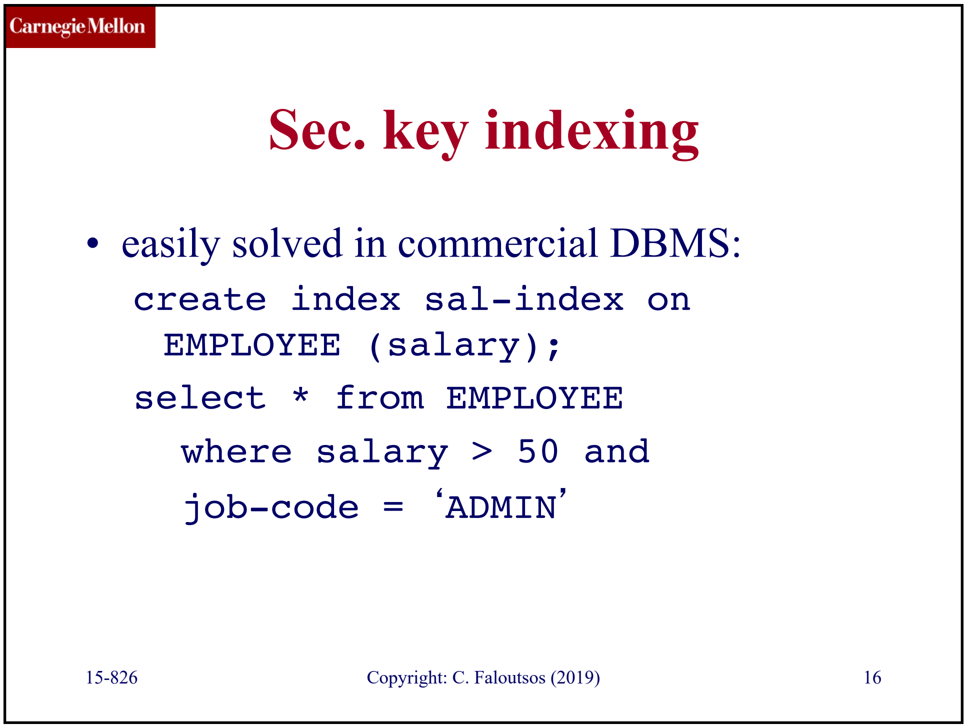
Carnegie Mellon

How to handle Boolean Queries?

- from indices, find lists of qual. record-ids
- merge lists (or check real records)

salary-index
50
50
70

Name	Job-code	Salary	Dept
Smith	PGM	70	R&D
Jones	ADMIN	50	R&D
….			
Tomson	ENG	50	SALES

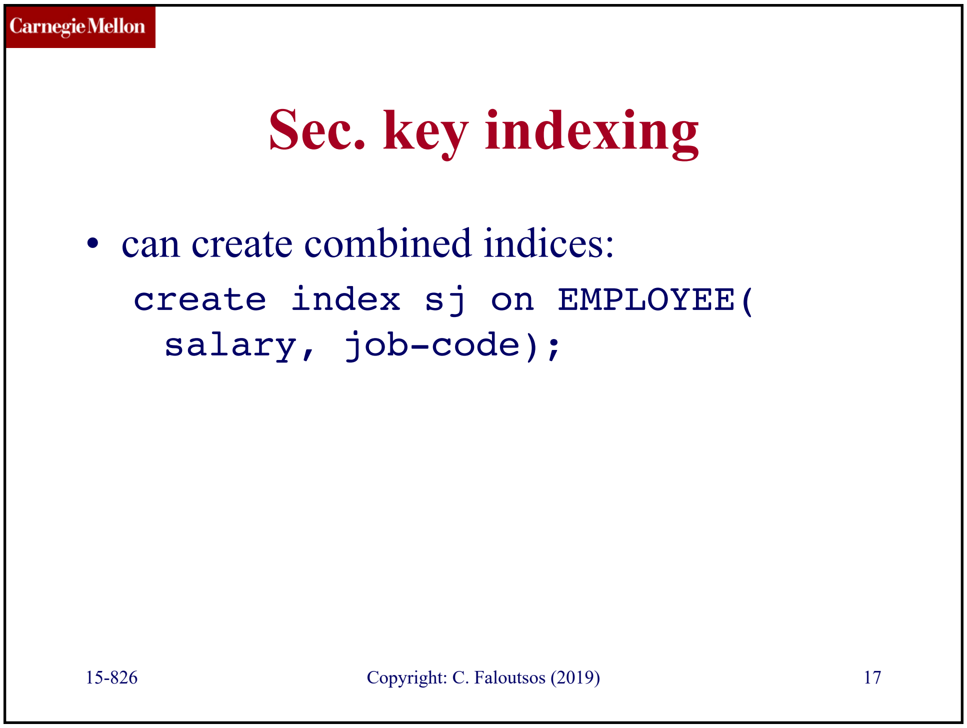


Carnegie Mellon

Sec. key indexing

- easily solved in commercial DBMS:

```
create index sal-index on
  EMPLOYEE (salary);
select * from EMPLOYEE
   where salary > 50 and
   job-code = 'ADMIN'
```

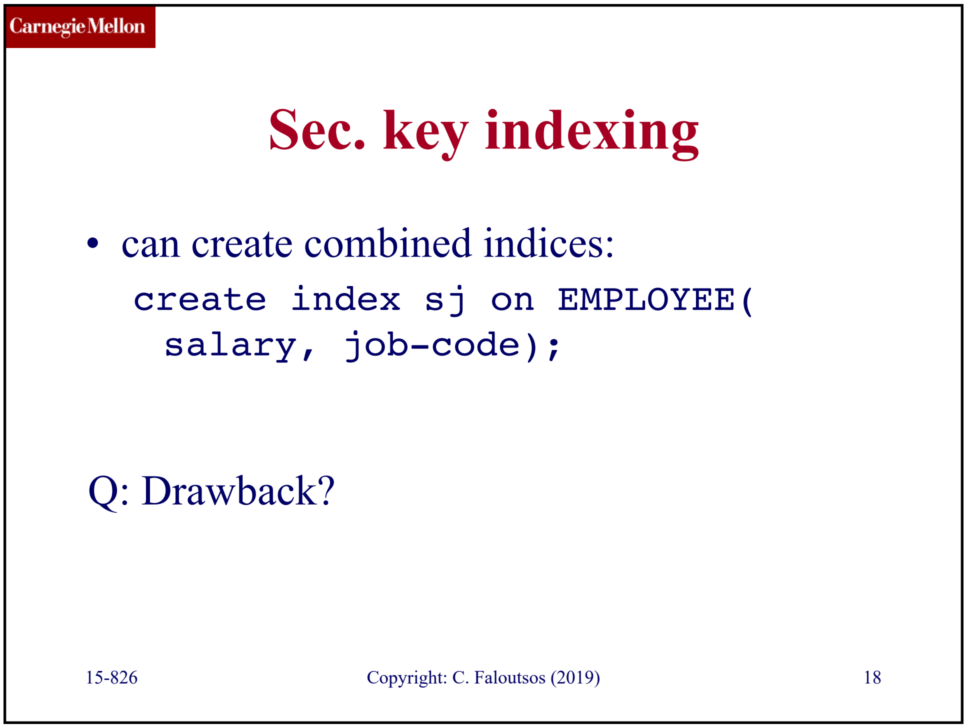


Carnegie Mellon

Sec. key indexing

- can create combined indices:

```
create index sj on EMPLOYEE(
  salary, job-code);
```



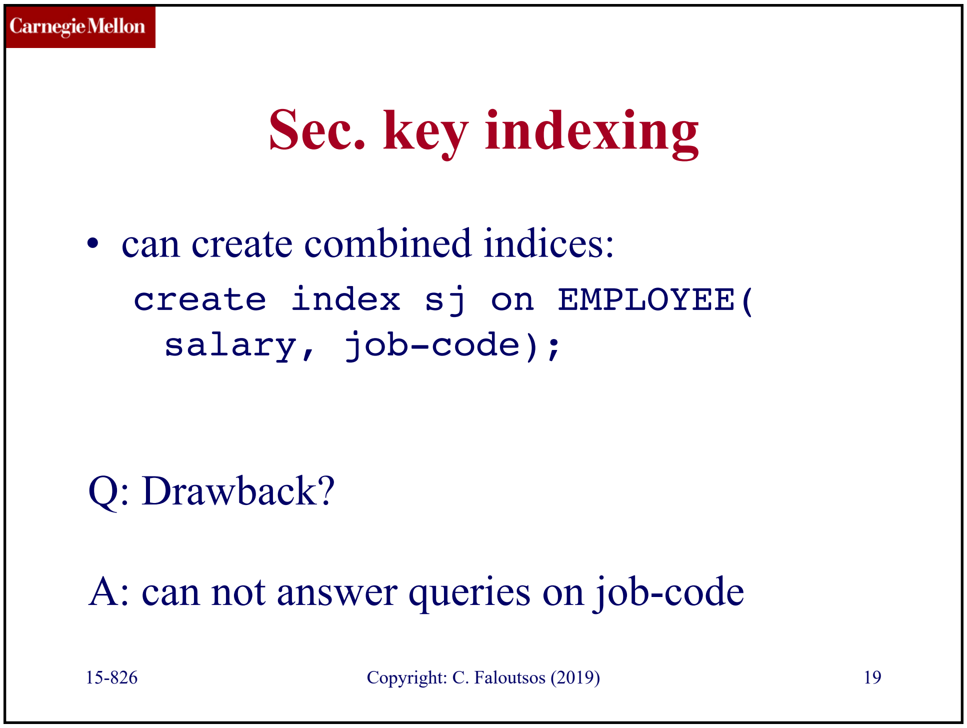
Carnegie Mellon

Sec. key indexing

- can create combined indices:

```
create index sj on EMPLOYEE(
  salary, job-code);
```

Q: Drawback?



Carnegie Mellon

Sec. key indexing

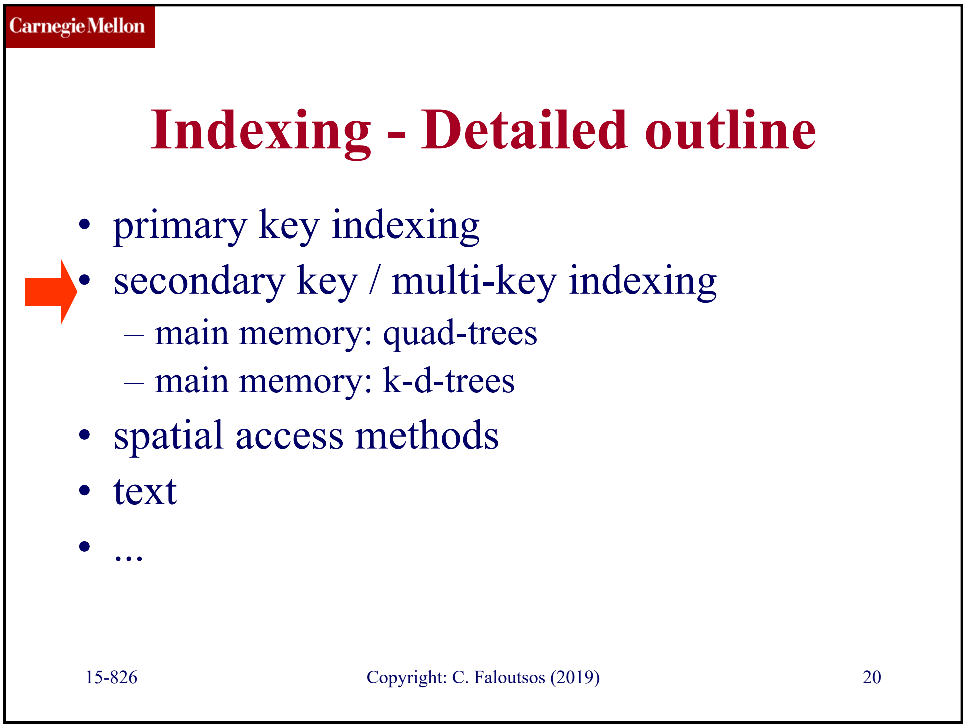
- can create combined indices:

```
create index sj on EMPLOYEE(
  salary, job-code);
```

Q: Drawback?

