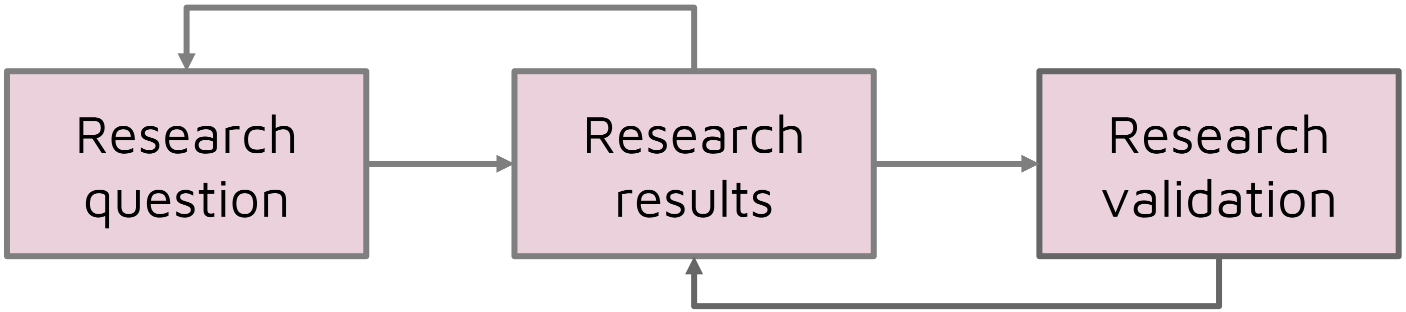
Research Methods in Software and Systems Engineering

Aldeida Aleti and Adel N. Toosi

Software Engineering Research process



Types of research questions

FEASIBILITY	Does X exist, and what is it? Is it possible to do X at all?
CHARACTERIZATION	What are the characteristics of X? What exactly do we mean by X? What are the varieties of X, and how are they related?
METHOD/MEANS	How can we do X? What is a better way to do X? How can we automate doing X?
GENERALIZATION	Is X always true of Y? Given X, what will Y be?
DISCRIMINATION	How do I decide whether X or Y?

Example: Software Architecture Research Questions

FEASIBILITY	Is it possible to automatically generate code from an architectural specification?
CHARACTERIZATION	What are the important concepts for modeling software architectures?
METHOD/MEANS	How can we exploit domain knowledge to improve software development?
GENERALIZATION	What patterns capture and explain a significant set of architectural constructs?
DISCRIMINATION	How can a designer make tradeoff choices among architectural alternatives?

QUALITATIVE & DESCRIPTIVE MODELS	Report interesting observations Generalize from (real-life) examples Structure a problem area; ask good questions
TECHNIQUES	Invent new ways to do some tasks, including implementation techniques Develop ways to select from alternatives
SYSTEM	Embody result in a system, using the system both for insight and as carrier of results
EMPIRICAL MODELS	Develop empirical predictive models from observed data
ANALYTICAL MODELS	Develop structural models that permit formal analysis

QUALITATIVE & DESCRIPTIVE MODELS	Early architectural models Architectural patterns Domain-specific software architectures
TECHNIQUES SYSTEM	UML to support object-oriented design Architectural languages
EMPIRICAL MODELS	Communication metrics as indicator of impact on project complexity
ANALYTICAL MODELS	Formal specification of higher-level architecture for simulation

PERSUASION	I thought hard about this, and I believe…
IMPLEMENTATION	Here is a prototype of a system that…
EVALUATION	Given these criteria, the object rates as…
ANALYSIS	Given the facts, here are consequences…
Formal model	Rigorous derivation and proof
Empirical model	Data on use in controlled situation
EXPERIENCE	Report on use in practice
Qualitative model	Narrative
Decision criteria	Comparison of systems in actual use
Empirical model	Data, usually statistical, on practice

Example: Automated testing (AT) research validation

PERSUASION	Early automated testing, random testing
IMPLEMENTATION	Implementation of AT on an industrial system
EVALUATION	Comparison of search-based software testing with random testing
ANALYSIS Formal model Empirical model	Algorithm selection for Automated Software Testing
EXPERIENCE Qualitative model Decision criteria Empirical model	User studies with industry experts on the usefulness of automated software testing

Building blocks for SE research

Question	Results	Validation
Feasibility	Qualitative model	Persuasion
Characterisation	Technique	Implementation
Methods/means	System	Evaluation
Generalisation	Empirical model	Analysis
Selection	Analytical model	Experience

