Distributed Database Systems

Databases, Aarhus University

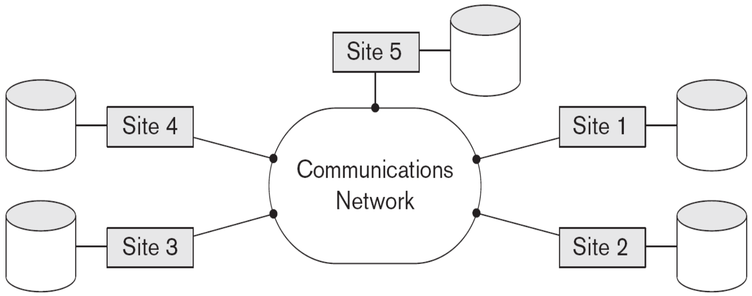
Ira Assent

Intended learning outcomes

- Be able to
  - Apply distributed concurrency and recovery approaches
  - Describe characteristics of distributed databases

ira@cs.au.dk

Recap: Distributed Databases



- Distributed Database (DDB)
  - collection of multiple logically related databases distributed over network
  - distributed database management system: software system
- Transparency: hide implementation details from users
  - Offers flexibility to user / developer
  - More challenging than centralized DBMS
- Use fragmentation and allocation schemata to identify possible query plans
  - For vertical fragmentation (π), keep attribute list for each fragment L_i
  - For horizontal fragmentation (σ), keep guard (condition) C_j
  - Mixed (hybrid) fragments: store both attribute list and condition L_i, C_j
- Cost optimization: least total data transfer across network
- MapReduce programming model
  - function style map and reduce tasks on key-value data, automatically parallelized
    - Fault tolerance, data distribution, load balancing, task communication
    - No distributed systems experience required by user/programmer

ira@cs.au.dk

Employee * Project

Strategies:

1. Transfer Employee and Project to query site, perform join
2. Transfer Employee to site 2, execute join at site 2, send result to query site
3. Transfer Project to site 1, execute join at site, send result to query site

Employee: 1000 rows, row size 20 bytes
Project: 3000 rows, row size 50 bytes
Each result tuple 65 bytes
Join selectivity: 9 projects per employee

What is the best strategy?

1. Strategy 1
2. Strategy 2
3. Strategy 3
4. Depends on query site

ira@cs.au.dk

Employee * Project

1. Transfer Employee and Project to query site, perform join there
   - Query site 3: transfer 20,000 + 150,000 = 170,000 bytes
   - But, if query site 2: only need to transfer Employee: 20,000 bytes
   - But, if query site 1: only need to transfer Project: 150,000 bytes
2. Transfer Employee to site 2, execute join at site 2, send result to query site

Query result of 9000 tuples if every employee contributes to 9 projects on average: query result size = 65 * 9000 = 585,000 bytes

   - Query site 3: total transfer 585,000 + 20,000 = 605,000 bytes
   - But, if query at site 2, only need to transfer Employee 20,000 bytes
   - But, if query at site 1: 605,000 bytes
3. Transfer Project to site 1, execute join at site 1, send result to query site
   - Query site 3: total transfer 585,000 + 150,000 = 735,000 bytes
   - If query site 2: 735,000 bytes
   - If query site 1: 150,000

Employee: 1000 rows, row size 20 bytes
Project: 3000 rows, row size 50 bytes
Each result tuple 65 bytes
Join selectivity: 9 projects per employee