A: can not answer queries on job-code

15-826 Copyright: C. Faloutsos (2019) 19

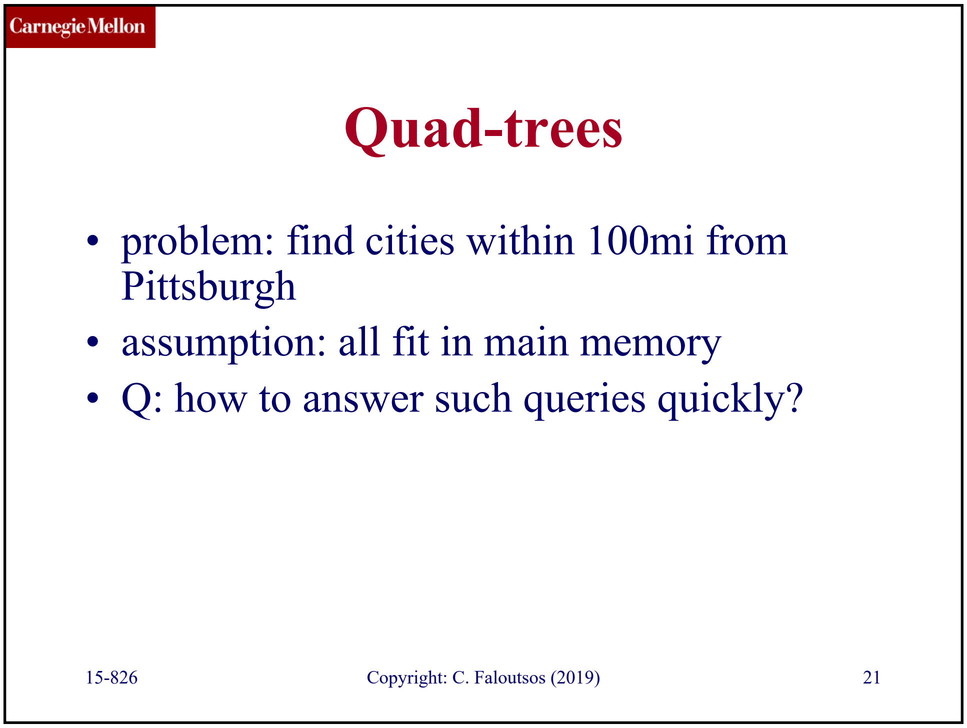


Carnegie Mellon

Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
  - main memory: quad-trees
  - main memory: k-d-trees
- spatial access methods
- text
- ...

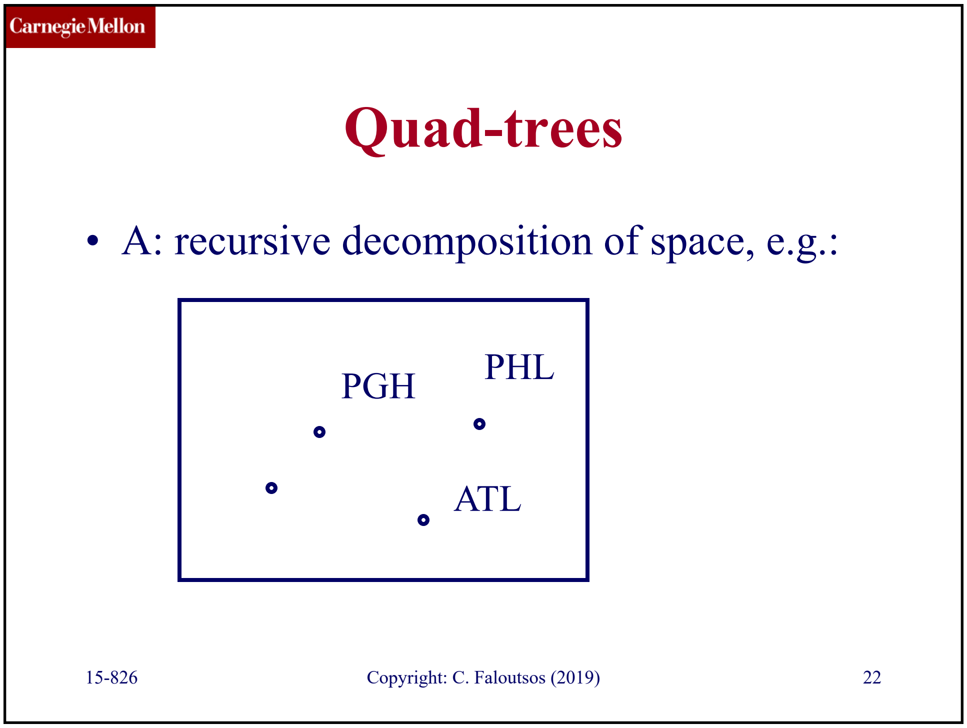
15-826 Copyright: C. Faloutsos (2019) 20



Carnegie Mellon

Quad-trees

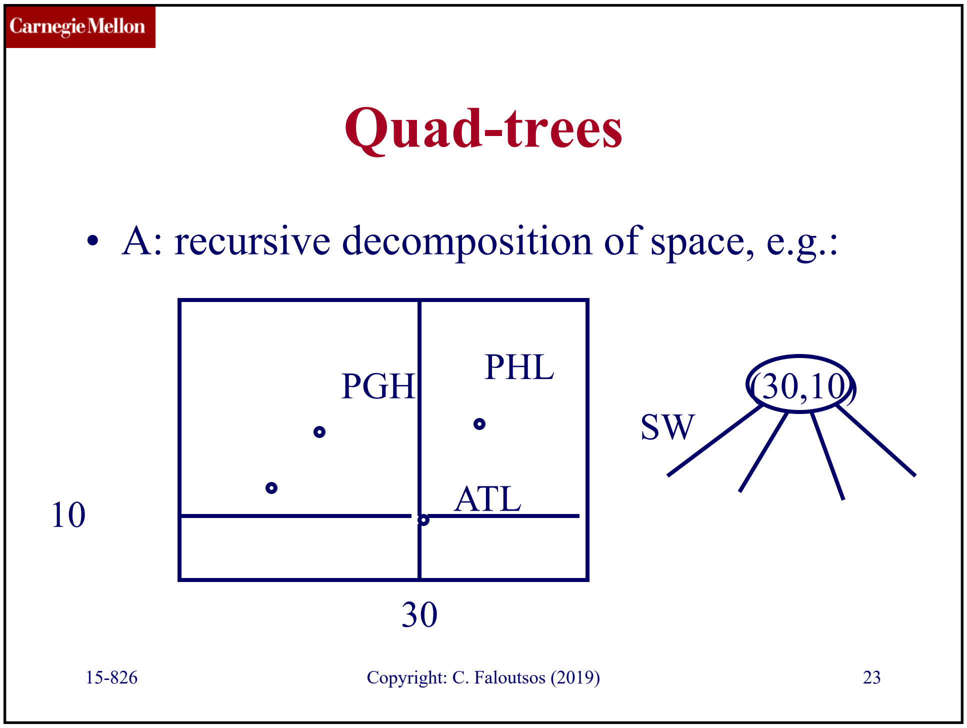
- problem: find cities within 100mi from Pittsburgh
- assumption: all fit in main memory
- Q: how to answer such queries quickly?



Carnegie Mellon

Quad-trees

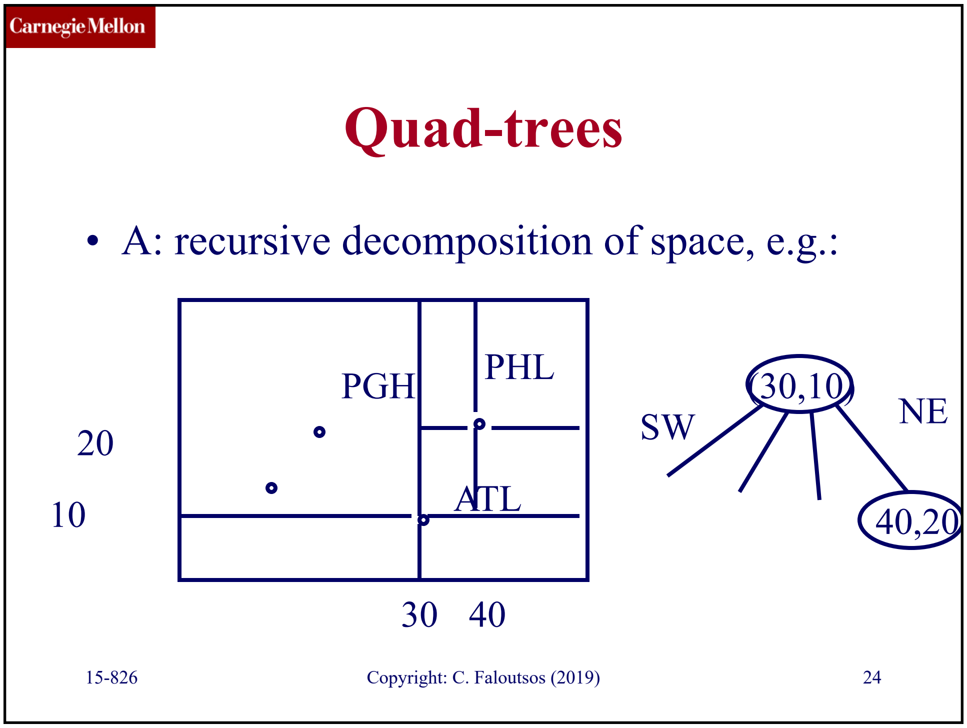
- A: recursive decomposition of space, e.g.:



Quad-trees

- A: recursive decomposition of space, e.g.:

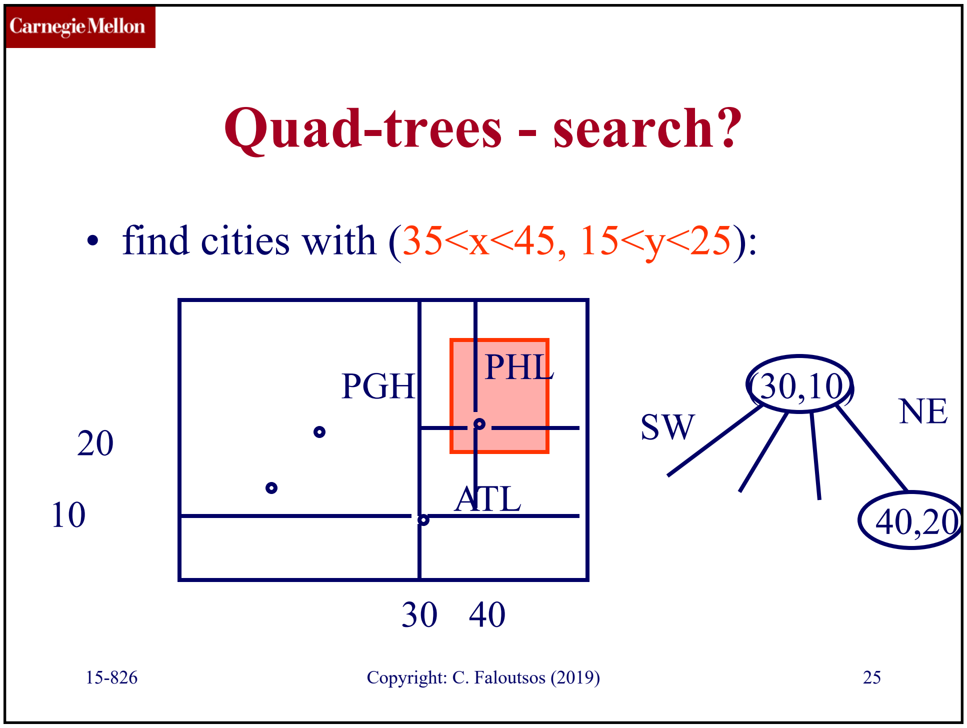
15-826 Copyright: C. Faloutsos (2019) 23



Quad-trees

- A: recursive decomposition of space, e.g.:

15-826 Copyright: C. Faloutsos (2019) 24

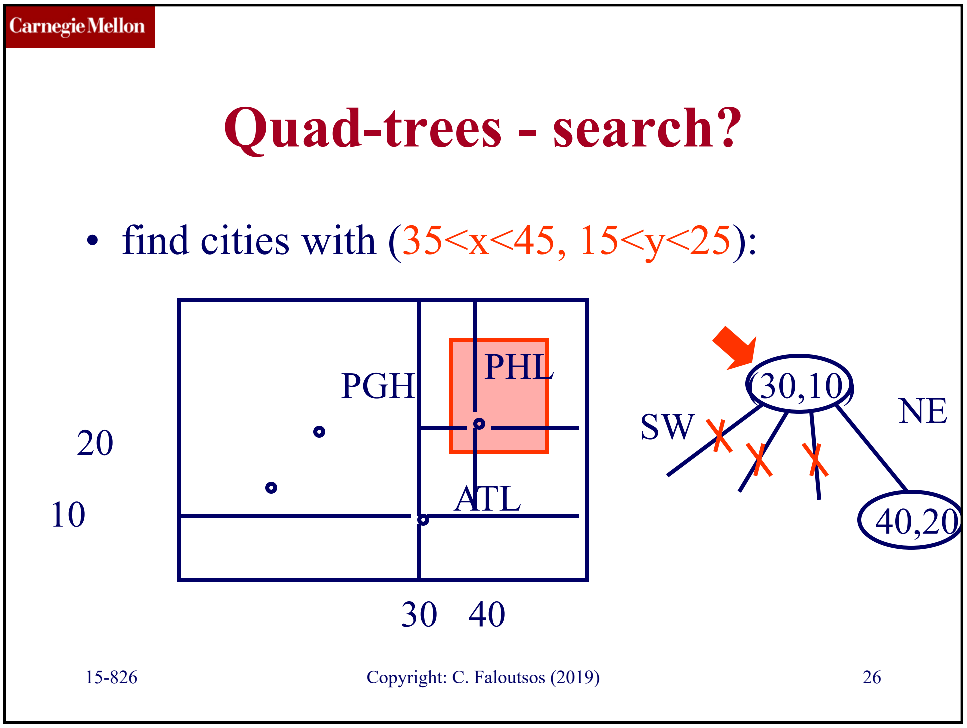


Carnegie Mellon

Quad-trees - search?

- find cities with (35<x<45, 15<y<25):

15-826 Copyright: C. Faloutsos (2019) 25

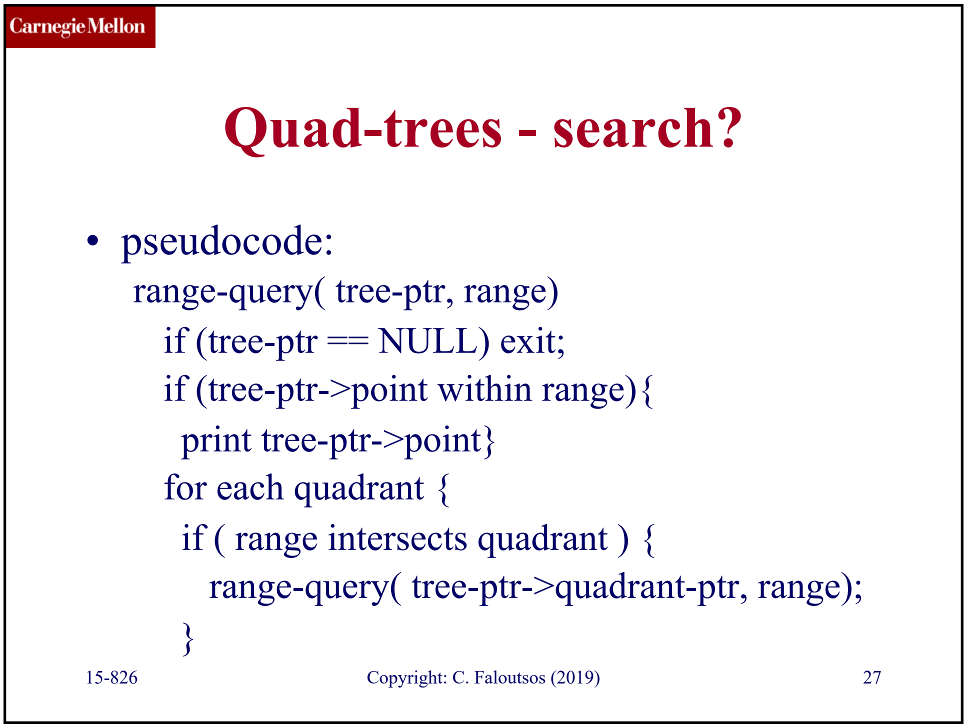


Carnegie Mellon

Quad-trees - search?

- find cities with (35<x<45, 15<y<25):

15-826 Copyright: C. Faloutsos (2019) 26

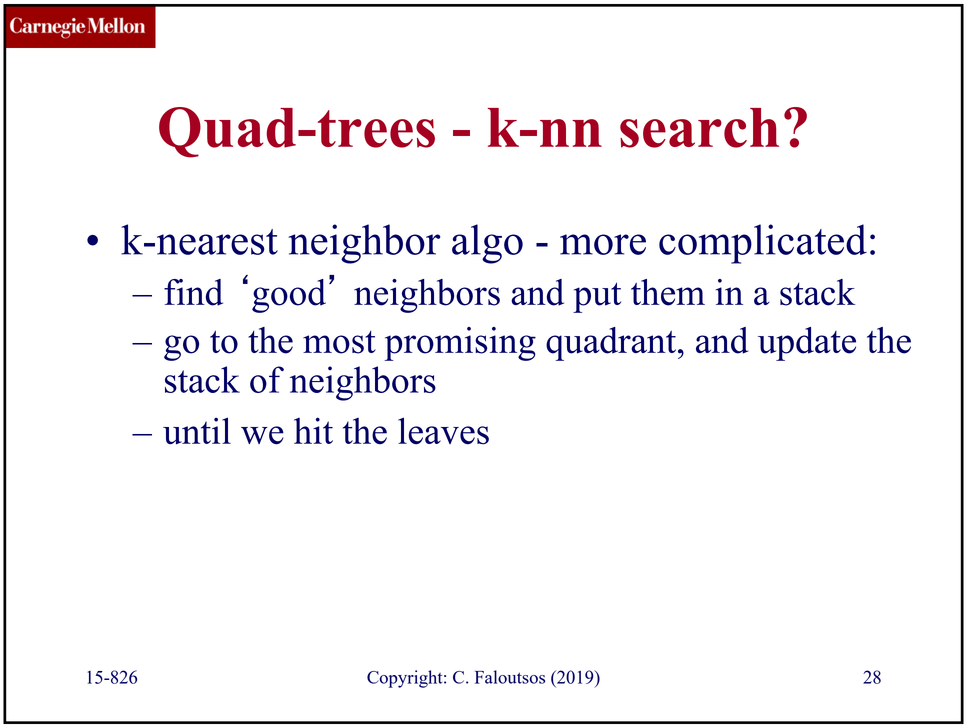


Carnegie Mellon

Quad-trees - search?

- pseudocode:

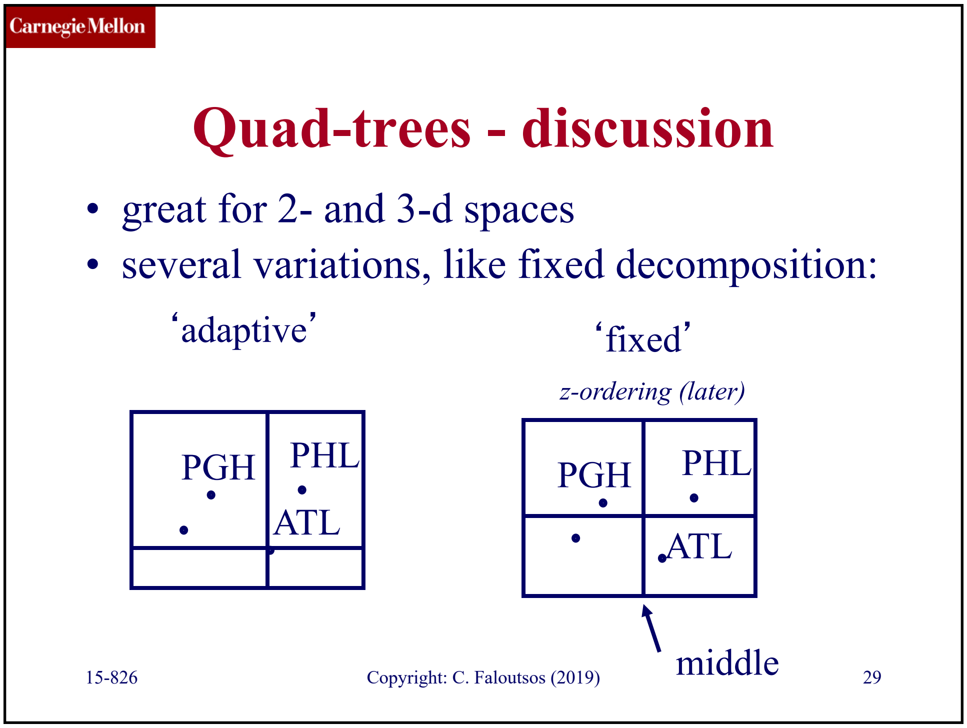
```
range-query( tree-ptr, range)
   if (tree-ptr == NULL) exit;
   if (tree-ptr->point within range){
     print tree-ptr->point}
   for each quadrant {
     if ( range intersects quadrant ) {
        range-query( tree-ptr->quadrant-ptr, range);
     }
```



Carnegie Mellon

Quad-trees - k-nn search?