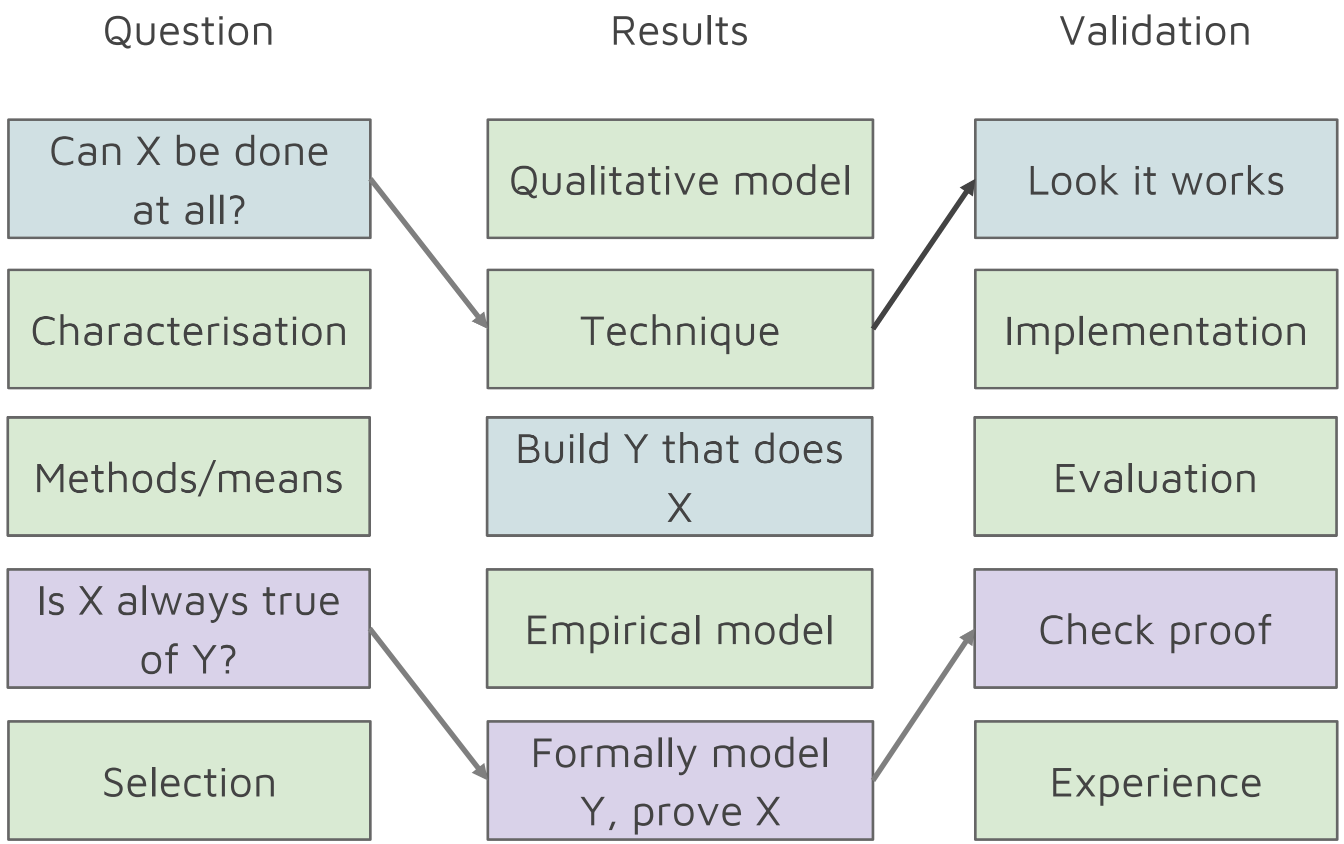
A common good plan

Question	Results	Validation
Feasibility	Qualitative model	Persuasion
Characterisation	Technique	Implementation
Can X be done better?	Build Y	Measure Y, compare to X
Generalisation	Empirical model	Analysis
Selection	Analytical model	Experience

A common, but bad plan

Question	Results	Validation
Feasibility	Qualitative model	Persuasion
Characterisation	Technique	Implementation
Methods/means	System	Evaluation
Generalisation	Empirical model	Analysis
Selection	Analytical model	Experience

Two other good plans



Interesting, novel, exciting results that significantly enhance our ability

to develop and maintain software

to know the quality of the software we develop

to recognize general principles about software

or to analyze properties of software

You should explain your result in such a way that someone else could use your ideas

What is new here?

