Distributed, Parallel, and Alternative Architecture Databases

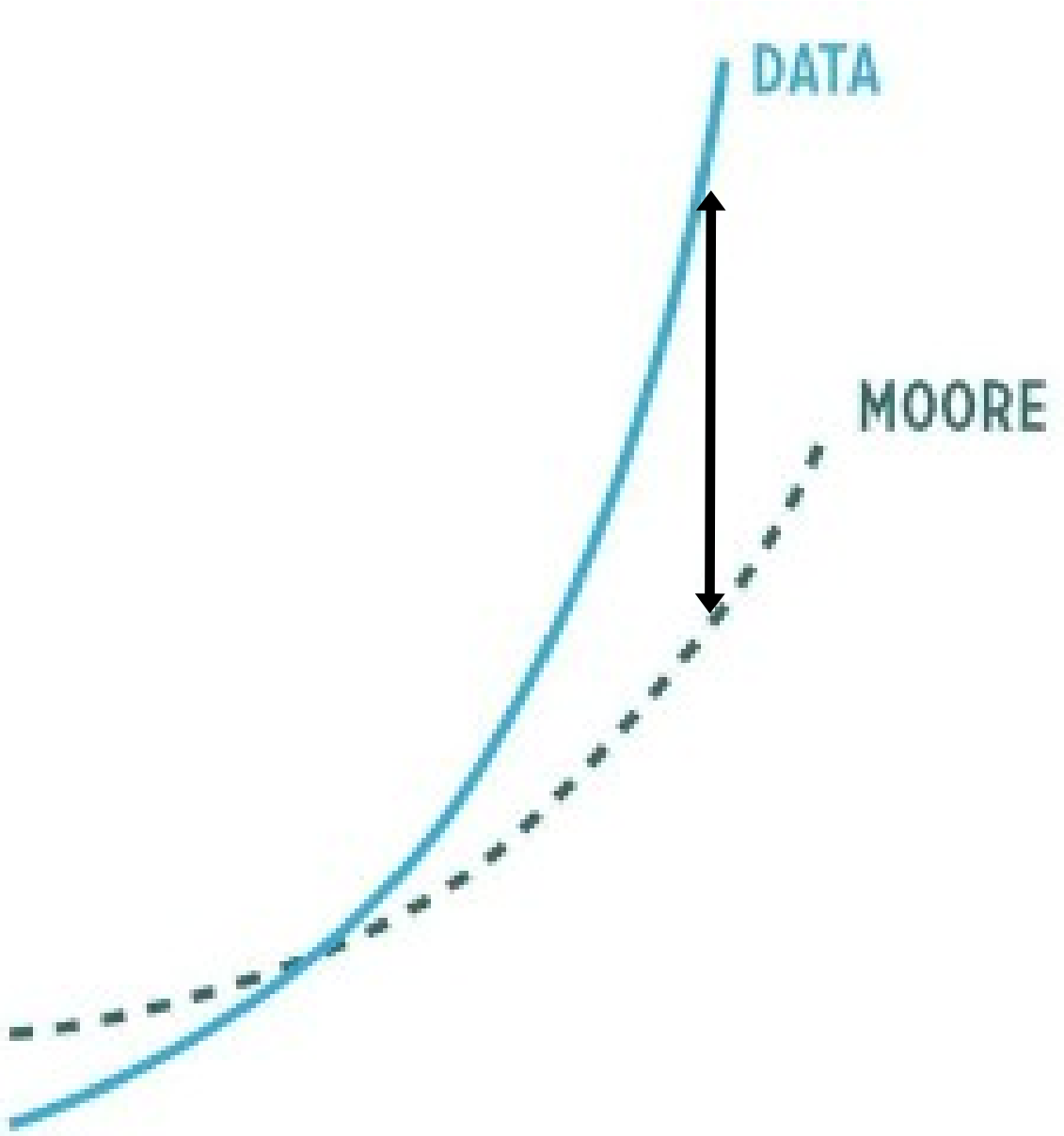
Bancos de Dados

Luiz Celso Gomes-Jr
gomesjr@dainf.ct.utfpr.edu.br

Outline

- Terminology
- Parallel Databases
- Distributed Databases
- Client-server Architecture
- Alternative Architectures