- k-nearest neighbor algo - more complicated:
  - find ‘good’ neighbors and put them in a stack
  - go to the most promising quadrant, and update the stack of neighbors
  - until we hit the leaves

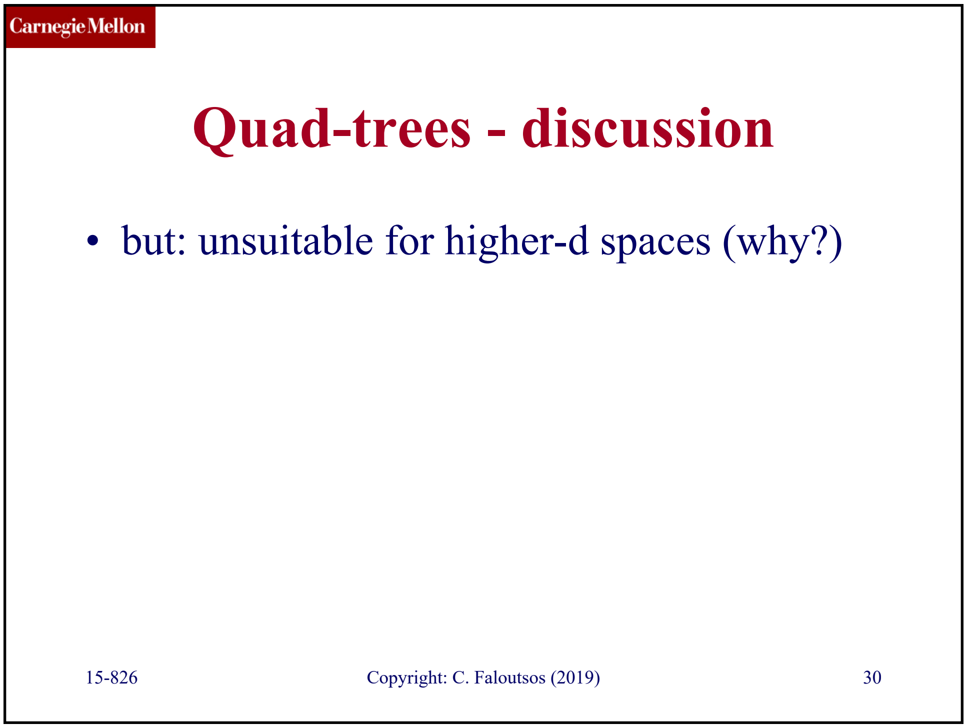


Carnegie Mellon

Quad-trees - discussion

- great for 2- and 3-d spaces
- several variations, like fixed decomposition:

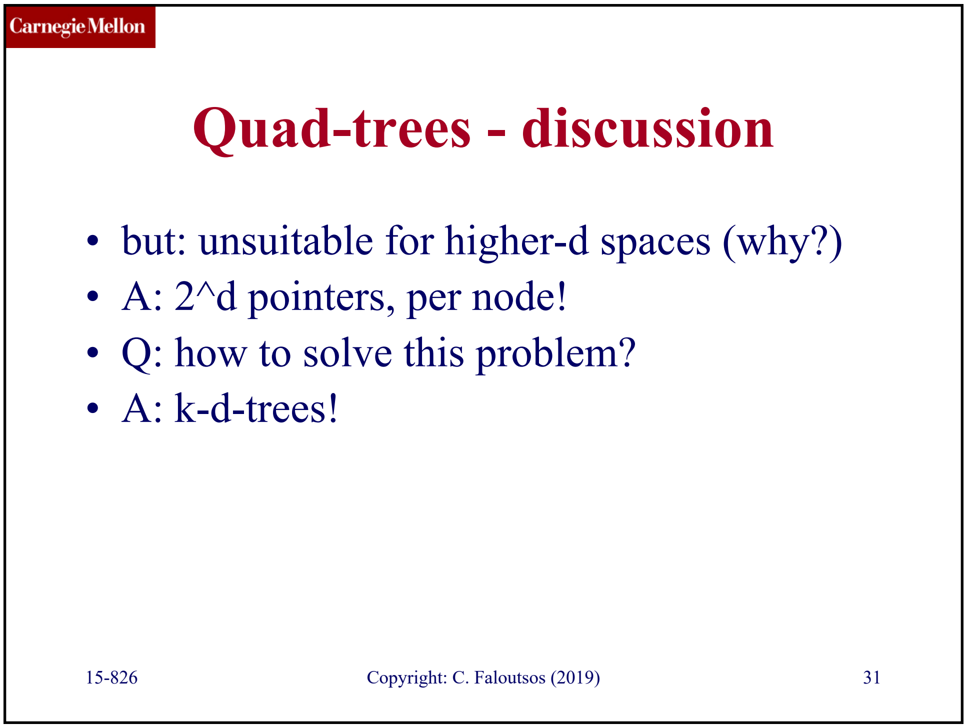
'adaptive'
PGH
PHL
ATL
'fixed'
z-ordering (later)
PGH
PHL
ATL
middle



Carnegie Mellon

Quad-trees - discussion

- but: unsuitable for higher-d spaces (why?)

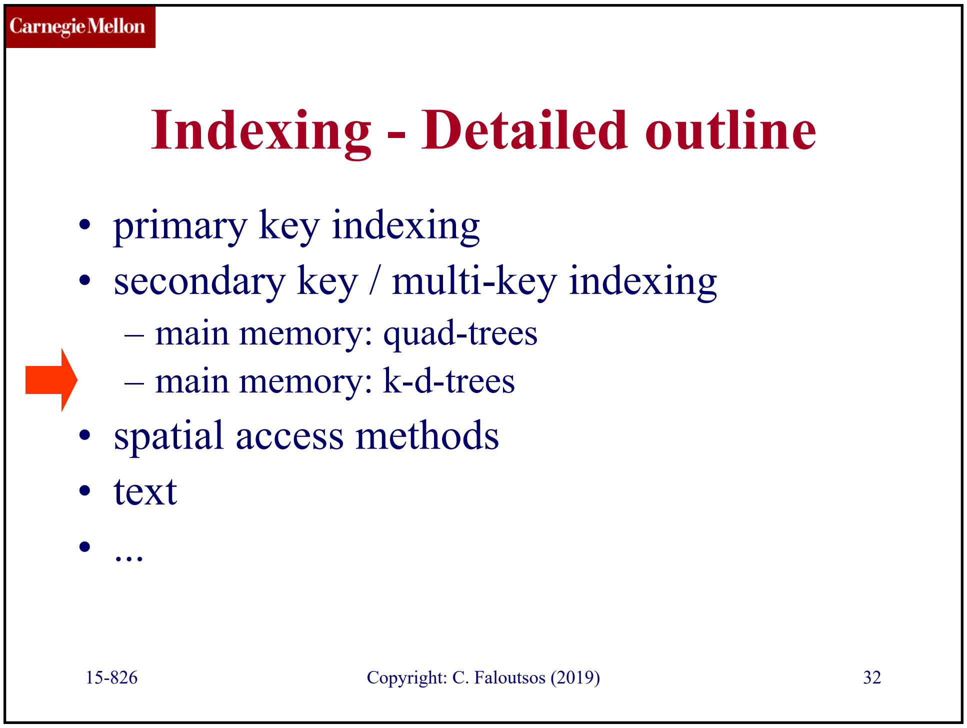


Carnegie Mellon

Quad-trees - discussion

- but: unsuitable for higher-d spaces (why?)
- A: 2^d pointers, per node!
- Q: how to solve this problem?
- A: k-d-trees!

15-826 Copyright: C. Faloutsos (2019) 31

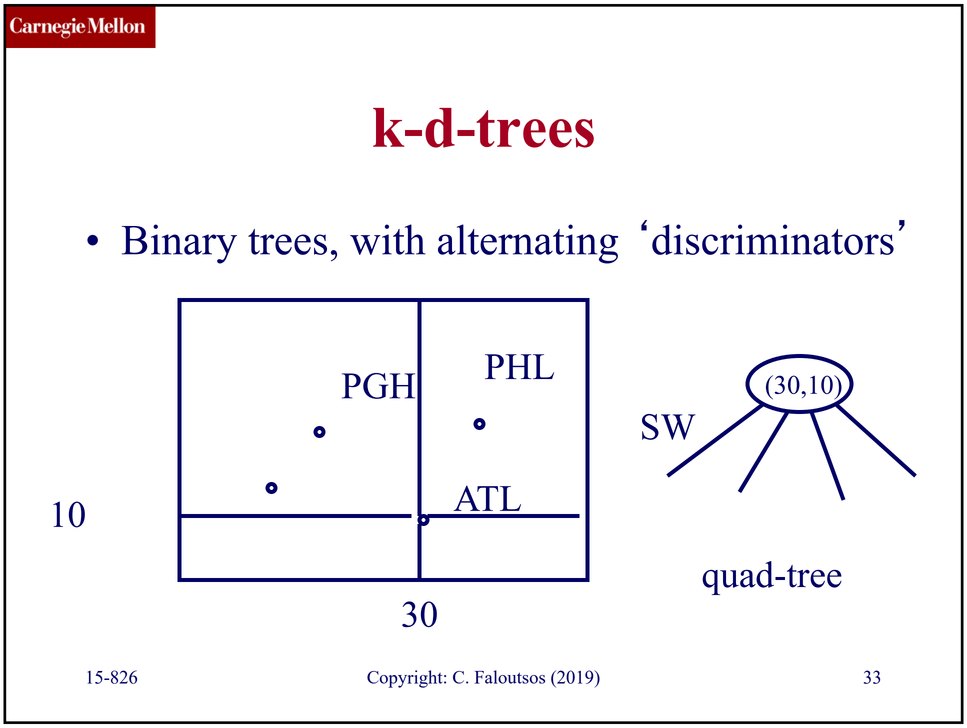


Carnegie Mellon

Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
  - main memory: quad-trees
  - ➡ main memory: k-d-trees
- spatial access methods
- text
- ...

