

Object Oriented & Component based Software Development
Programming Style, classes, methods, UML
Interface, polymorphism, encapsulation, inheritance, AOT
Frameworks
4. Multithreading, i/o, n/w
Design Patterns
6. Typing & Memory management
Software engg - I
8. Software Engineering
Component based programming
Java beans, ESB components
Web services components

Fundamentals of Computing and Computer Systems
Digital Logic, Arithmetic
Sets, Theory of Computation
Data Structures

Algorithms
Computer Architecture - instruction sets, CPU, RISC, CISC
~~Supercomputer~~

OS - threads, processes
OS - concurrency, memory management
OS - I/O, File Systems

ADVANCES IN DATA MANAGEMENT

1. Relational model, SQL

→ Databases: collection of data; shared

→ DBMS: set of prog for managing database

→ database system: database + DBMS

→ 3 schema architecture: physical design + conceptual design (data model) + views

→ physical & logical independence

→ DDL + DML

→ objectives:

- controlling redundancy
- data structuring & efficient access
- simultaneous access
- interfaces
- security/integrity/backup
↓
(ref, val, structural)

→ relational model:

- table, tuple, column, domain (schema)
- key: rows
- relational algebra

- Key-value stores
 - store data indexed by key
 - ins / del / look up ops

→ doc stores

- complex, nested data models
- 1st, 2nd order
- data more flexibly partitioned

→ wide col stores

- records extended by attrs
- parallel & horizontal partitioning of data

→ graph dbs

- relations == entities
- graph specific indexes, ql etc

→ New SQL

- data fits into MM
- logging on peristent db

DATA INTEGRATION & WAREHOUSING

1- Introduction & basics, web data models

Why? apps need to work with data from diff sources

→ differences: - data models, schema, entities, duplication, conflicts

→ Proven of consolidation in order to provide -

- correctness & completeness
- single data model & schema
- 1 entity rep
- no conflicts

→ Heterogeneity

- Tech: means to access data
- Syn: encoding

- data preprocessing - normalize
- gather data, normalise values, apply sim measures, combine scores, decide match/non-match

blocking

- partition records into buckets
- sort :- generate key, sort by key

evaluation

- grand truth
TP, FP, FN, TN
- precision, recall, F-1

learning matching models

- training data & feature generation
- convert each training ex to pair (v, c)
& apply learning alg.

6. Data cleaning, quality & fusion

- providers have diff levels of knowledge, views, intentions
- info is wrong, biased, outdated, incomplete, inconsistent
- depends on consumer on how to use it
- Data Provenance
 - info on how data was produced
 - simple v/s full prov claims
who process
 - publish prov info on web
 - vocab
 - timestamp
 - metadata

→ Data Quality

- = fitness func
- enterprise / web context
(control) (no control)
- dimensions
 - content tech intellectual formatting

- significance level

- Errors

type I

(reject null when true)

type II

(fail to reject null when false)

- Tests

two-tailed

(true mean is diff from claimed)

upper-tailed

(true mean higher)

lower-tailed

(true mean lower)

- F-score

- ANOVA: analysis of variance

- two or more sample

- between / within group variance

- independence of observations / normality

DATA MINING

3. Data Mining Process, Preprocessing

→ extraction of useful patterns from data source, which are valid, novel etc

→ basic tasks:-

classification, clustering, association rule mining, regression

→ process:- understand application domain

select sources, collect data

preprocess

data mining

post processing

incorporate them

→ applications:- marketing

engineering

health

web

- distances of 2 clusters

Single Complete avg centroids

- distance functions

- euclidean

- manhattan

- chebychev

- document

- cosine

- data standardization

- attrs have a common range value

- interval-scaled

- ratio-scaled

- trial / error

- evaluation

- confusion matrix, precision, recall, f-1

- entropy, purity

- intra cluster cohesion / inter cluster separation

7. classification

- learn a target function that can predict values

- training / testing

- distance of training test data axes are

- decision tree

- NP hard best tree

- convert test of rules

- recursive partitioning based on information gain

- overfitting

preprocessing

postprocessing

- evaluation of classifiers

- accuracy

- holdout set, n-fold cross validation

- precision, recall, f-1

- score (prob that ex belongs to +ve class)

