File Organization

Reminder

- Final exam
  - The date for the Final has been decided:
  - Saturday, November 16th
  - 8am – 10am
  - 01-2000

Project Notes

- Change your generic solver?
  - Don't forget to change your Clock and Farmer problem as well.
- Memory Management
  - Using purify
    - Add to Makefile:
      - `CCC = purify CC`
    - Or better yet, create a file `header.mak`
  - `Workshop` also has memory management tools.
- Parking Lot Problem: due Nov 11th

New plan

- Today: Files 1
- Tuesday: Exam / Files 2
- Thursday: Files 3

- Monday: Ethics
- Tuesday: Final Review
- Then we are done!

Before we start

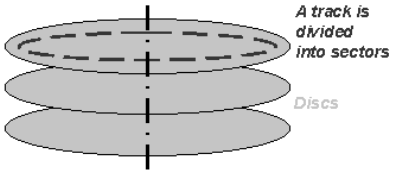
- Any questions

File Organization

- How to find stuff in a file
  - Improve access time by imposing a defined structure on a file
  - Database applications
  - Advanced searching strategies

It is always best to start at the beginning…

- Hard Disk – Physical layout



Hard Disk – Physical layout

- Hard disk
  - Is composed of a number of cylinder
  - Each cylinder is composed of tracks
  - Each track is composed of sectors
    - Sector is the lowest unit of read/writable data
  - Each sector can be addressed by giving
    - Cylinder number
    - Track number
    - Sector number
  - Info on a sector is called a block of data

Hard Disk – Performance

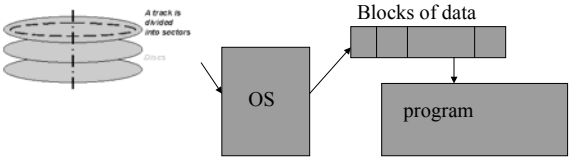
- Seek Time
  - time to reposition the head and increases with the distance that the head must move.
- Rotational latency time
  - time from when the head is over the correct track until the data rotates around and is under the head and can be read.
- Access Time
  - time from when a read or write request is issued to when the data transfer begins. It is the sum of the seek time and latency time.

Hard Disk – Performance

- Data Transfer Rate
  - rate at which data can be retrieved from the disk and sent to the controller.
- Mean Time to failure
  - is the number of hours (on average) until a disk fails.

Going up a level

- It is the responsibility of the Operating System to:
  - Map logical files to physical sector I/O.
  - Return blocks of data to a program.



Going up a level

- The OS must also:
  - Organize it's file system for maximum performance (I.e. low access time)
  - File System
- If the file is a searchable database
  - This optimization can be improved by imposing an additional structure on a file.

Terminology

- Record
  - Collection of data pertaining to one entity
    - E.g. Employee
  - Each record consists of a number of data fields.
  - Files are composed of a collection of records.
- Key
  - Field within a record upon which a search is made

Organization Strategies

- Sequential
- Indexed
- Hashing
- Tree-based Organization

Sequential Organization

- Simplest organization
  - Place records one right after another
  - Array / linked list of records

Sequential Organization

- Can order info by sorting records by key.
  - Search for record with given key is O(n)
  - 1,000,000 records
    - on avg 500,000 records reads to find desired record
  - Can this be improved by doing a binary search?

Only if underlying OS allows for random access

Sequential Organization

- Problem with sequential organization
  - Search is O(n).
  - Increases linearly as number of records increase.

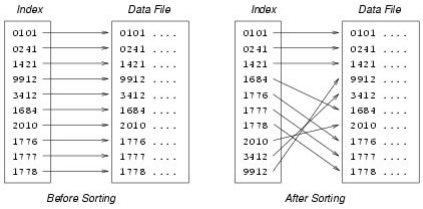
Indexed Files

- Construct an index
  - The index contains:
    - Each record
    - Record's location in the file.
    - Dense index – contains every key for every record
  - Searching the file involves
    - Searching the index for the key
    - Directly accessing the record based on the record's address in the index file.

Indexed Files

- To be efficient:
  - Index must be sorted by key
  - Index file must support direct access
  - Data file must support direct access
- Note that data need not be sorted, just the index.
- Searching the index is more efficient than searching the file since the index file will be smaller than the file being indexed.