Need for speed

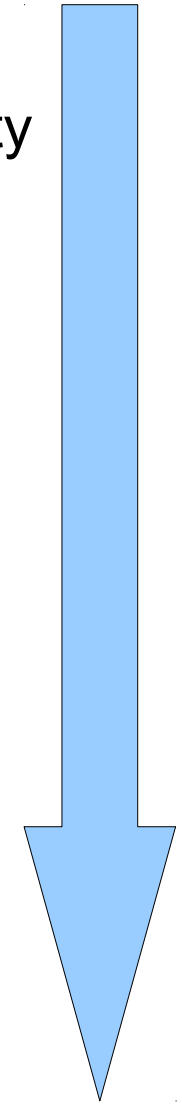


Exercício 1

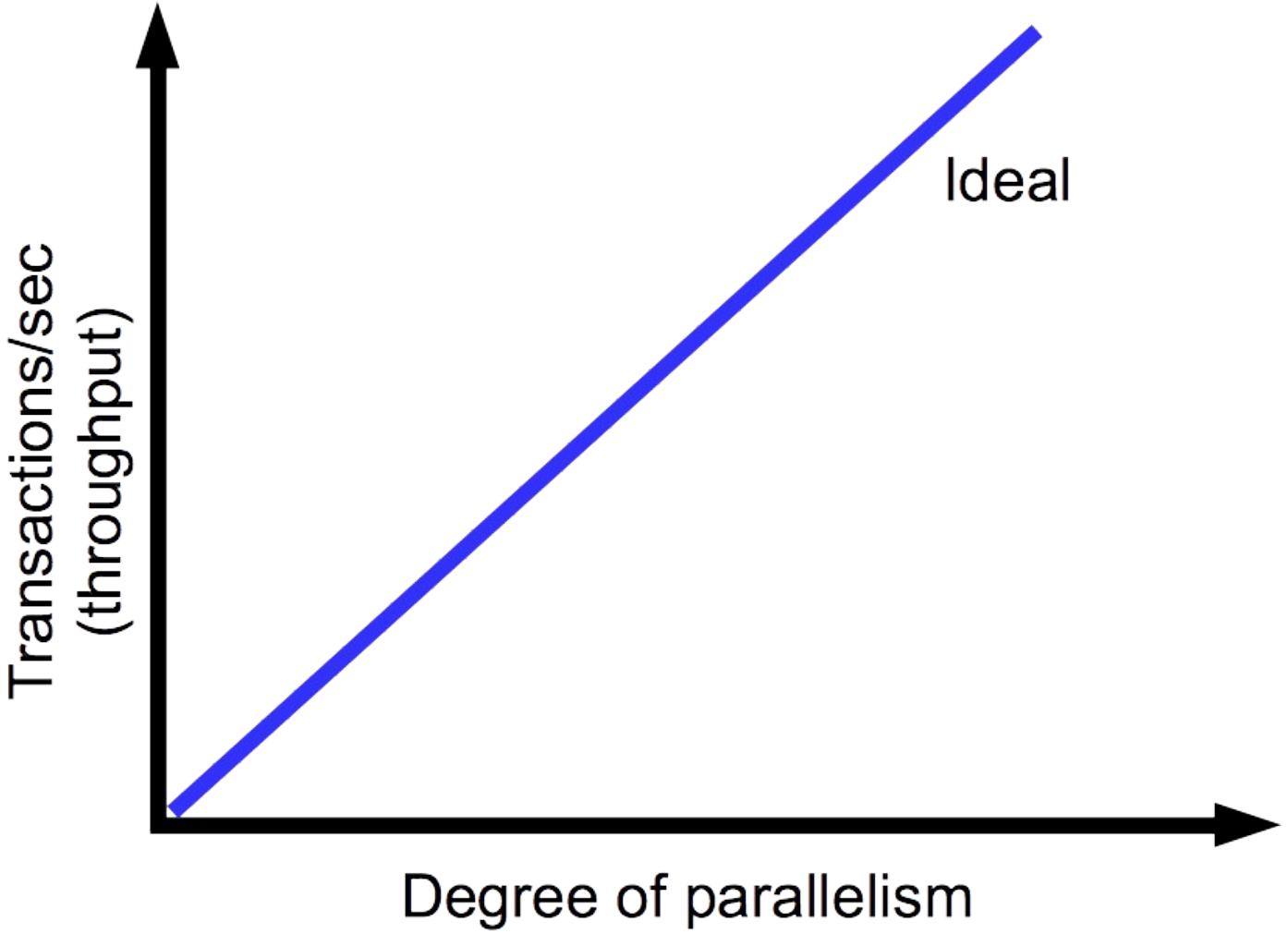
- [Preliminares] Suponha que a DIRGRAD esteja enfrentando problemas para atender as consultas online de CR dos alunos (o tempo de resposta é muito longo). As tabelas do banco são descritas abaixo. Quais técnicas (ao menos duas) vocês poderiam aplicar para melhorar o desempenho das consultas?
- Aluno(RA, nome, curso)
- Disciplina(codigo, nome)
- Cursa(RA, codigo, nota)