Use verbs that shows RESULTS, not only efforts

Awful	I completely and generally solved … (unless you actually did)
Bad	I worked on (studied, investigated, sought, explored) skedaddling
Poor	I worked on improving (contributed to, participated in, helped with) skedaddling
Good	I showed the feasibility of predicting software defects with machine learning. I significantly improved the accuracy of detecting software defects (or proved, demonstrated, created, established, found, developed)
Better	I automated the generation of software tests. With a novel application of search techniques, I achieved a 10% increase in coverage and a 15% improvement in detecting bugs over the standard method.

What has been done before? How is your work different or better?

Awful	The skedaddling problem has attracted much attention [2, 3, 4, 5, 7].
Bad	Trer [4] and Amil [6] worked on skedaddling.
Poor	Trer [4] addressed skedaddling by jumping, while Amil [6] took a skipping approach.
Good	Trer's jumping approach to skedaddling [4] achieved 60% coverage [8]. Amil [6] achieved 80% by skipping, but only for light-free cases [34].
Better	Trer's jumping approach to Skedaddling [4] achieved 60% coverage [8]. Amil [6] achieved 80% by skipping, but only for light-free cases [34]. We modified the jumping approach to use the agility representation of skipping and achieved 90% coverage while relaxing restrictions so that only running is prohibited.

Distributed System definitions - many and varying:

- A system in which **hardware** or **software** components located at networked computers **communicate** and **coordinate** their actions only by passing **message** [Coulouris]
- A collection of independent computers that appears to its users as a **single coherent system** [Tanenbaum]

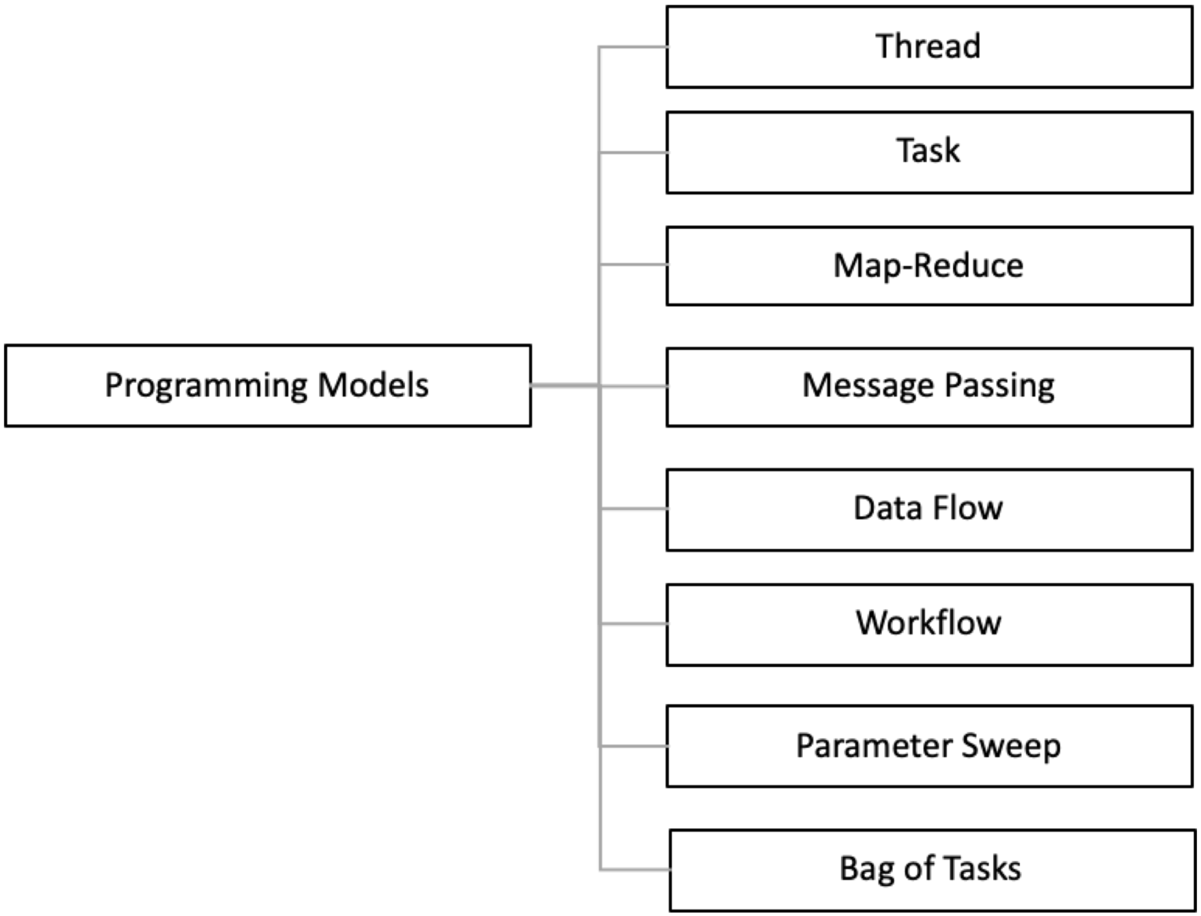
Computer Networks vs **Distributed Systems**:

A **Computer Network**: Is a collection of spatially separated, interconnected computers that exchange messages based on specific protocols. Computers are addressed by IP addresses.

- A **Distributed System**: Multiple computers on the network working together as a system. The spatial separation of computers and communication aspects are hidden from users.

- Heterogeneity
- Openness
- Security
- Scalability
- Failure Handling
- Concurrency
- Transparency
- Quality of Service

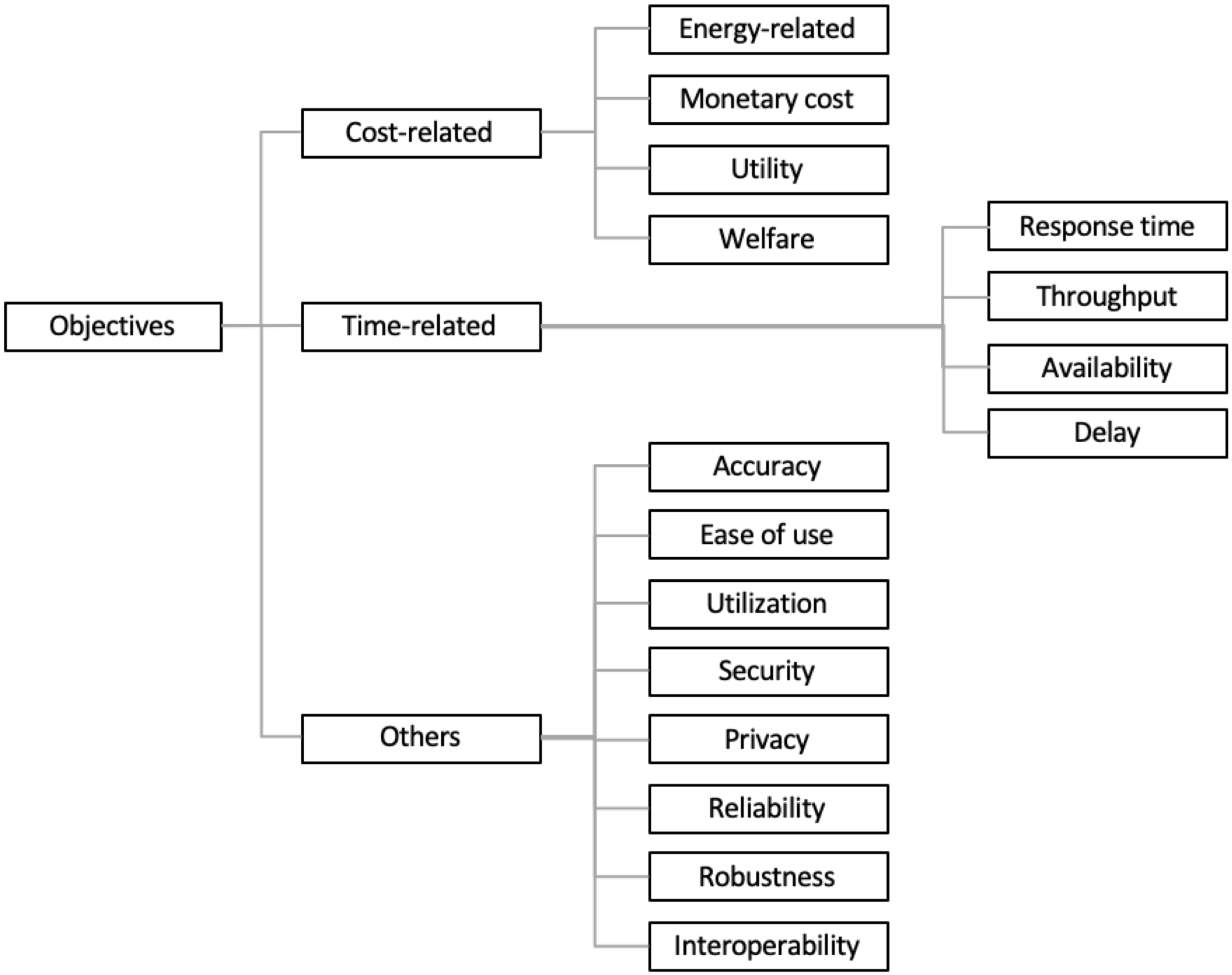
Application Programming Models In Distributed Systems