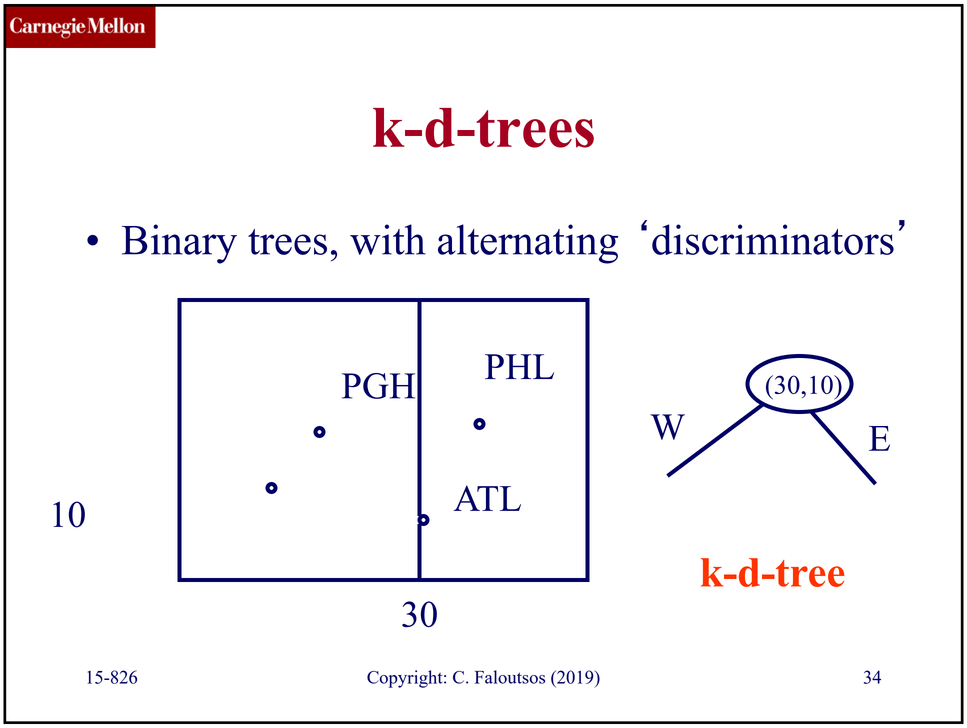
15-826 Copyright: C. Faloutsos (2019) 32



Carnegie Mellon

k-d-trees

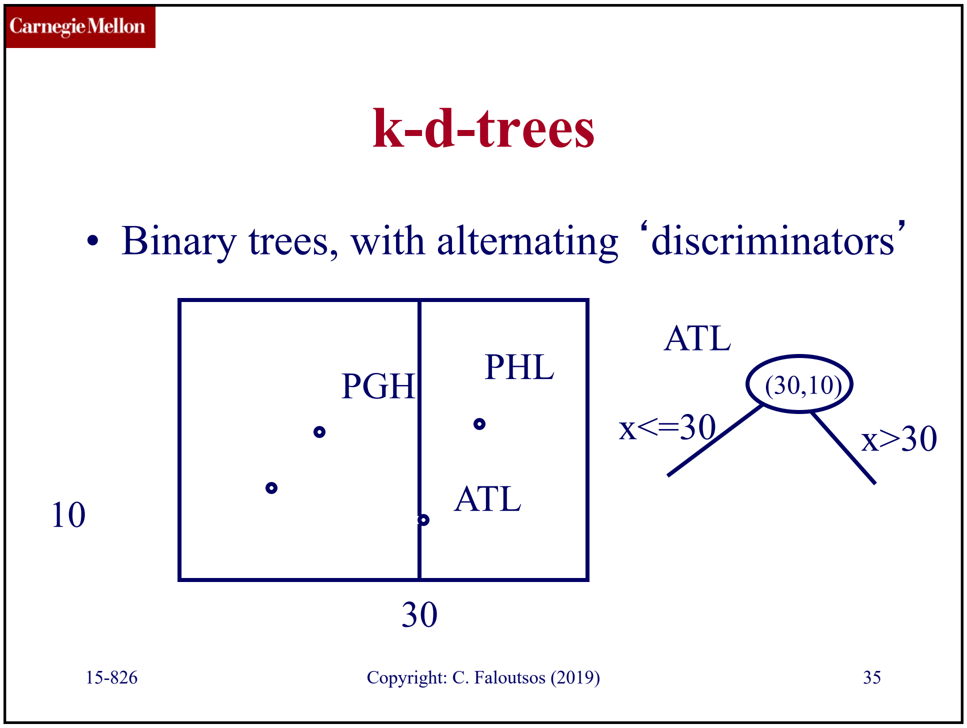
- Binary trees, with alternating ‘discriminators’



Carnegie Mellon

k-d-trees

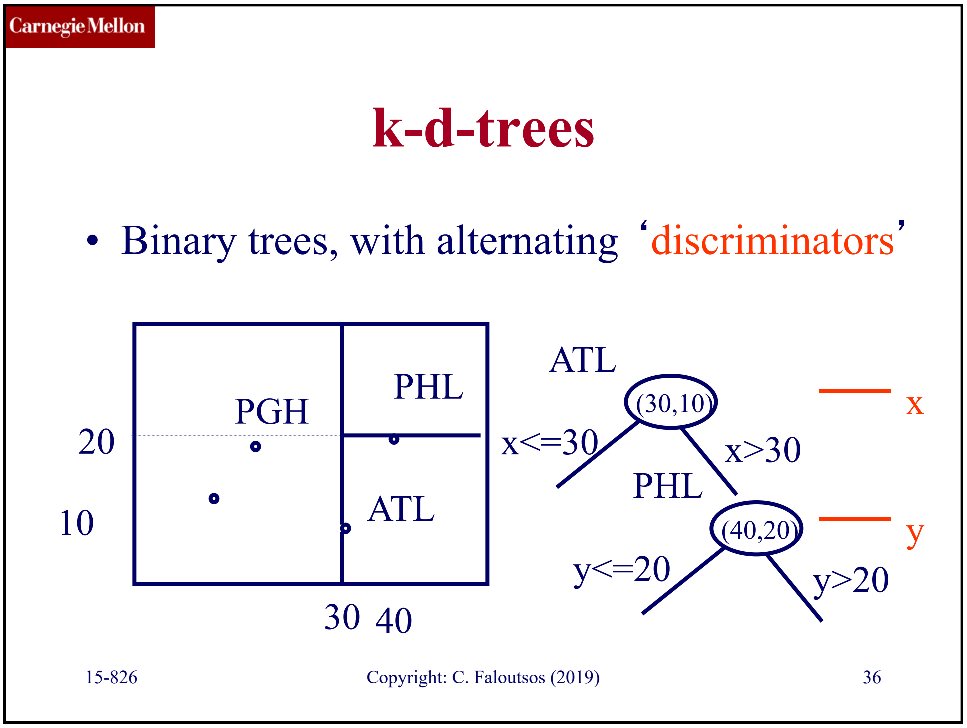
- Binary trees, with alternating ‘discriminators’



Carnegie Mellon

k-d-trees

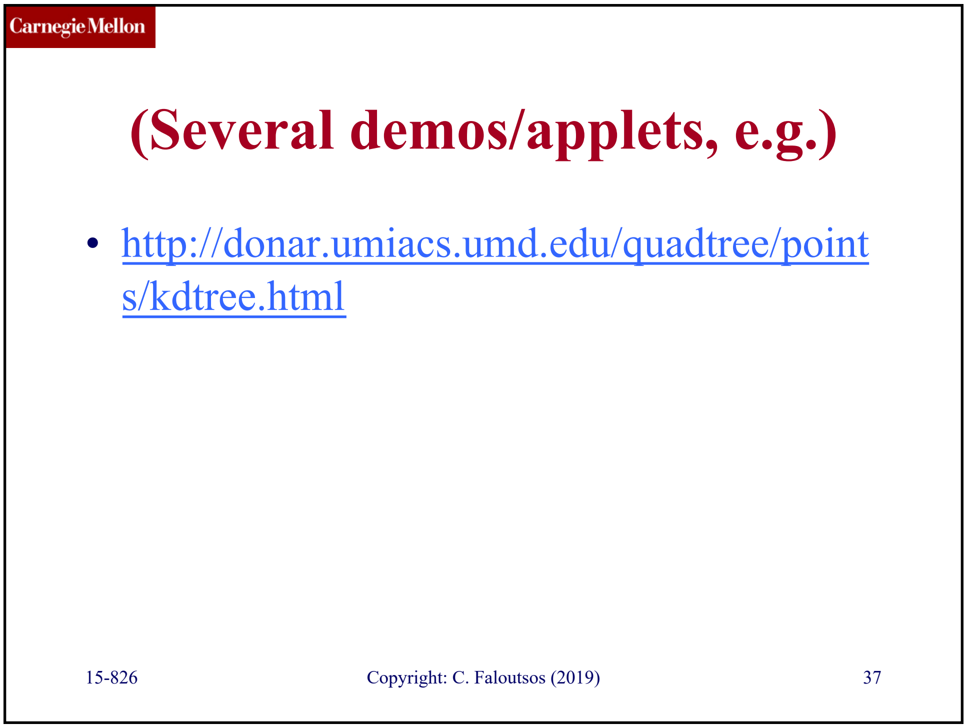
- Binary trees, with alternating ‘discriminators’



Carnegie Mellon

k-d-trees

- Binary trees, with alternating ‘discriminators’

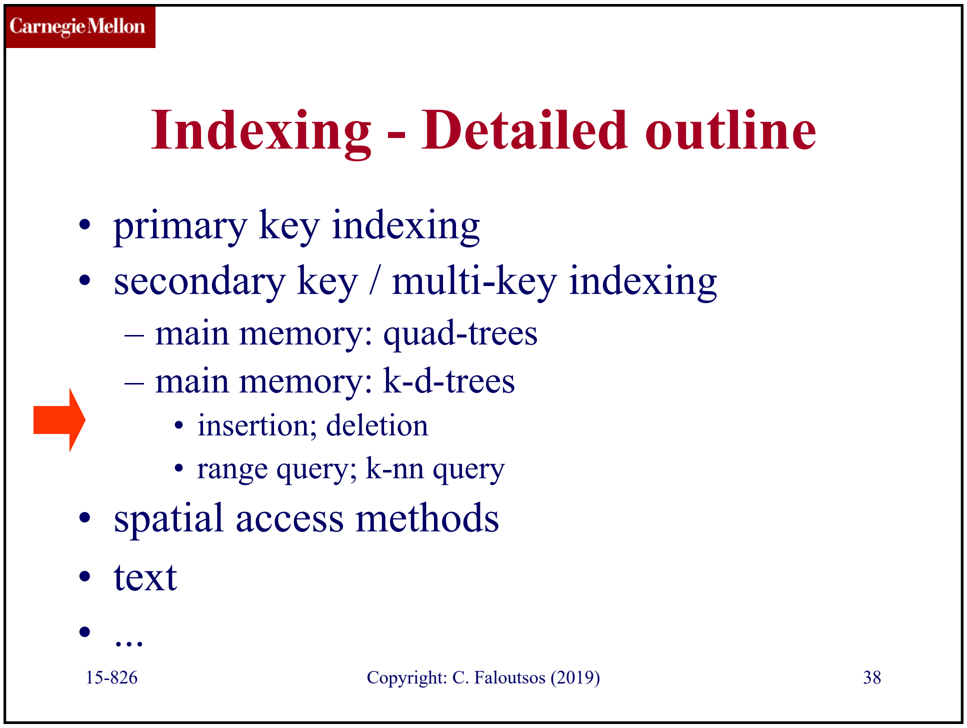


Carnegie Mellon

(Several demos/applets, e.g.)