Need for speed

- Bigger computers: Faster CPUs
- Parallel: Multiple CPUs
- Distributed: Multiple Servers
- Alternative Architectures: Specialized CPUs
- Alternative Frameworks: adapt DBMS to the task (NoSQL, next class)
- Alternative Data Structures: adapt DBMS to the type of data (Spatial, Multimedia, Temporal, Active, Documents, Graphs... soon)

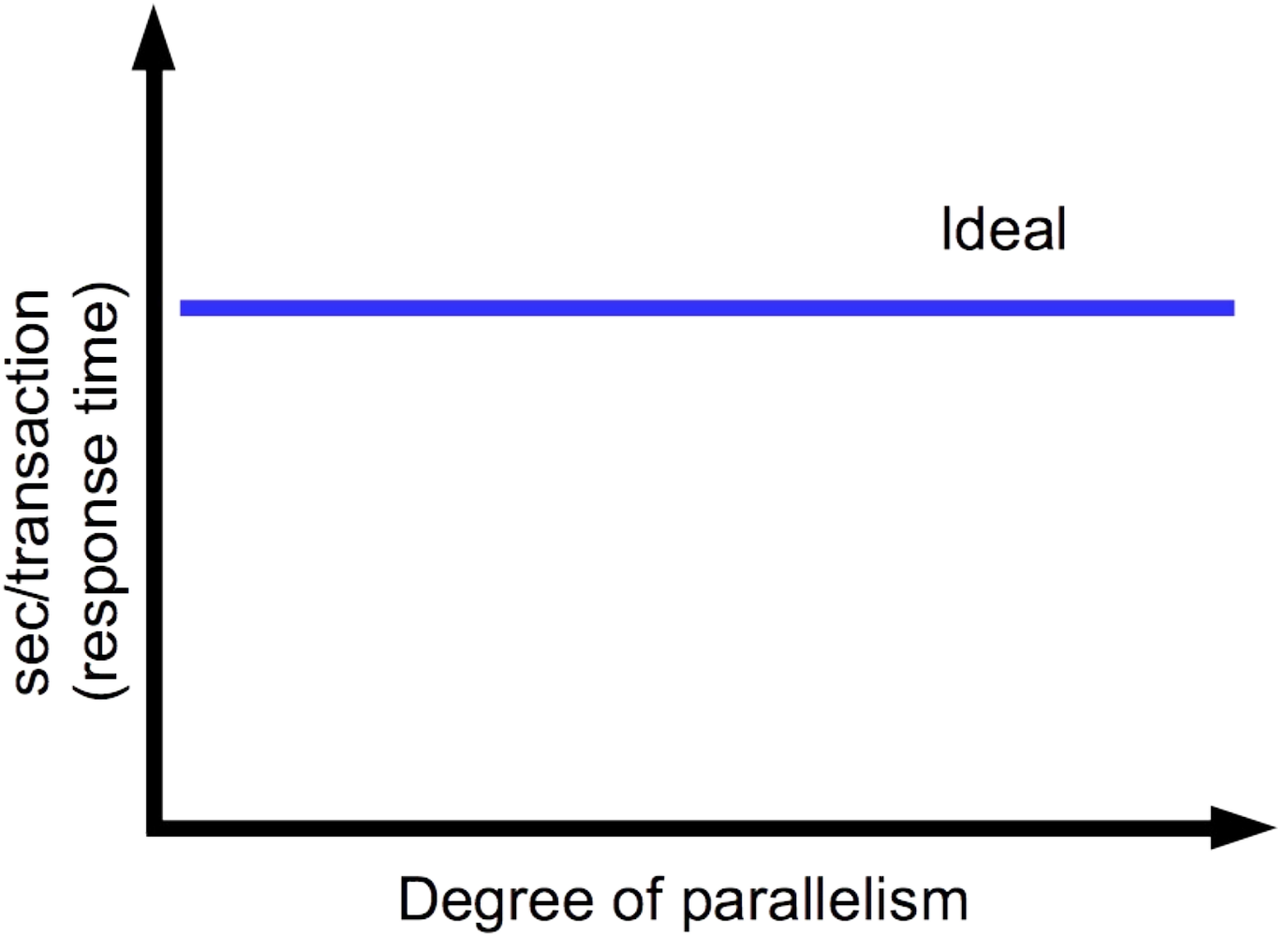


Terminology - Speed-Up



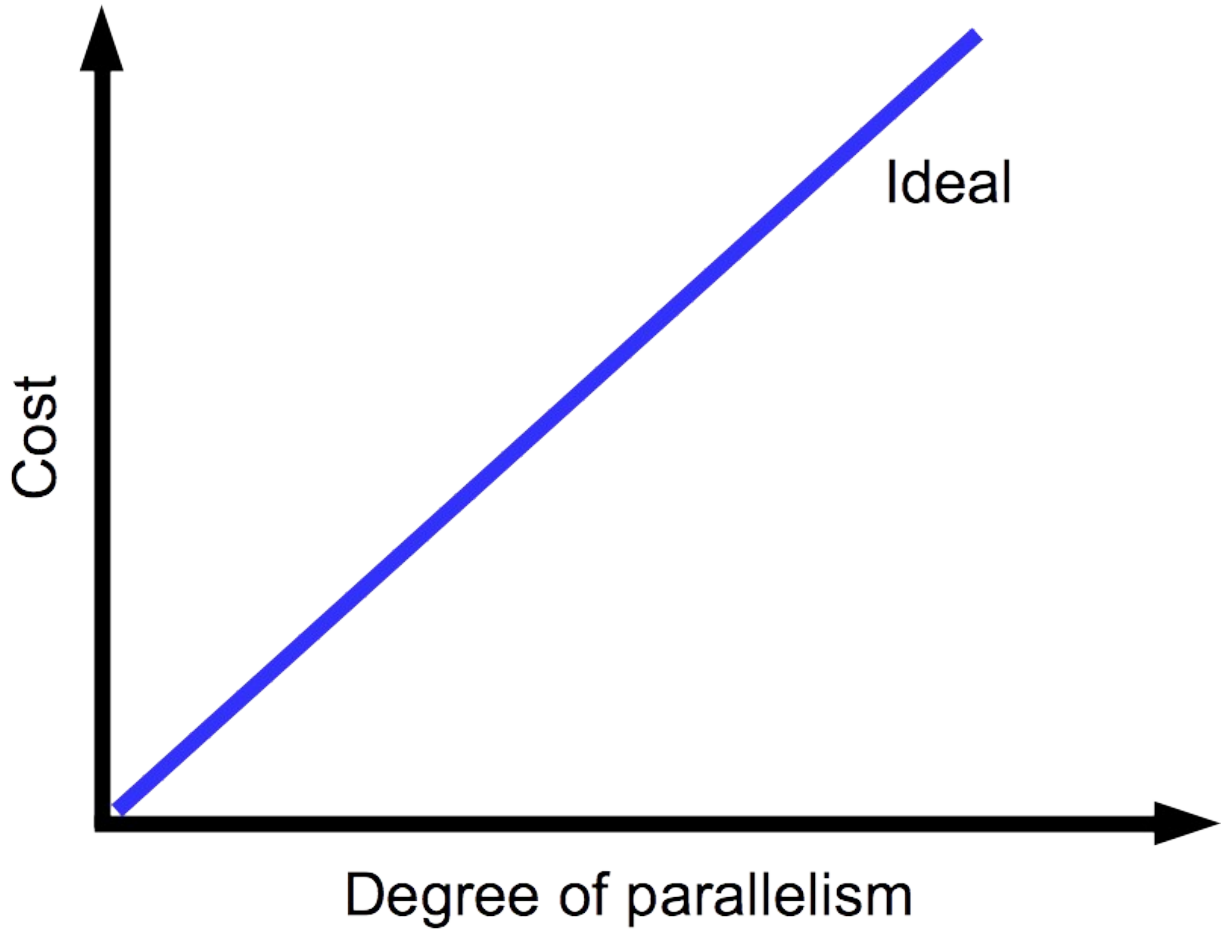
More resources means proportionally less time for given amount of data.