Platforms

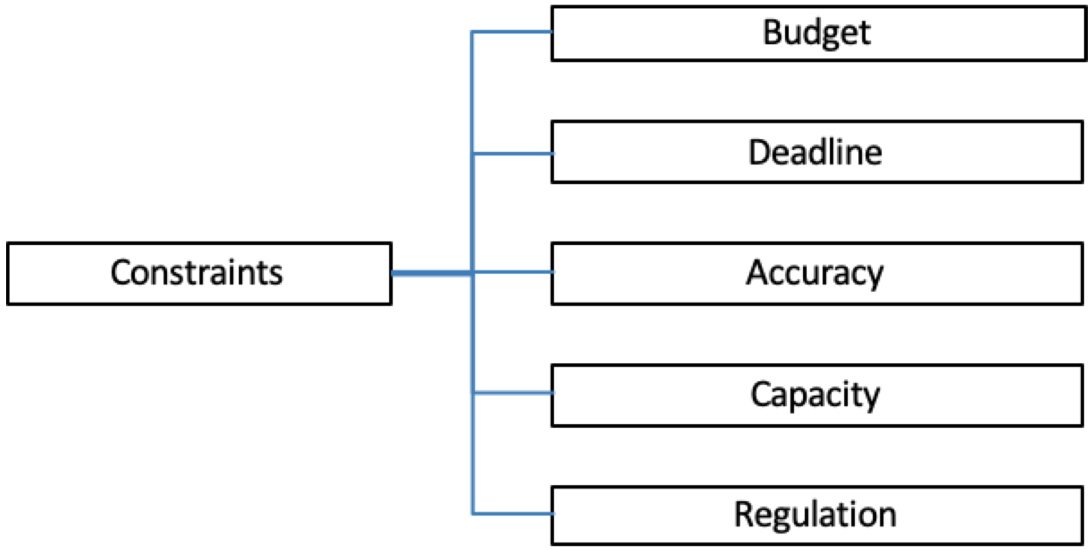
- Cluster
- Grid
- Cloud
- Peer-to-Peer Systems
- Supercomputers
- Mobile Computing
- Sensor Networks
- Internet of things
- Edge and Fog Computing
- Content delivery networks (CDN)
- Software Defined Networks (SDN)
- …