ira@cs.au.dk

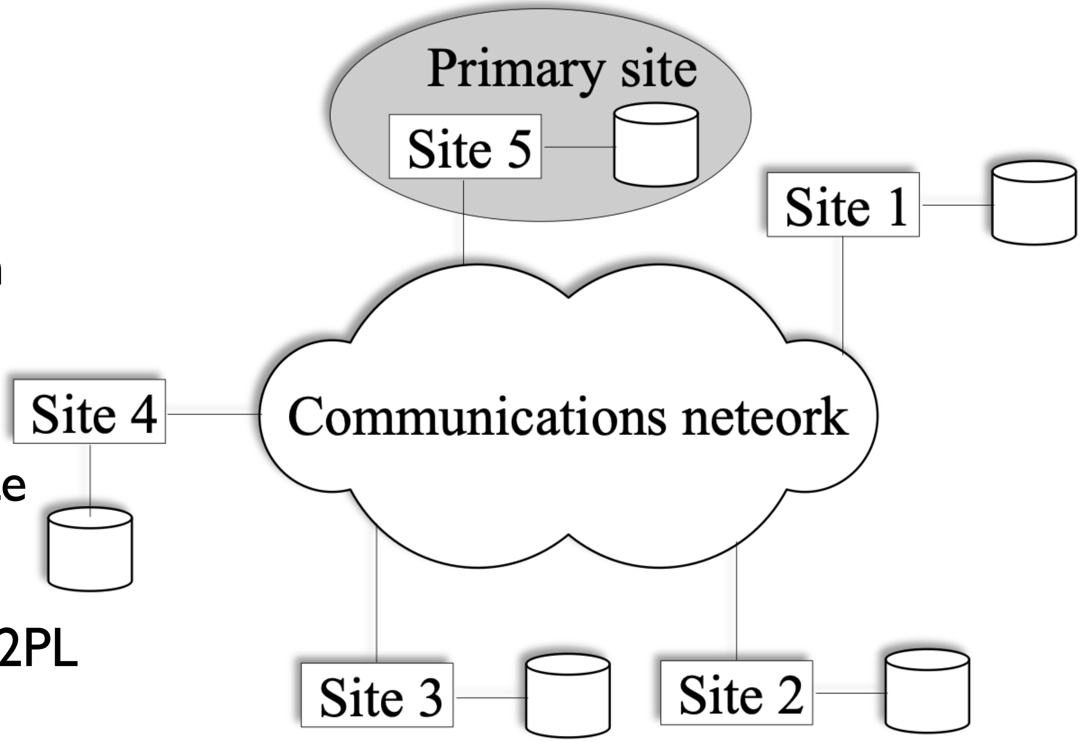
Concurrency Control and Recovery

- DDBs: additional concurrency control and recovery issues due to multiple copies and fragmentation of data items:
    - Maintain global consistency
    - Recover all copies and ensure their consistency
- Failure of individual sites
  - Database availability must not be affected due to the failure of one or few sites
  - Recovery scheme must recover them before they are available for use
  - Communication link failure
    - Communication link failure may create network partition
    - May affect database availability even though all database sites running
  - Distributed commit
    - Transaction may be fragmented and executed by several sites
    - Requires a two- or three-phase commit approach for transaction commit
  - Distributed deadlock
    - Two or more sites may get involved in deadlock

ira@cs.au.dk

Distributed CC: primary site

- Distributed Concurrency control based on a distributed copy of a data item
  - Primary site technique
    - Single site serves as coordinator for transaction management
    - Concurrency control and commit managed by this site
    - In 2PL, it manages locks
      - If all transactions follow 2PL at all sites, serializability guaranteed



ira@cs.au.dk

Primary site discussion

- Advantages of primary site distributed concurrency control
  - Extension of centralized two phase locking
    - Implementation and management simple

  - Data items locked only at one site
  - Can be accessed at any site
- Disadvantages
  - All transaction management activities go to primary site
    - likely to overload the site
  - If primary site fails, the entire system is inaccessible
- To aid recovery
  - Backup site designated
  - Shadow of primary site
  - In case of primary site failure, backup site can act as primary site

ira@cs.au.dk

Primary Copy Technique

- Instead of a site, a data item partition is designated primary copy
  - Different items have their primary copy potentially at different sites
- To lock a data item lock just the primary copy of the data item
- Advantages
  - Since primary copies distributed at various sites, single site is not overloaded with locking and unlocking requests
- Disadvantages
  - Identification of a primary copy complex
  - Distributed directory must be maintained, possibly at all sites

ira@cs.au.dk

Coordinator failure

Recovery from a coordinator failure

- In both approaches coordinator site or copy may become unavailable
- Requires selection of new coordinator

Primary site approach with no backup site

- Aborts and restarts all active transactions at all sites
- Elects new coordinator and initiates transaction processing

Primary site approach with backup site

- Suspends all active transactions
- Designates backup site as primary site and identifies new backup site
- Primary site receives all transaction management information to resume processing