Terminology - Scale-Up



If resources increased in proportion to increase in data size, time is constant.

Also: proportional cost

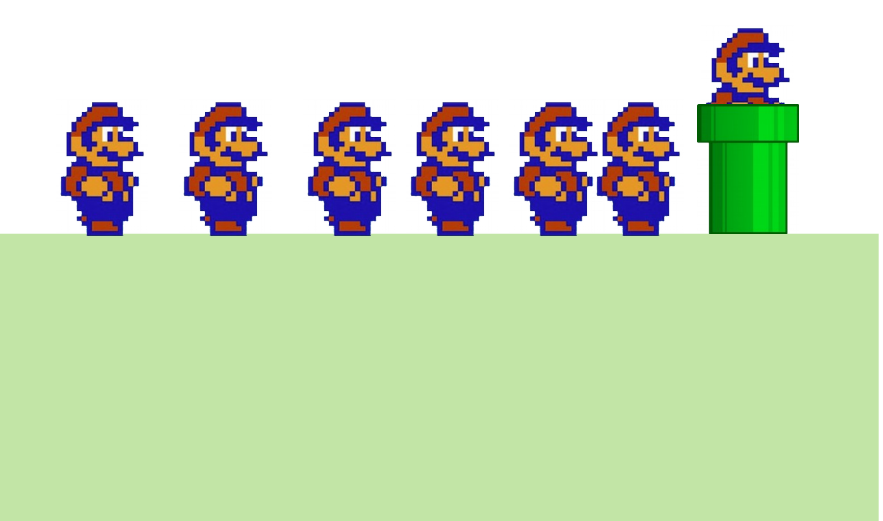


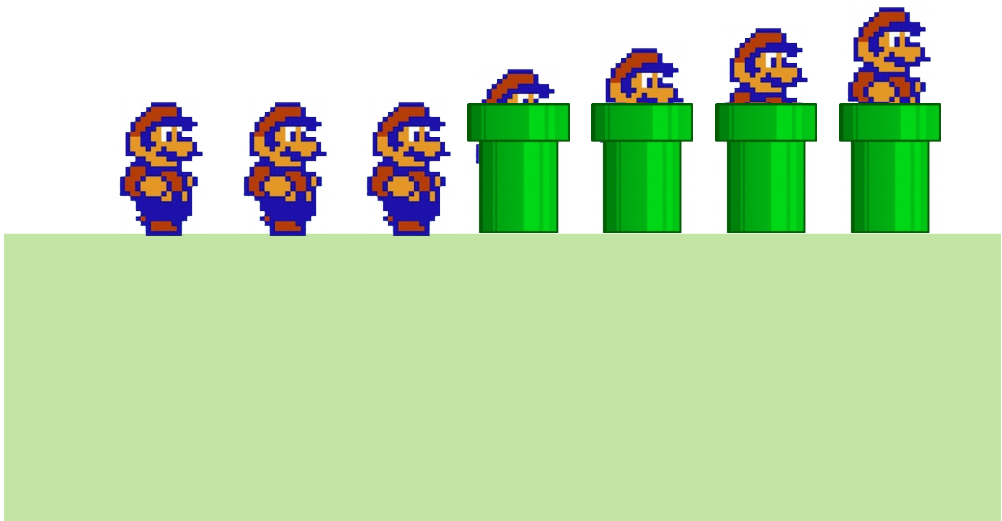
Infrastructures cost should remain proportional as number of CPUs grow.

Parallel Databases

Parallelism

- More processors -> Better Throughput
- Divide big problems into smaller ones