Objectives

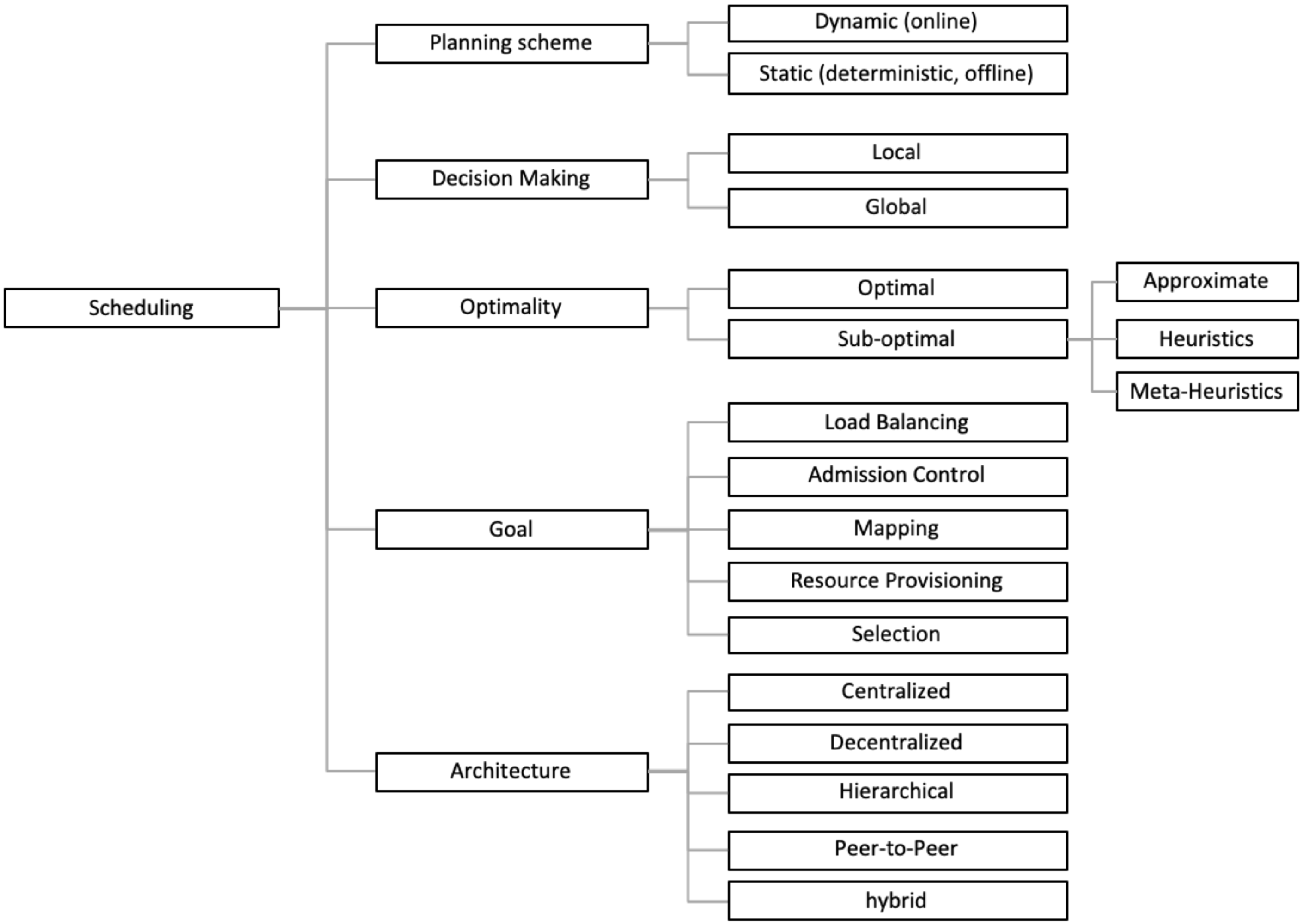


Objectives
Cost-related
Time-related
Others
Energy-related
Monetary cost
Utility
Welfare
Response time
Throughput
Availability
Delay
Accuracy
Ease of use
Utilization
Security
Privacy
Reliability
Robustness
Interoperability