Indexed Files



Indexed Files

- Example:
  - 1,000,000 records
  - 10 key/location pairs fit in a block
    - Index file 100,000 blocks long
  - On avg, search index would require 50,000 index block reads + 1 data block read.

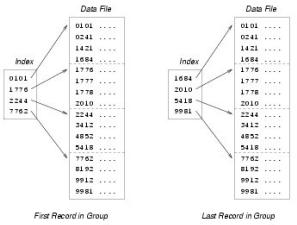
Indexed Files

- Problem with indexed files
  - Index contains key for each record
  - As number of records grow, so does the size of the index file.
  - Index file can become unmanageably large.

Indexed Sequential Files

- Variation on the indexed file
  - Index file is sparse
  - Data is stored sequentially by key
- Index files indexes groups of records
  - Index gives location of a group of records
    - Index can point to first record in a group
    - Index can point to last record in a group
  - Sequential search within each group

Indexed Sequential Files



Indexed Sequential Files

- Example
  - 1,000,000 records
  - 10 key/location pairs fit in a block
  - Every 10th record key is placed in index
    - Index size: 10,000 records
  - On average: 5,000 index block reads + 5 data record reads

Indexed Sequential Files

- For efficiency
  - Sparse index can be loaded into memory (if it is small enough).

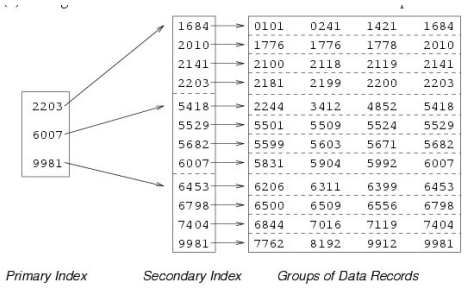
Searching Indexed Files

- Dense Index:
  - If key is found in index, location of record is given
  - Else the desired record is not present
- Sparse Index
  - If key is in index, location of record is given
  - Else index indicates group where record should be found
  - Sequential search from start of record group to end of group.
- Questions?

Multi-Level Indicies

- If data file becomes unmanageably large:
  - Construct a sparse index of the data
  - Construct a sparse index of the sparse index.

Multi-Level Indicies



Multi-Level Indicies

- Searching
  - Search primary index for where to start searching the secondary index
  - Search secondary index for where to start searching data.
  - Search data sequentially from location given in secondary index.
- Use as many levels as needed

Multi-Level Indicies

- Example
  - 1,000,000 records
  - 10 key/location pairs per record
  - Every 10th key in index
  - Create 3 index files:
    - 100,000 entries
    - 10,000 entries
    - 1,000 entries
  - On average: 500 blocks from 1st index, 5 from second, 5 from third, 5 data records.

Multi-Level Indicies

- Commercial Implementation
  - ISAM (Indexed Sequential Access Method) from IBM
    - Records blocked into tracks
    - Tracks grouped into cylinders
    - Each disk has a cylinder index
    - If multiple disks, there was a disk index
    - Useful for "static" files
      - Additions / deletions expensive
- Questions?

Sequential and Indexed Files

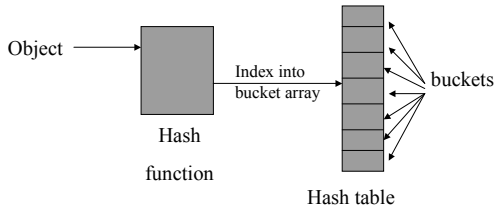
Organization	Records / avg search
Sequential	500,000
Indexed	50,001
Indexed Sequential	5,005
Multi-Level Index	515

Sequential and Indexed Files

- Questions?

Hashing

- Recall from CS2



Hashing

- Hashing Schemes
  - Each bucket corresponds to a sector
    - Best if 1 record fits nicely in a block
  - For large records (n blocks / record)
    - Hash to first block of record
  - For small record (< 1 block/ record)
    - Hash to record
    - Use extra space for collision management

Summary

- File organization Schemes
  - For efficient access to data records
- Organizations
  - Sequential
  - Indexed
    - Dense index
    - Indexed Sequential (Sparse Index)
    - Multi-level
  - Hashing
  - Next time: Tree-based organizations