DBMS are suited for parallelism

- Bulk processing of data partitions
- Natural pipelining (execution plan)
- Users don’t need to write parallel queries

Parallelism over time

- Before: big parallel computers
- Now: small multicore servers organized in clusters

Levels of sharing

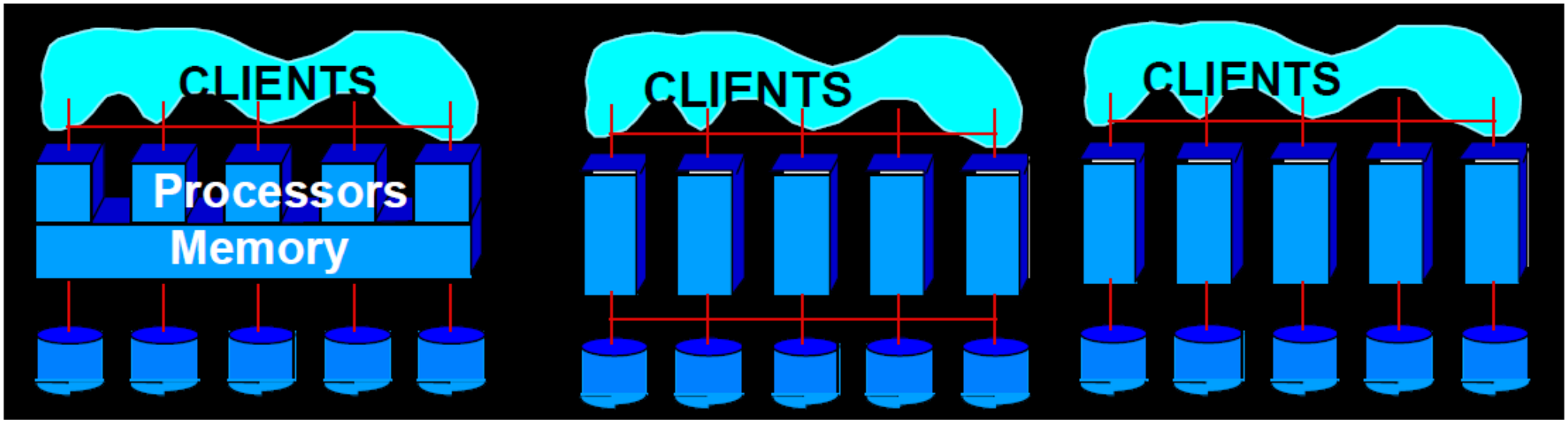
- Shared memory
- Shared disk
- Shared nothing (network)

Architecture Issue: Shared What?