Primary and backup sites fail or no backup site

- Use election process to select a new coordinator site

ira@cs.au.dk

Concurrency control based on voting

- No primary copy of coordinator
- Send lock request to sites that have data item
- If majority of sites grants lock then the requesting transaction gets the data item
- Locking information (grant or denied) is sent to all these sites
- To avoid unacceptably long wait
  - Time-out period
  - If requesting transaction does not get any vote information it aborts

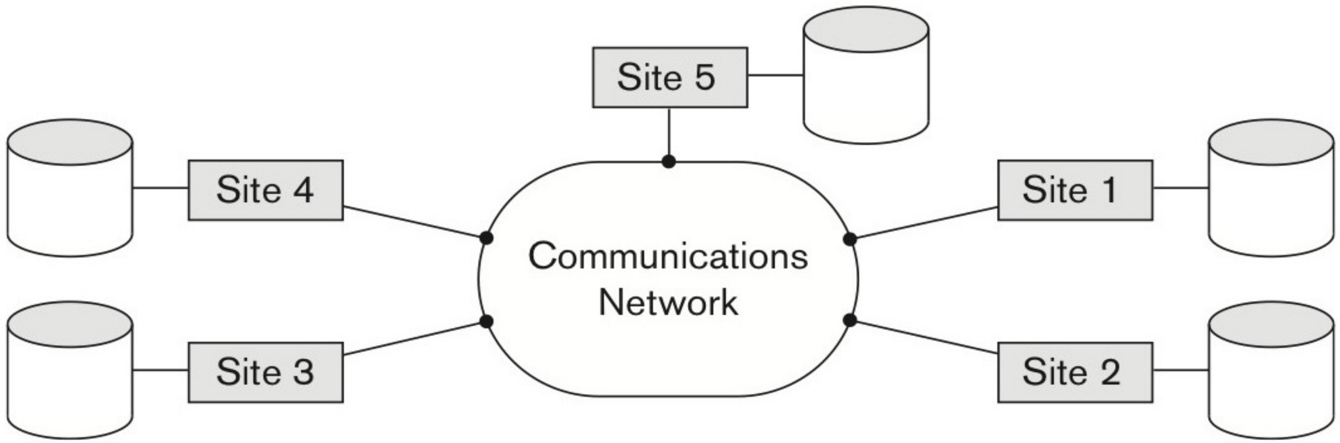
ira@cs.au.dk

Which choice of coordinator is the most efficient choice in distributed concurrency control?

1. Primary copy
2. Primary site with backup
3. Primary site without backup
4. Voting

Concurrency and recovery

- A transaction may run in a distributed fashion at multiple nodes
  - Sequence of predefined steps
    - Transaction commits only when all nodes agree to commit individually the part of the transaction they were executing
- Coordinator manages the protocol
- This commit scheme is referred to as "**two-phase commit**" (**2PC**)
  - If any one of these nodes fails or cannot commit the part of the transaction, then the transaction is aborted
- Each node recovers the transaction under its own recovery protocol