- http://donar.umiacs.umd.edu/quadtree/points/kdtree.html

15-826 Copyright: C. Faloutsos (2019) 37

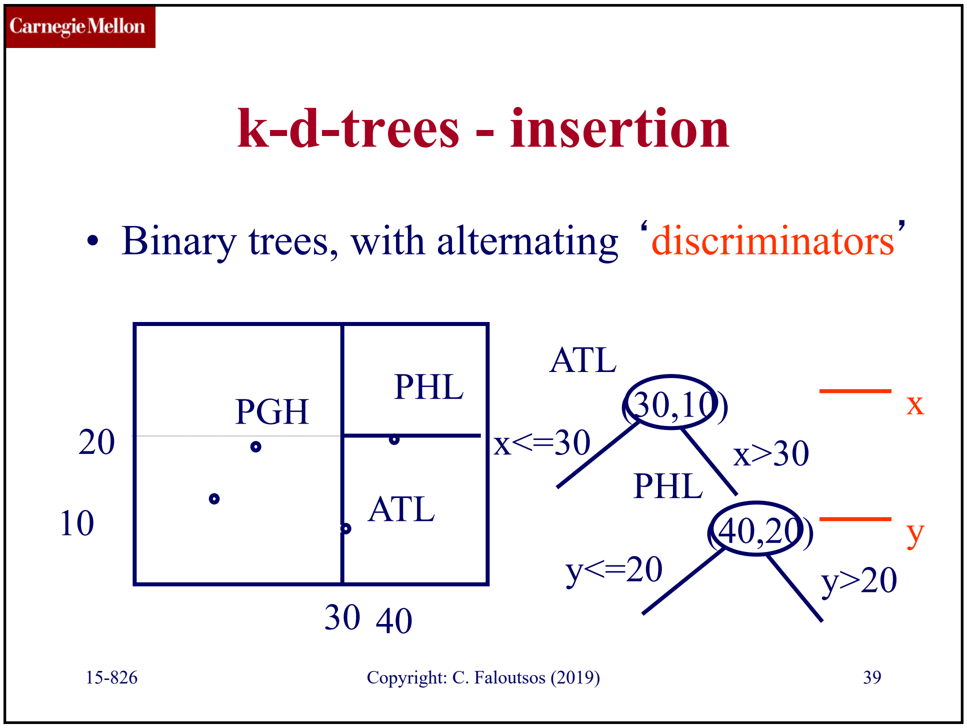


Carnegie Mellon

Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
  - main memory: quad-trees
  - main memory: k-d-trees
    - ➡ insertion; deletion
    - range query; k-nn query
- spatial access methods
- text
- ...

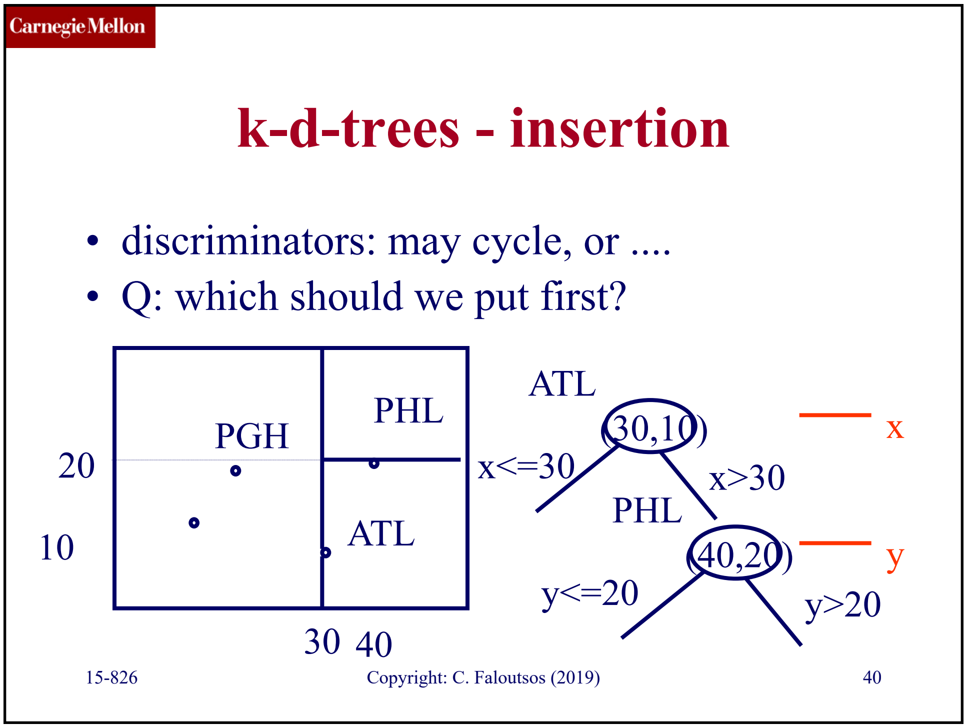
15-826 Copyright: C. Faloutsos (2019) 38



Carnegie Mellon

k-d-trees - insertion

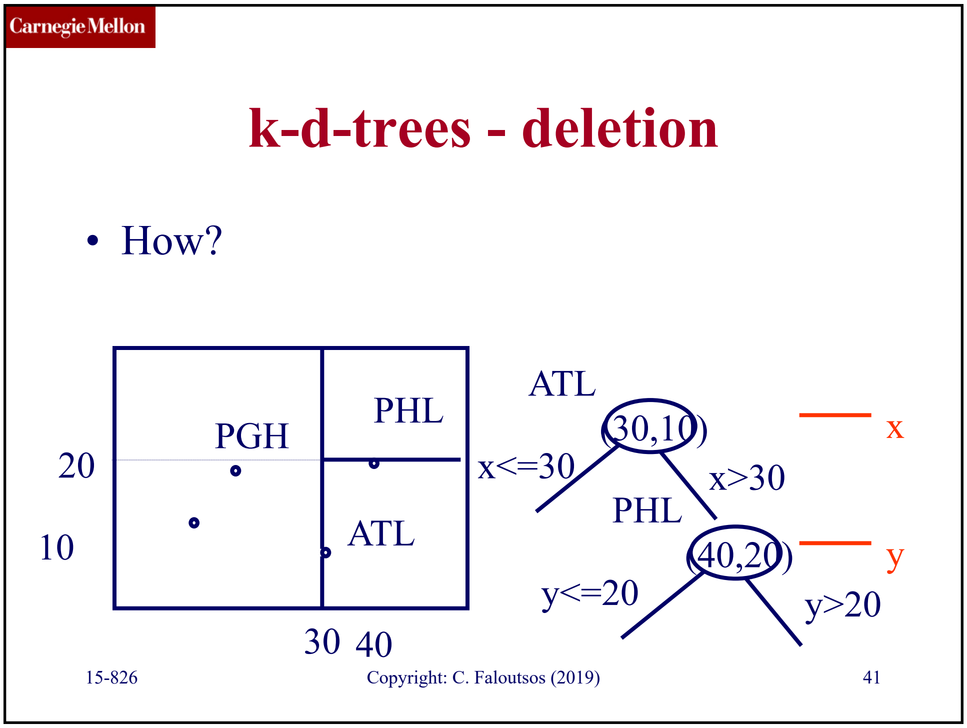
- Binary trees, with alternating ‘discriminators’



Carnegie Mellon

k-d-trees - insertion

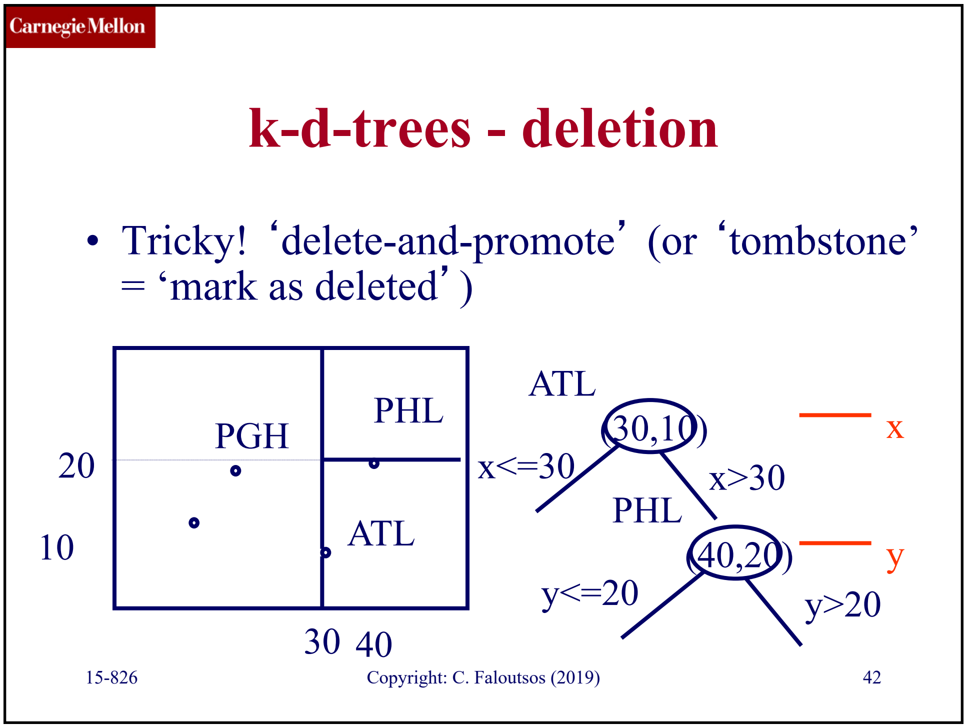
- discriminators: may cycle, or ....
- Q: which should we put first?



Carnegie Mellon

k-d-trees - deletion

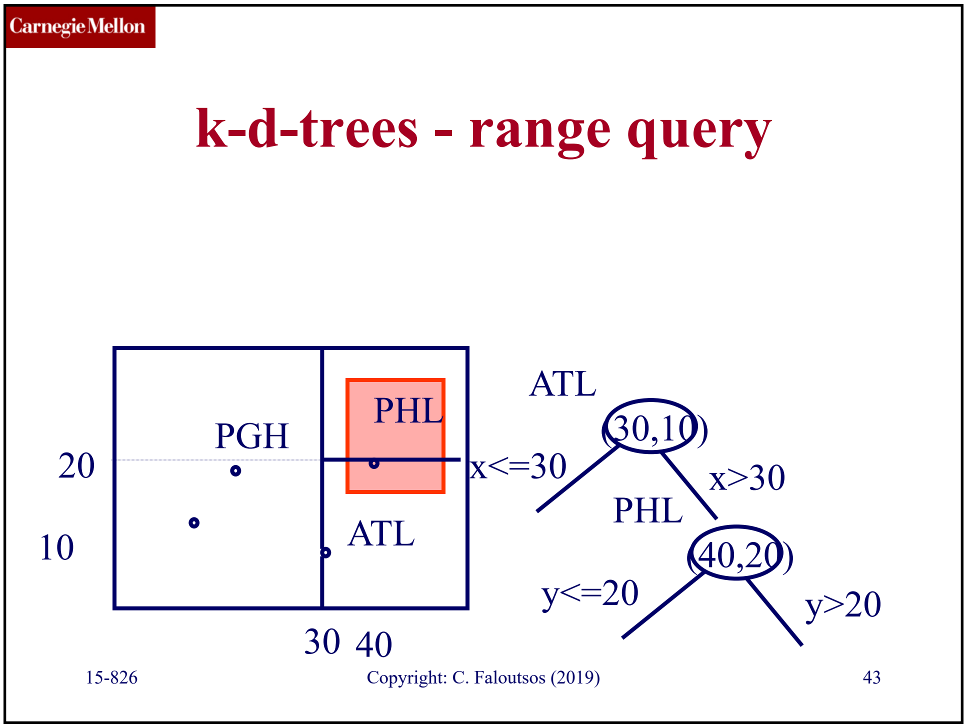
- How?