2. search strategies

→ Concepts

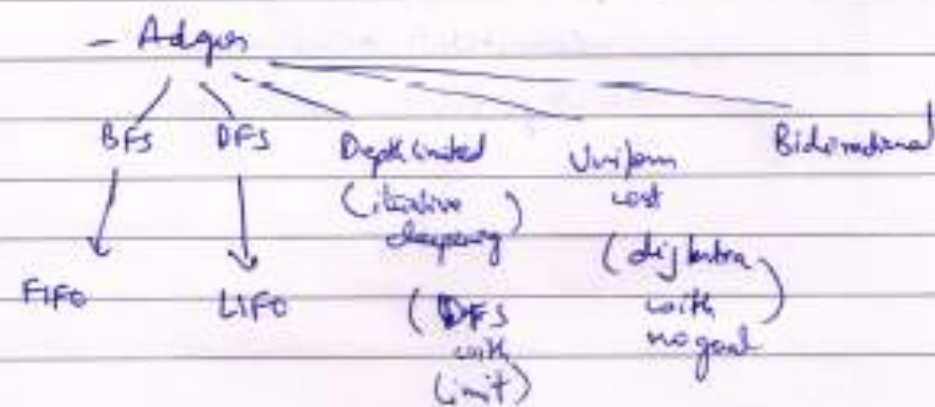
- State space
- path
- frontier (readable)
- solution (goal test)

→ Algos

- tree search (exp)
- graph search (no repeated states)

→ Informed Search

- strategy for expansion
- evaluation - completeness, space-time complexity, optimality (branching factor, depth)



→ Informed Search

- node expansion based on some estimate of distance to goal (best first search)

↓
heuristic for distance to goal

- greedy bfs - heuristic h is used for expansion
- A^* - $g(n) + h(n)$

→ Local search & optimization

- hill climbing (gradient descent)
- (local maxima)

→ Simulated annealing

- accept bad moves with prob

- genetic algorithms (random, mutate, crossover, fitness)

→ Game playing AI

→ Game search

- game trees
- evaluation function (for board position)

Adversarial search

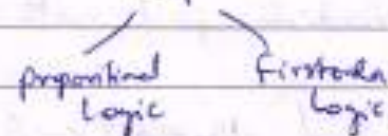
- min max (best move with no error)
- alpha-beta pruning Improve

* Knowledge representation and Logic

- knowledge based agent

- knowledge base

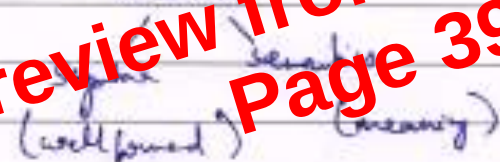
- rep



- reasoning via inferencing

- Logic

- sentences



- model: true/false

- entailment

- inference - alg is sound, complete

- Propositional Logic

- Syntax

$\neg, \vee, \wedge, \rightarrow, \leftrightarrow, T, F$

- inferencing

with rules

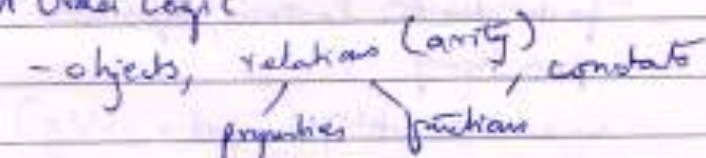
like $\{a \rightarrow b, a\} \models b$

- HKB²

- linear with horn clauses

(at most 1 +ve literal)

- First Order Logic



- quantifiers

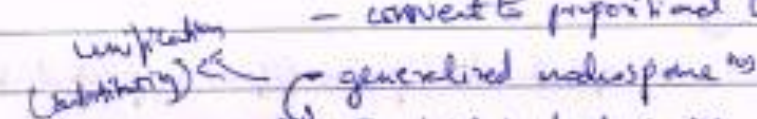


- equality

- closed world assumption

- inferencing

- convert to propositional logic (complete, inefficient)



- Forward/Backward chaining

- resolution

(convert to CNF, remove P.L. without quantifiers)
by Skolemization

name node in main memory

- data node

- block server, block reports

- pipelining

- block placement, data correctness, failures,

- distributed access structures

- indexing

tree / hash

- consistent hashing - distributed hash directories

- Amazon dynamo - BTree

- Bigtable - BTree + non-indexed files

↓
data model: row + key (tablets)

5. Map Reduce & Hadoop

→ divide & conquer

→ scale-out, hide sys details from programmer

- sync workers

(comm + shared resources access)

- prog models lower (mpi, threads)

- DP: - master/slave, prod-cons, Shared worker queues

→ map - shuffle & sort - reduce

extract group aggregate

→ MR - runtime handles

- scheduling

- data distribution

- sync

- error & faults

- coordination & dealing with failures

→ DFS + LFS

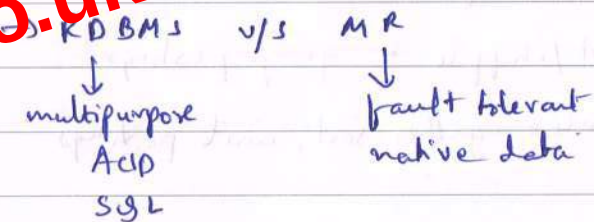
- value-key inversion
- sort by value

→ Beyond MR:

· BSP

- processors + comm
- supersteps - local comp + comm + barrier sync
- pregel, graph, hama

7. Data Management, Link Analysis, information retrieval in the cloud



→ DB workloads

- OLTP

real-time, small queries, random reads & writes

- OLAP

batch, less concurrency, complex query,

large scans for query

- hadoop: cheaper than partitioned disks

- rel alg ops: proj, sel, join, cartesian, union, diff, groupby, rename

- proj: map over tuples

sel: map over tuples

groupby: reduce

join: map-side, reduce-side, in-memory (hash)

- HIVE, PIG

→ Index construction

- map over all docs, emit term as key
(doc, tf) as value
- sort/shuffle - group postings
- reduce - gather, sort, write postings

→ Retrieval

- MR optimized for throughput
latency, mrstart is exp, not for
real-time
- partitioning of terms/docs
- replication, caching
- routing.