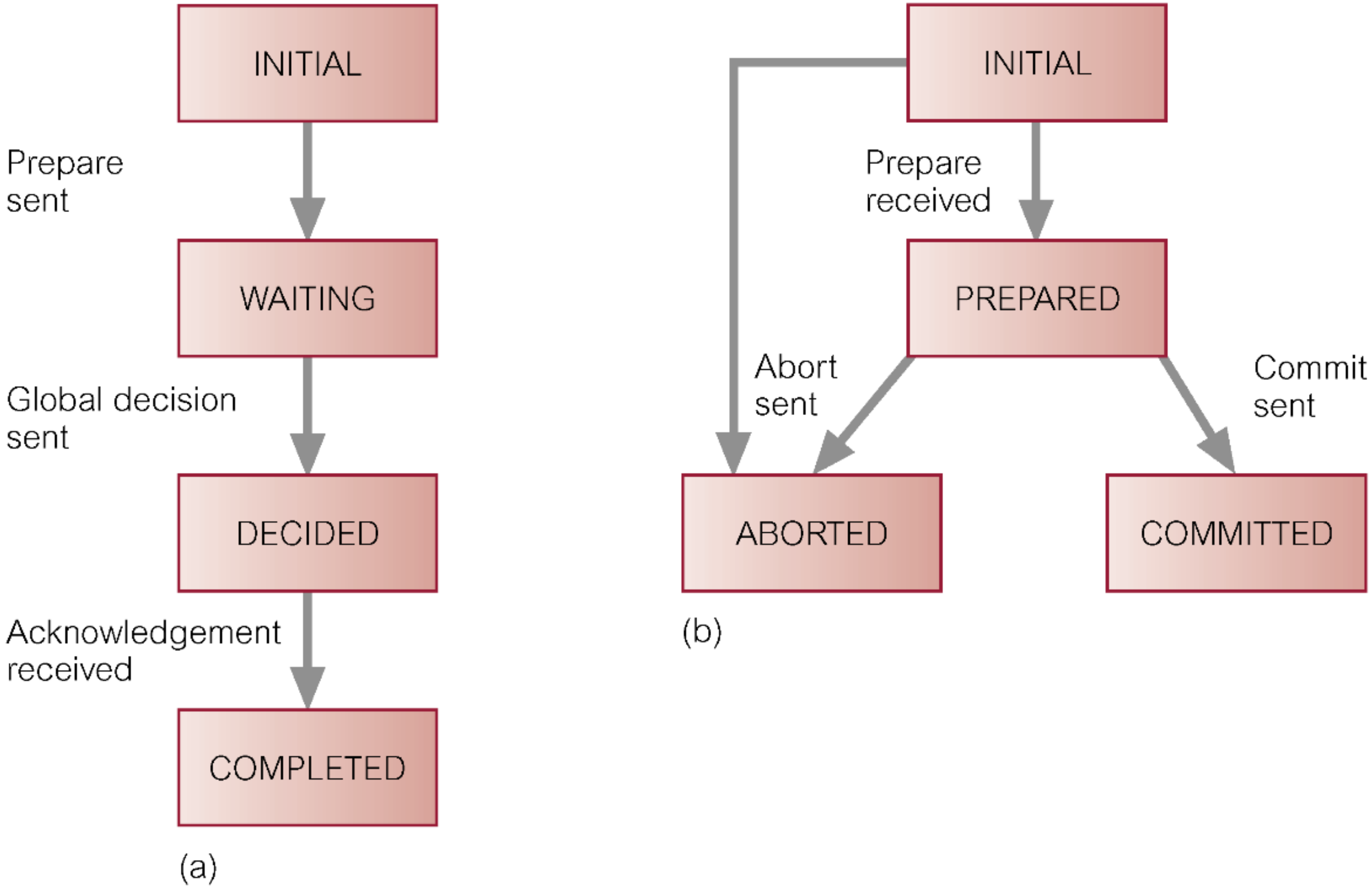
ira@cs.au.dk

2PC protocol

- Global recovery manager / coordinator maintains information in addition to local logs and tables (for each database)
- Phase 1
  - All participating databases signal the coordinator that their part of the transaction has concluded
  - Coordinator sends ”prepare for commit” message to all participants
  - Participants receiving message force-write all log records to disk
    - If done, return ”ready to commit” message to coordinator
    - If this fails, return ”cannot commit” message
    - No reply within certain time frame is interpreted as ”cannot commit”
- Phase 2
  - If all votes are positive, coordinator sends ”commit” message to all participants
    - Participants commit
  - If there is at least one negative (or missing) vote, then send ”roll back” message to all participants
    - Participants undo locally

ira@cs.au.dk

State Transition Diagram for 2PC



(a) coordinator; (b) participant

ira@cs.au.dk

What happens?

A node fails after voting commit (ready-to-commit) but before receiving global instruction (commit)…

A. Proceeds after local recovery
B. Proceeds if no other component has aborted
C. Is blocked if only communication with other sites that have no global instruction yet
D. Is blocked even if communication with other sites that have received instructions

ira@cs.au.dk

Three-phase commit

- Two-phase commit may be blocking
  - E.g. a process that times out after voting commit but before receiving global instruction is blocked if only communication with other sites that are also unaware of global decision
  - **Rare situation, so in practice 2PC common**
- Three-phase commit
  - **Introduce a third phase, precommit**, between voting and global decision
  - When receiving votes from all participants, coordinator sends out **precommit message**
  - Acknowledge pre-commit
  - Coordinator receives all pre-commits then sends global commit message

ira@cs.au.dk

3PC

- Using a precommit message
  1. Means that sites have more time to prepare local commit
  2. Means that participants receiving such a message know that all participants have voted commit
  3. Means that in case of failure of a node, participants abort
  4. Means that in case of failure of a node, participants are blocked