Carnegie Mellon

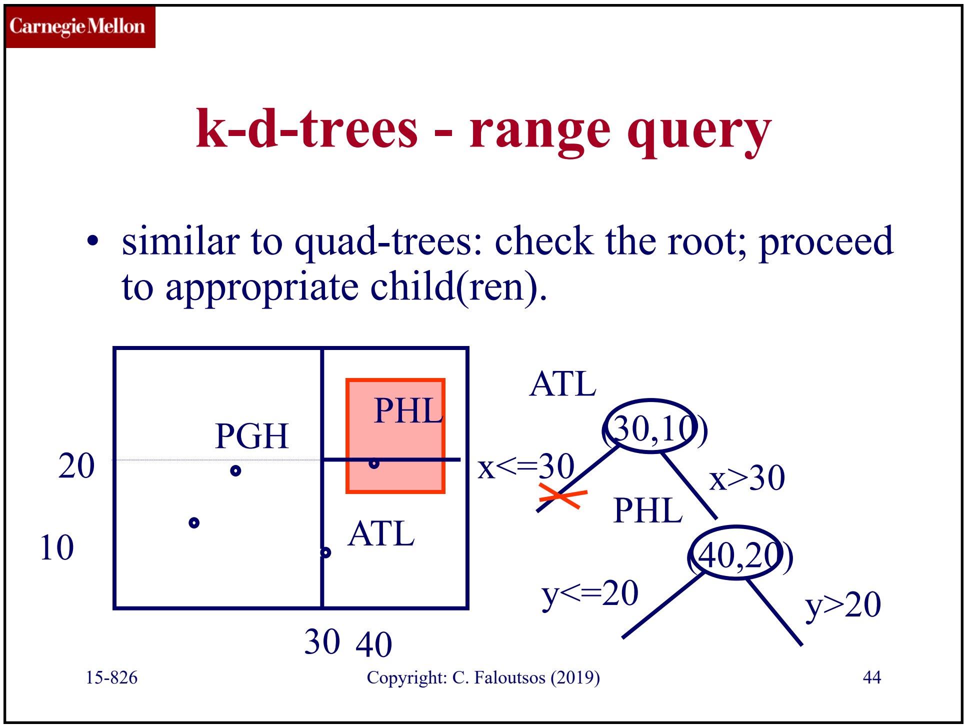
k-d-trees - deletion

- Tricky! ‘delete-and-promote’ (or ‘tombstone’ = ‘mark as deleted’)



Carnegie Mellon

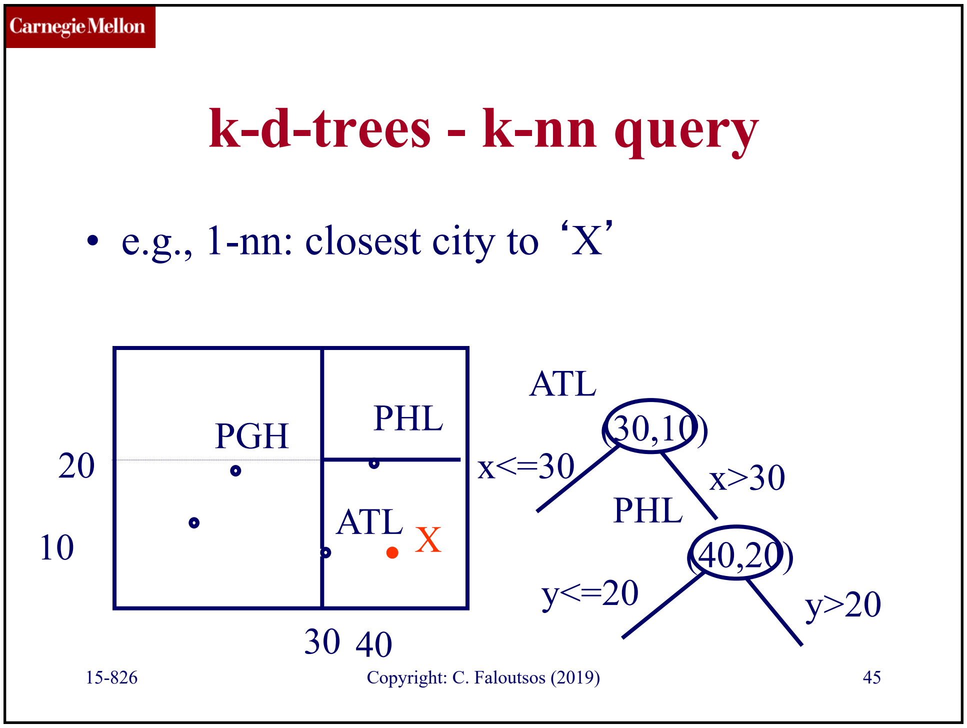
k-d-trees - range query



Carnegie Mellon

k-d-trees - range query

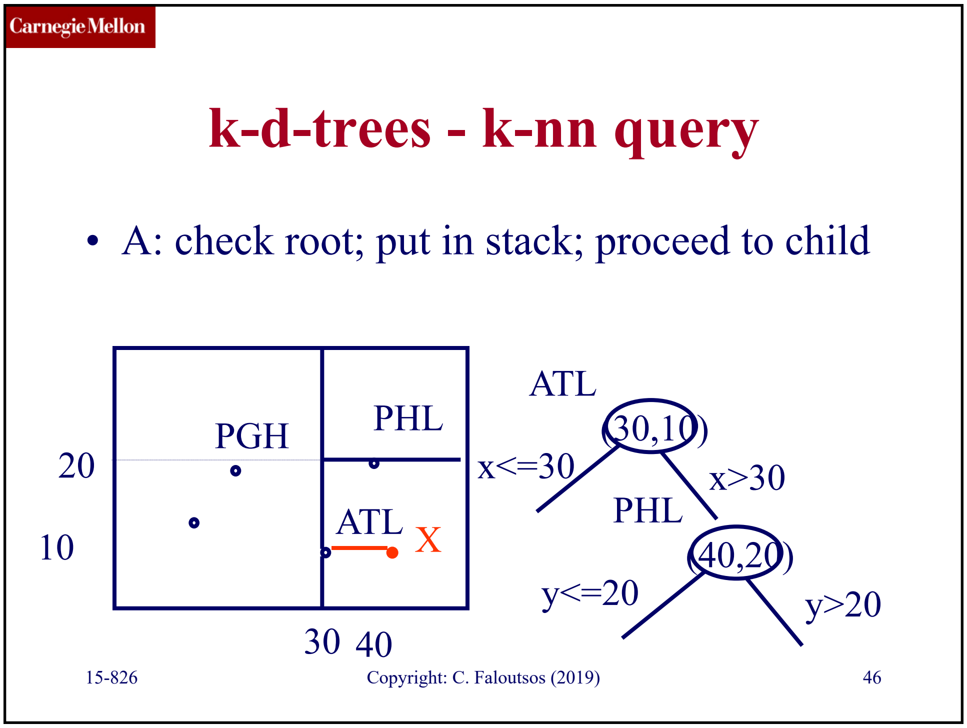
- similar to quad-trees: check the root; proceed to appropriate child(ren).



Carnegie Mellon

k-d-trees - k-nn query

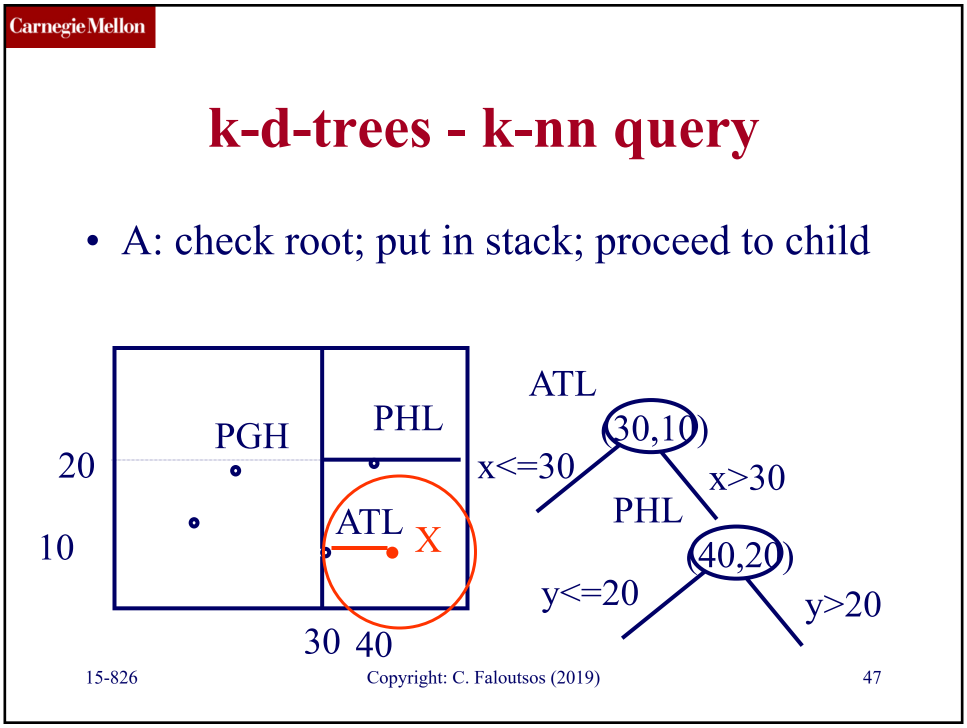
- e.g., 1-nn: closest city to ‘X’



Carnegie Mellon

k-d-trees - k-nn query

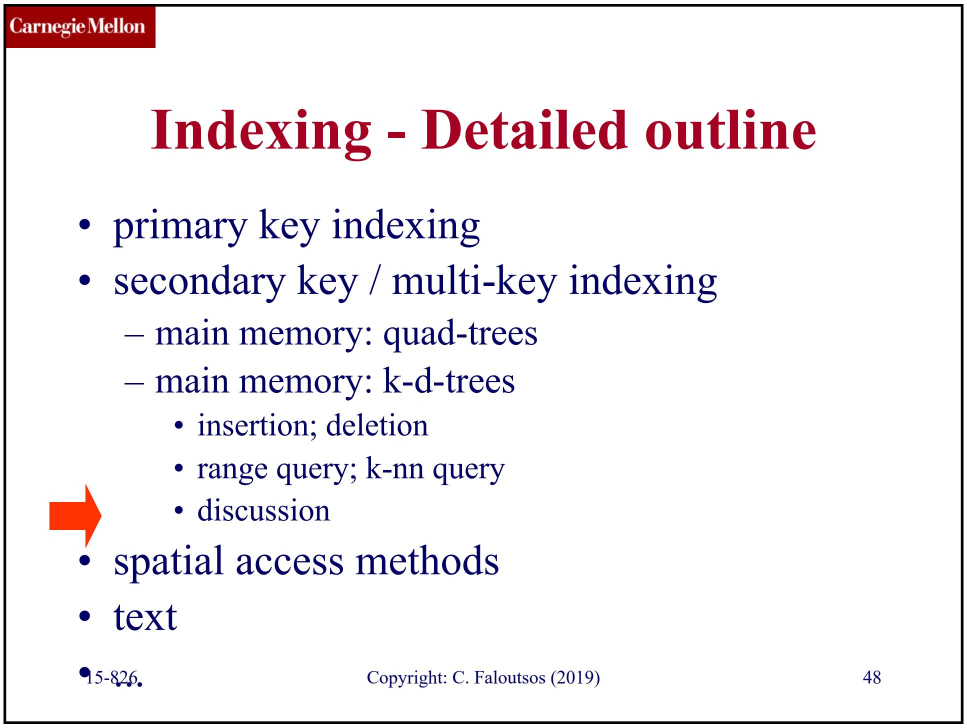
- A: check root; put in stack; proceed to child