→ Link analysis

key: node n

value: adjacency list

map: adjacency list

sort/shuffle: group distances by reachable
nodes

reduce: set min distance for each
reachable node

SEMANTIC WEB TECHNOLOGIES

1. Introduction, Layers

→ goals & foundations of semantic web

→ today: syntactic web

computers do presentation, humans do

linking & interpretation

impossible tasks

- complex queries involving background
knowledge

- web service

- agents

- location info in data repositories

→ make implicit knowledge to explicit

→ add extra info in a std way to
make it m/c readable

→ SW is a set of technologies to realize
web of data

→ formal m/c understandable languages,
rules, ontologies

ontology: explicit specification of
a conceptualization

layers:

- unicode + uri
- xml + namespace + xmlschema
- rdf + rdfls
- ontology vocab
- logic
- proof
- trust

HTML: is about presentation

XML: more structure (nesting)
content / formatting (data exchange)
tree model
DTD/XMLS
namespace
XSL

- XSLT
- xpath
- XSL-FO

not powerful enough for sw

2. RDF ~~rest~~ →

→ subject, predicate, object (mathematical logic)

↓
domain, range (unary relation)
(n-ary)

→ resource description framework

→ s, p, o all are uris

→ triples, xml syntax

→ blank node

→ container elements

- bag
- seq
- alt

→ verification

- start abt start

→ adapter

- resolve incompatible interfaces

→ visitor

- define new op without changing class

→ singleton

- only 1 instance of a class created

→ proxy

- for an object to control references to it

→ command

- encapsulate request to it

→ factory

- create objects without exposing instantiation logic to clients

6. Typing, Memory management

→ Typed vs untyped languages

- for every operation, types of data for which it is applicable

- weak / strong typing

↓
type-safe

prevent value of one type to be treated as another

- static / dynamic typing

↓
compile-time

run-time

(no type for variable during prog.)

→ Activation records: allocated when proc is entered & dealloc when proc is exited
(info needed by single exec of proc)

- interchangeable
- high composability
- goals:
 - manage complexity
 - change
 - reuse
- specification, implementation, installed component → one or many

- principles: decomposition, abstraction, reusable, S/W dependable, productivity, standardization

- $C = \langle P, M, E, I \rangle$

id module event interface
 (visible, type, access, parameters)

- infrastructure: component model
connection model
deployment model

- contracts

10. Java beans, EJB

Java beans

- software component model for Java
- properties
 - simple (appearance & behavior)
 - bound (value change, ^{notified} ~~modified~~)
 - constrained (value changes can be vetoed)
 - indexed (null-value)
- events
 - introspection
 - bean info
- persistence via serialization

- deadlock

- conditions

- mutual exclusion
- hold & wait
- no preemption
- circular wait

- Resource Allocation Graph

- deadlock prevention / avoidance

- (future process requests)
- banker algorithm
- (claim matrix, resource matrix)

- Memory Management

- prog relative address → actual address
- translation
- register: base & bound

- partitioning, placement

- fixed
- dynamic
- (internal fragmentation)

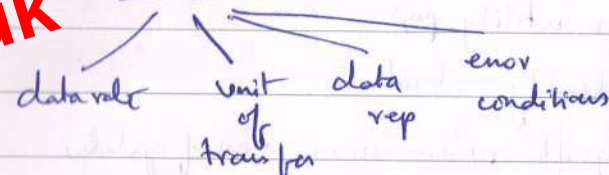
- best fit
- first fit
- next fit

- virtual memory

- paging (replacement, LRU, tables, locality, ref, thrashing)

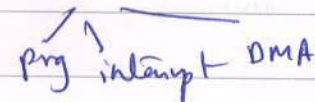
8. Operating Systems - I/O, File systems

- devices



- device controllers

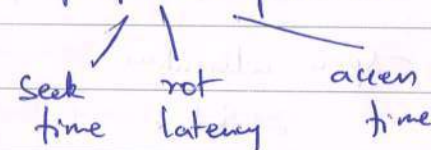
- I/O



- device drivers

- buffering
 - block
 - stream

- disk performance parameters



- disk scheduling

- FIFO, SSTF, SCAN

- RAID

- disk cache, LRU