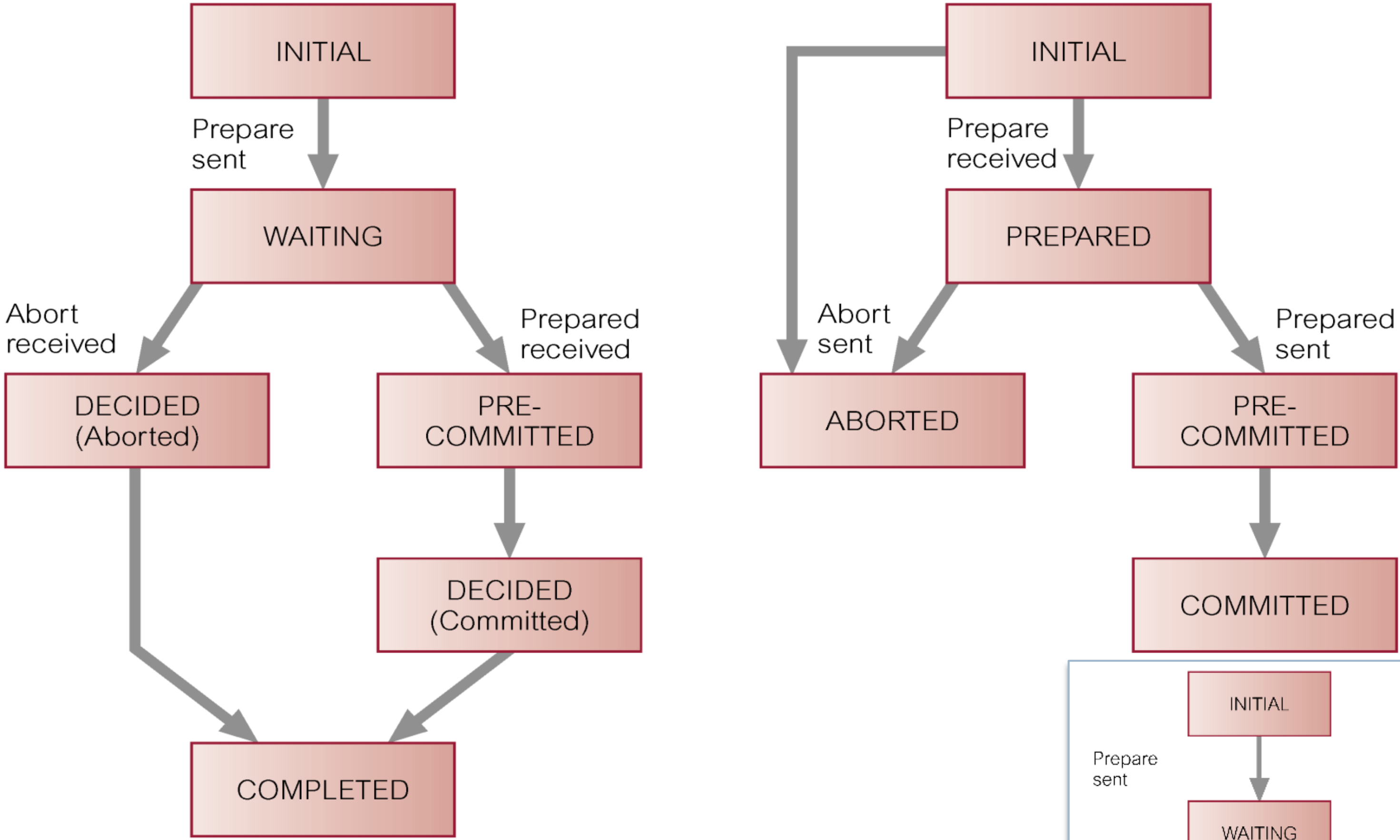
ira@cs.au.dk

Three-Phase Commit Protocol (3PC)