Carnegie Mellon

k-d-trees - k-nn query

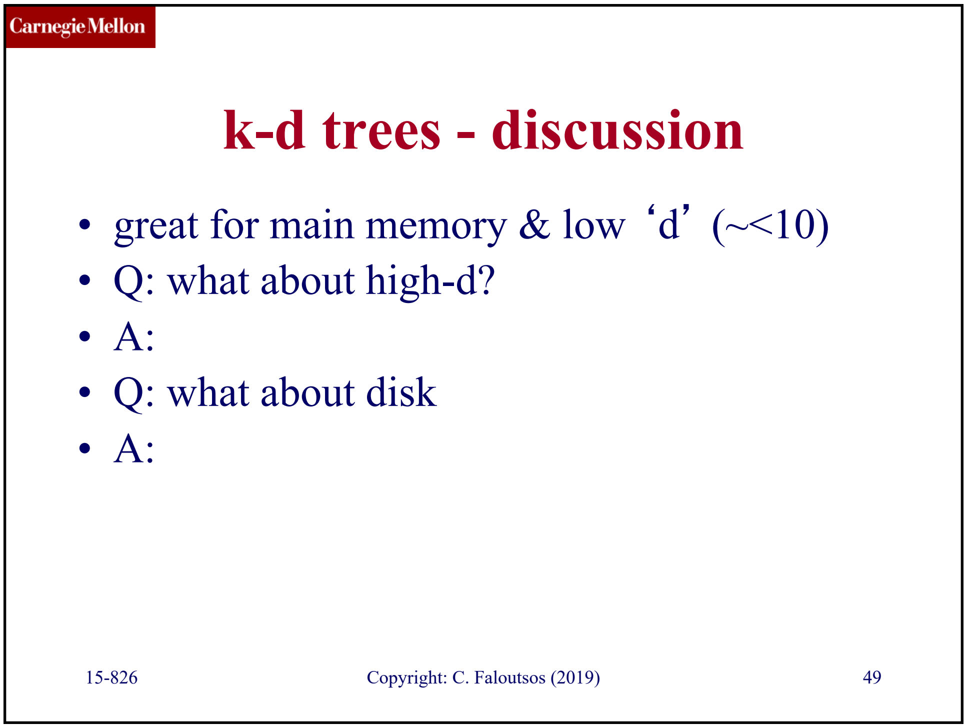
- A: check root; put in stack; proceed to child



Carnegie Mellon

Indexing - Detailed outline

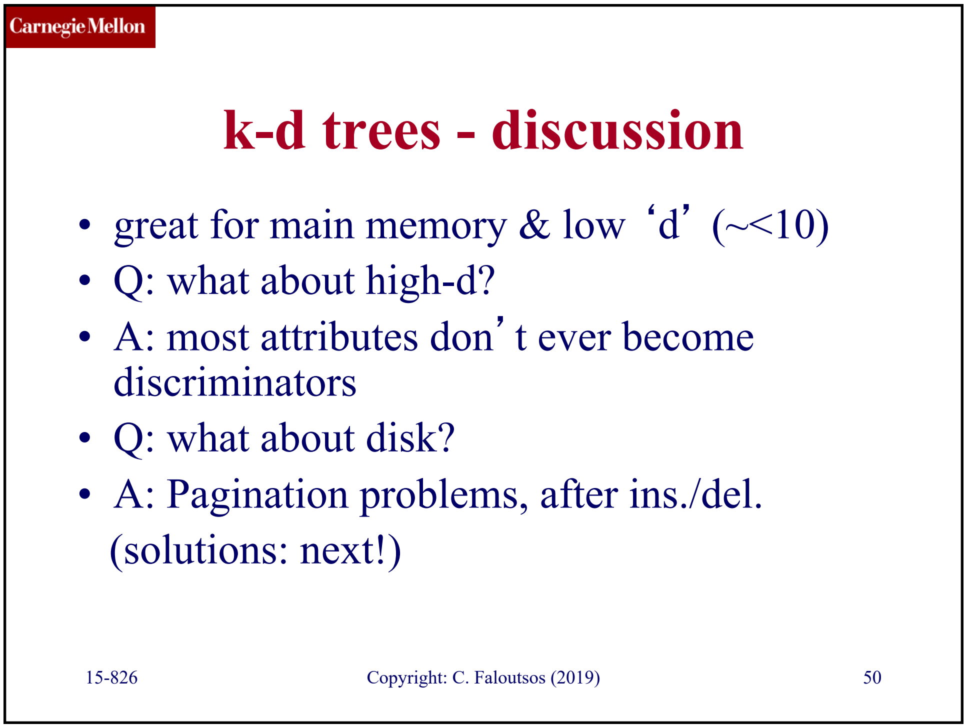
- primary key indexing
- secondary key / multi-key indexing
  - main memory: quad-trees
  - main memory: k-d-trees
    - insertion; deletion
    - range query; k-nn query
    - discussion
- spatial access methods
- text
- ...



Carnegie Mellon

k-d trees - discussion

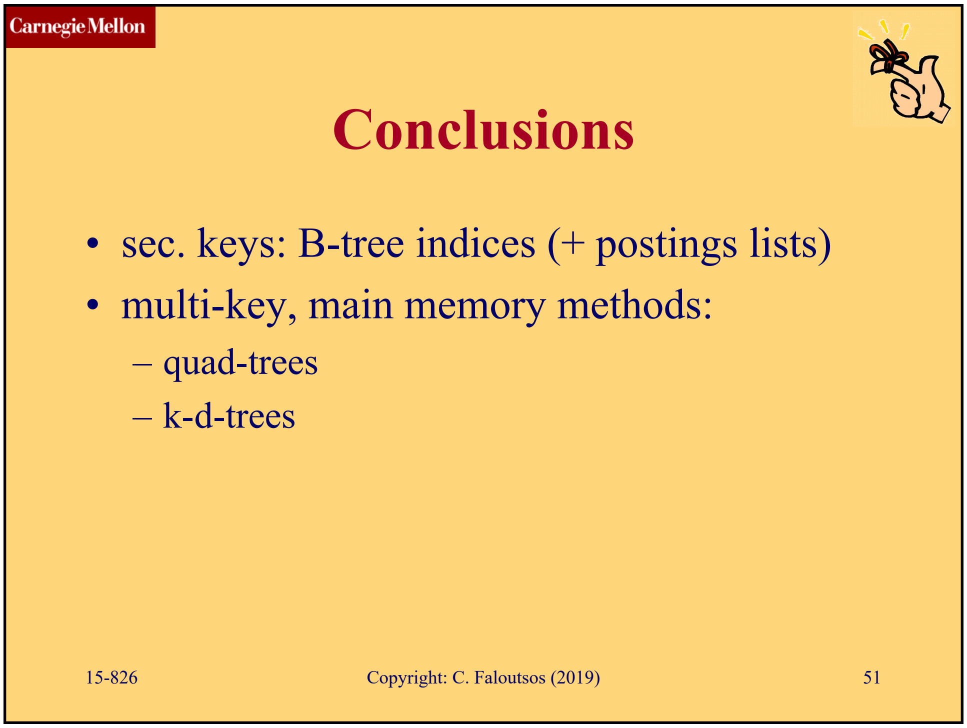
- great for main memory & low ‘d’ (~<10)
- Q: what about high-d?
- A:
- Q: what about disk
- A:



Carnegie Mellon

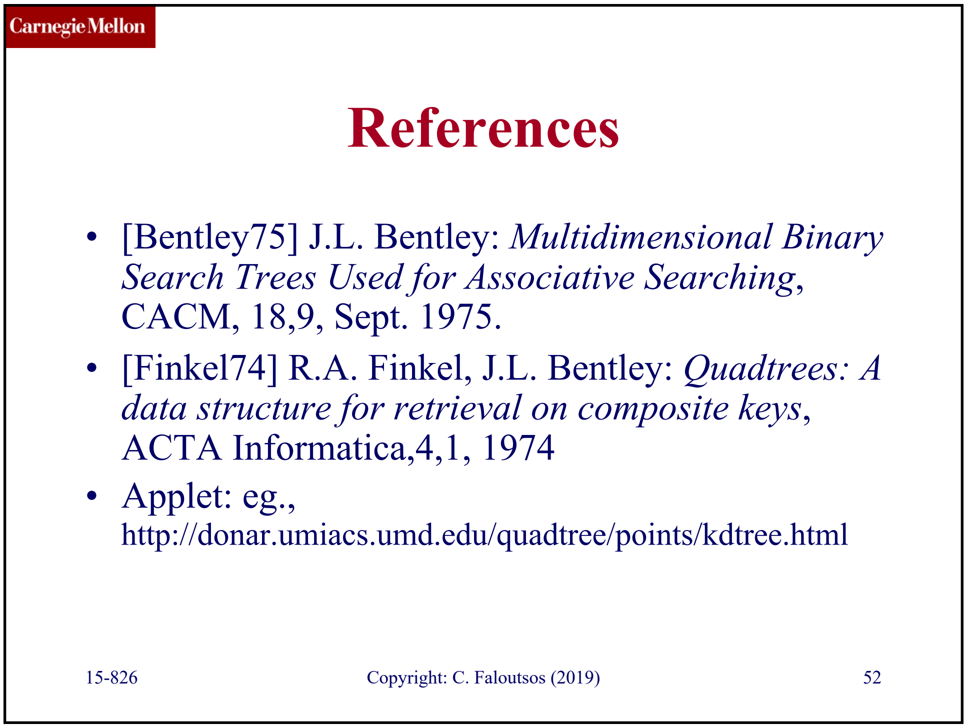
k-d trees - discussion

- great for main memory & low ‘d’ (~<10)
- Q: what about high-d?
- A: most attributes don’t ever become discriminators
- Q: what about disk?
- A: Pagination problems, after ins./del. (solutions: next!)



Conclusions

- sec. keys: B-tree indices (+ postings lists)
- multi-key, main memory methods:
  - quad-trees
  - k-d-trees



References

- [Bentley75] J.L. Bentley: *Multidimensional Binary Search Trees Used for Associative Searching*, CACM, 18,9, Sept. 1975.
- [Finkel74] R.A. Finkel, J.L. Bentley: *Quadtrees: A data structure for retrieval on composite keys*, ACTA Informatica,4,1, 1974
- Applet: eg., http://donar.umiacs.umd.edu/quadtree/points/kdtree.html