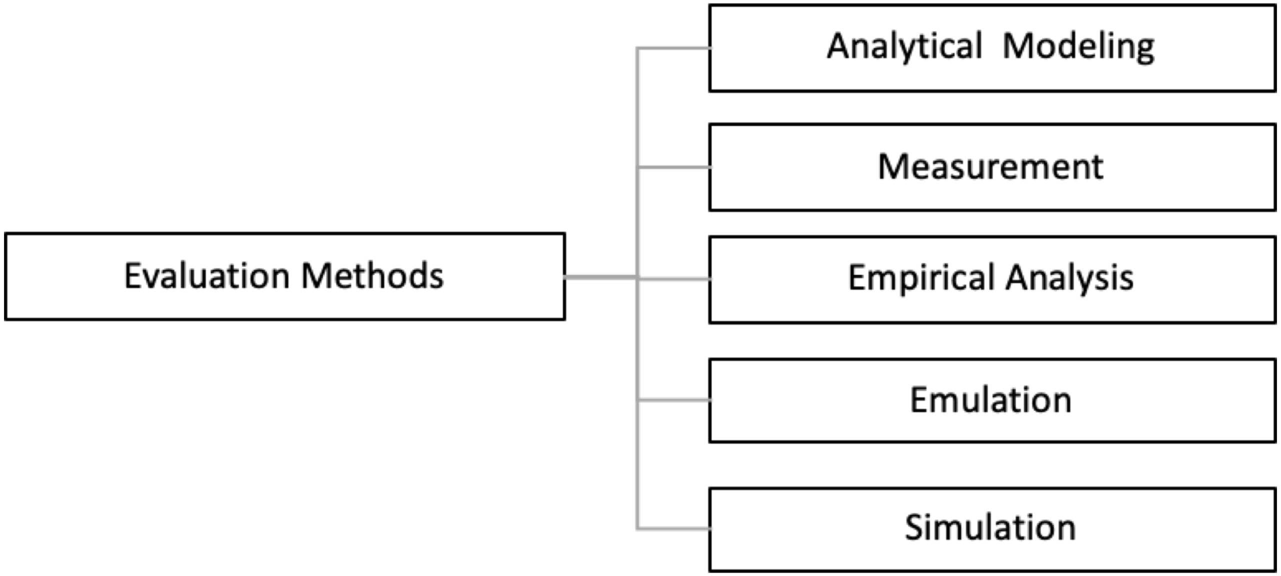
Constraints



Scheduling: One of the challenging areas of research in DS



Evaluations Methods



Common Mistakes in Performance Evaluation

- No Goals/Biased Goal
- Unsystematic Approach
- Analysis without understanding the problem
- Incorrect Performance Metrics
- Unrepresentative workload
- Wrong Evaluation Technique
- Overlooking Important Parameters
- Ignoring significant factors
  - ***Sensitivity analysis***
- Inappropriate Experimental Design
  - ***Full factorial design***
- Inappropriate level of detail
- No analysis/Erroneous Analysis
- Ignoring Errors in Input
- Improper Treatment of Outliers

Common Mistakes in Performance Evaluation (Cont.)

Too complex Analysis

Assuming No change in the Future

Ignoring Social Aspects

- Weak presentation leads to rejection of the high-quality analyses

Ignoring Variability

- If the variability is high the mean alone is misleading.

Improper Presentation of Results

Ignoring or Omitting Assumptions and limitations Variability

Jain, Raj. ***The Art Of Computer Systems Performance Analysis: Techniques For Experimental Measurement, Simulation, And Modeling****. john wiley & sons.*

Please find more complete slides here: ***http://adelnadjarantoosi.info/ppt/common.pptx***

Problem

- Short Background (If necessary)
- Scope
    - Application model, e.g., Map-reduce
    - Platform, e.g., Cluster
- Objective, e.g., Cost and Energy Consumption
- Constraints, e.g., Capacity and Available Renewable Energy

Methodology

- E.g., Online scheduling using meta-heuristics

Evaluation Method

- Analytical proofs, Simulation, Emulation, Real Implementation

Results/Findings

Conclusion/Implications

A sample of good abstract

Problem:

- In this paper, we present BlinkDB, a massively parallel, approximate query engine for running interactive SQL queries on large volumes of data. BlinkDB allows users to trade-off query accuracy for response time, enabling interactive queries over massive data by running queries on data samples and presenting results annotated with meaningful error bars.

Methodology:

- To achieve this, BlinkDB uses two key ideas: 1) an adaptive optimization framework that builds and maintains a set of multi-dimensional stratified samples from original data over time, and 2) a dynamic sample selection strategy that selects an appropriately sized sample based on a query's accuracy or response time requirements.

Evaluation:

- We evaluate BlinkDB against the well-known TPC-H benchmarks and a real-world analytic workload derived from Conviva Inc., a company that manages video distribution over the Internet. Our experiments on a node cluster show

Results and Conclusions:

- that BlinkDB can answer queries on upto 17TBs of data in less than seconds(over 200× faster than Hive),with in an error of 2-10%.

Agarwal, S., Mozafari, B., Panda, A., Milner, H., Madden, S. and Stoica, I., 2013, April. BlinkDB: queries with bounded errors and bounded response times on very large data. In Proceedings of the 8th ACM European Conference on Computer Systems (pp. 29-42).

Summary

- Software Engineering Research process
  - Research Question, Research Results, and Research Validation
- Three Evaluation Technique:
  - Measurement
  - Simulation
  - Analytical Modeling
- Common mistakes in performance evaluation
  - Sensitivity analysis
  - Factorial design