- Essentially divide the second commit phase into two subphases
  - Pre-commit
  - Commit
- Pre-commit
  - Communicate the result of the vote phase to all participants
  - If all participants vote yes, coordinator sends precommit message
  - If a participant receives a precommit message, it knows all have voted commit
    - Can continue even in case of failure
- Means that all operational processes have been informed of a global decision to commit prior to the first process committing, and thus act independently in the event of failure
  - No blocking in case of node failure (but, possible for communication failure…)

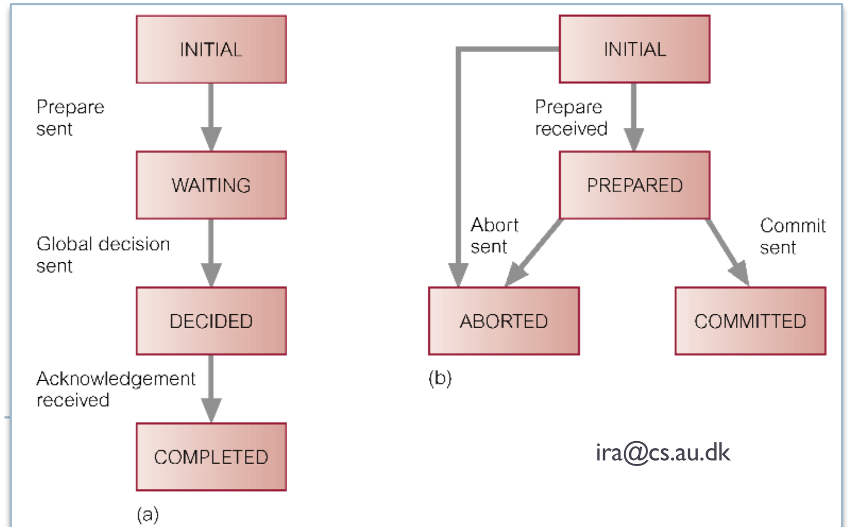
ira@cs.au.dk

State Transition Diagram for 3PC (vs. 2PC)



(a) (b)

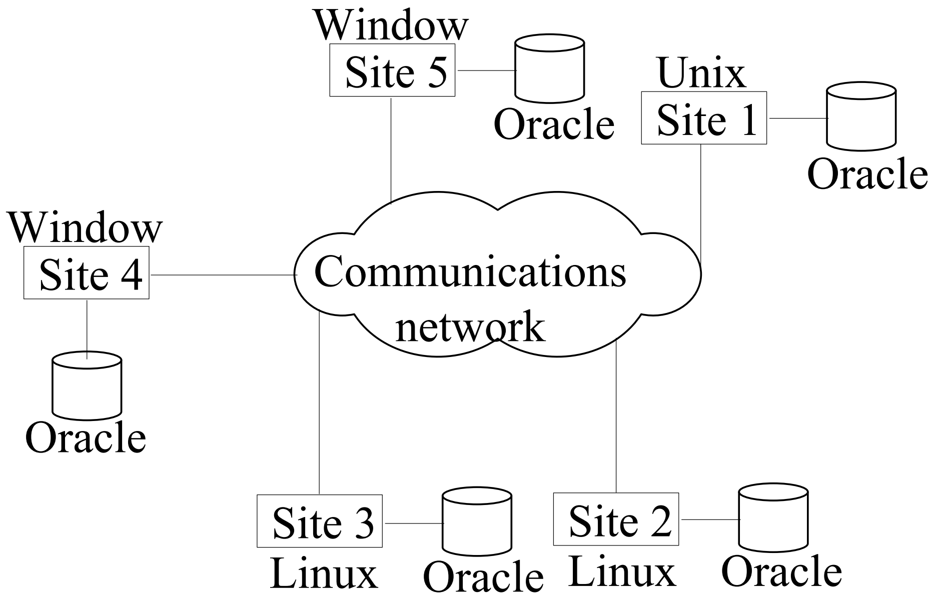
(a) coordinator; (b) participant



ira@cs.au.dk

Homogeneous Distributed Database Systems

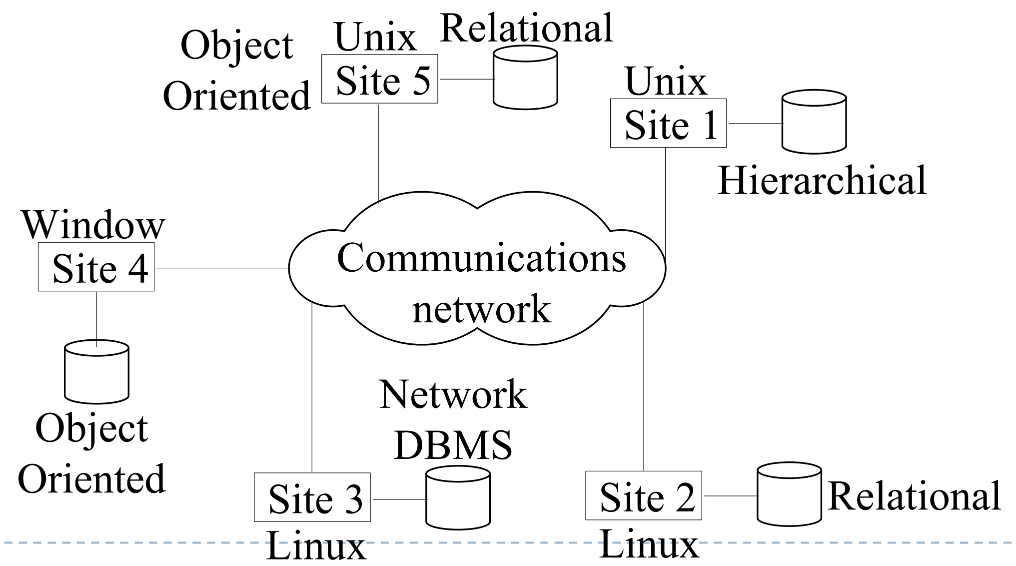
- Homogeneous
  - All sites of database system have identical setup, i.e., same database system software
    - For example, all sites run Oracle or DB2, or Sybase or some other database system, but same for everyone
  - Underlying operating system may be different
    - Underlying operating systems can be a mixture of Linux, Windows, Unix, etc.



ira@cs.au.dk

Heterogeneous Distributed Database Systems

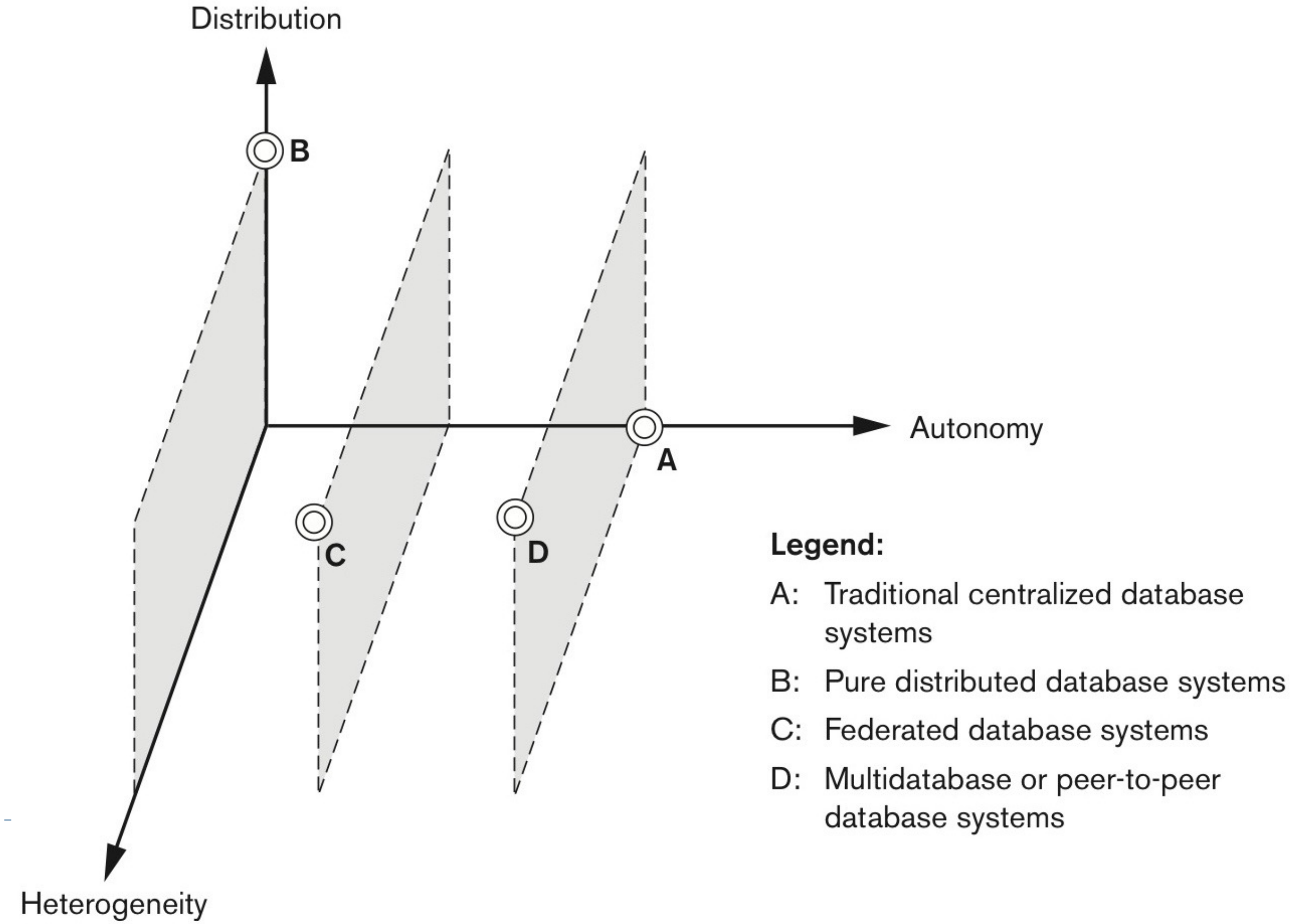
- Heterogeneous
  - Federated: Each site may run different database system but the data access is managed through a single conceptual schema
    - This implies that the degree of local autonomy is minimum
    - Each site must adhere to a centralized access policy
    - There may be a global schema
  - Multidatabase: There is no one conceptual global schema
    - For data access a schema is constructed dynamically as needed by the application software



ira@cs.au.dk

Types of Distributed Database Systems

- Federated Database Management Systems Issues
  - Differences in data models: Relational, Objected oriented, hierarchical, network, etc.
  - Differences in constraints: Each site may have its data accessing and processing constraints
  - Differences in query language: Some site may use SQL-89, some may use SQL-92, etc.



What is efficient for catalog mgmt?

Assume a dynamic database application, i.e., many updates. How should we manage the catalogs, i.e., the metadata information on the relations stored, attributes, etc. ?

1. Central catalog at coordinator site
2. Fully replicated catalogs at all sites
3. Local catalogs of data at local site
4. Local catalogs with replicates

Distributed catalog management

- Catalogs: databases with metadata about distributed database system
  - Centralized catalogs
    - Entire catalog stored in single site
    - Easy to implement, but little reliability, availability, autonomy or load distribution
    - Reads lock catalog and sent to reading site
    - All updates process through central site: bottleneck for write-intensive work loads
  - Fully replicated catalogs
    - Each site stores copy of entire catalog
    - Faster reads locally
    - Updates broadcasted, treated as transactions, using centralized two-phase commit
    - Also performance bottleneck for write-intensive work loads
  - Partially replicated catalogs
    - Each site stores catalog information on data stored at that site
    - Sites may cache entries from remote sites, but these may not be fully updated
    - System tracks catalog entries for original (birth) site and sites with copies; changes to copies propagated to original
    - Updating copies may be delayed until access to this data occurs

ira@cs.au.dk

Intended learning outcomes

- Be able to
  - Apply distributed concurrency and recovery approaches
  - Describe characteristics of distributed databases

Coming up:
Wrap up, example exam questions and pointers to research and courses

ira@cs.au.dk

What was this all about?

Guidelines for your own review of today's session

- Distributed database systems have the benefit of…
  - However, we need to account for…
- Compared to centralized databases, distributed concurrency control and recovery need to address…
  - 2PC and 3PC work as follows…
  - They were designed to…
- A distributed system is defined as…
  - We have discussed the following types…
- Query and update decomposition means…
  - The catalog stores…

ira@cs.au.dk