Shared Memory

Shared Disk

Shared Nothing (network)



- Easy to program
- Expensive to build
- Difficult to scale up

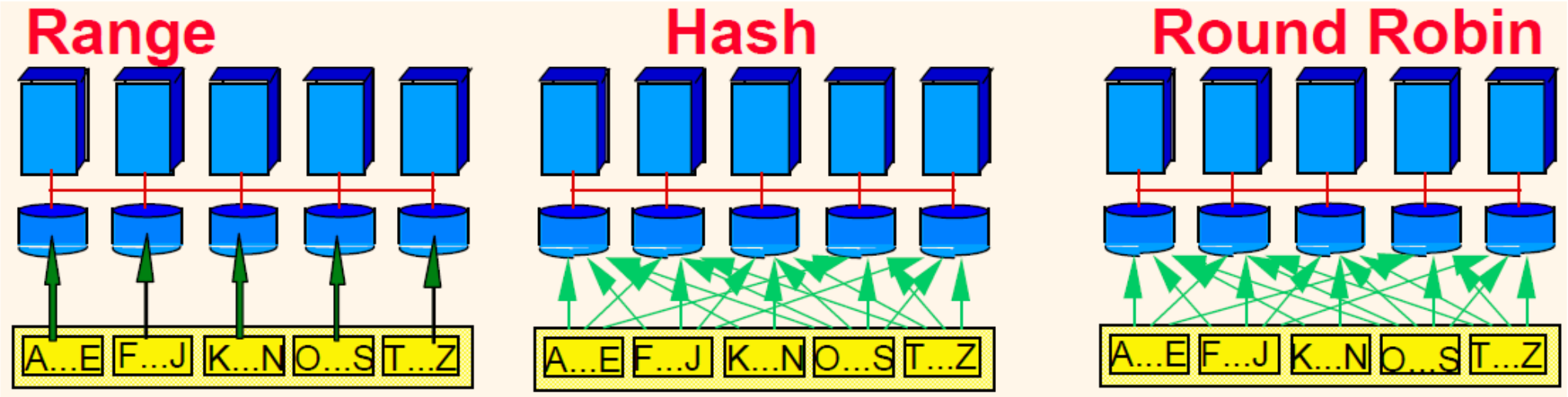
- Hard to program
- Cheap to build
- Easy to scale up

Types of DBMS parallelism

- Intra-operator parallelism
  - get all machines working to compute a given operation (scan, sort, join)
- Inter-operator parallelism
  - each operator may run concurrently on a different site (exploits pipelining)
- Inter-query parallelism
  - different queries run on different sites

Automatic Data Partitioning

Partitioning a table:



Good for equijoins, range queries, group-by

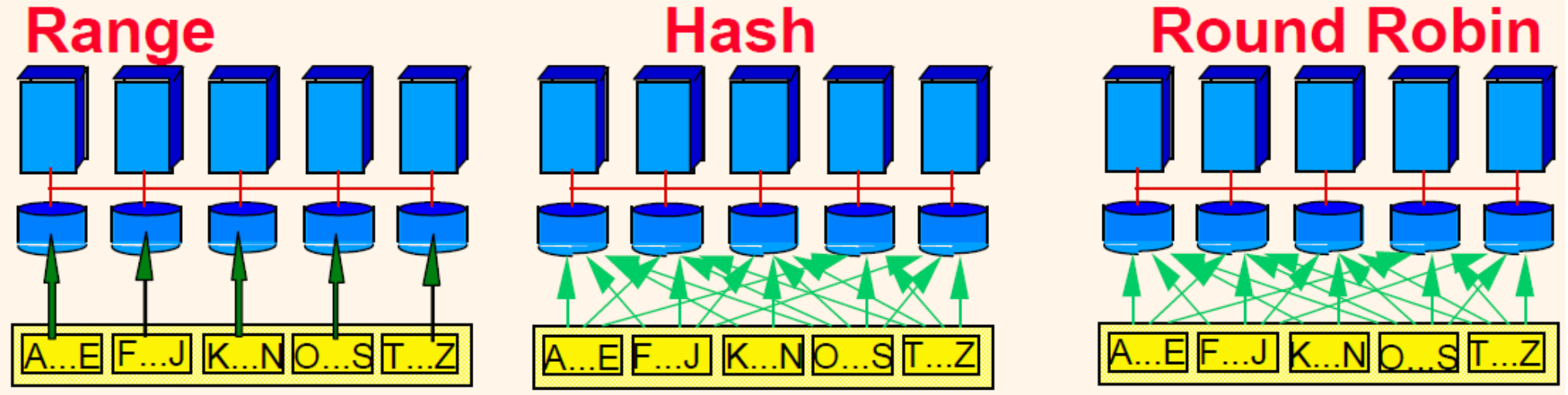
Good for equijoins

Good to spread load

Shared disk and memory less sensitive to partitioning,
Shared nothing benefits from "good" partitioning

Exercício 2

Ordene os tipos de técnica de particionamento de dados (Range, Hash, Round Robin) de acordo com o tamanho físico dos índices que precisam ser mantidos para localizar o disco ou CPU que contém cada tupla. Justifique sua resposta.



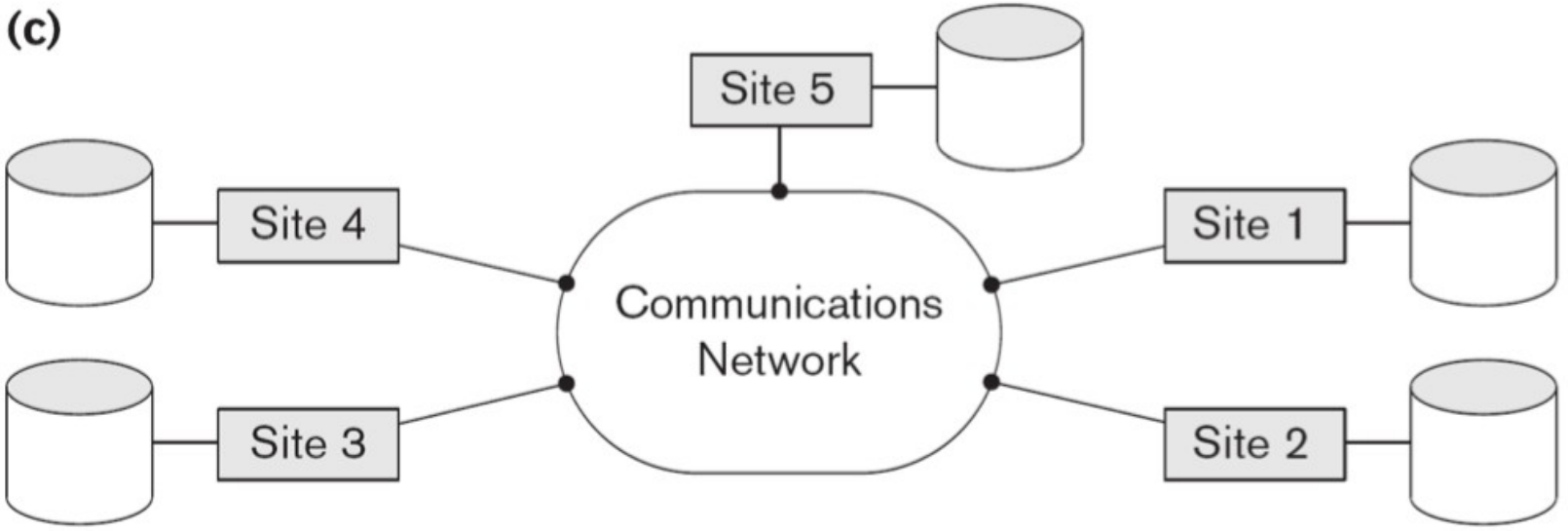
Distributed Databases

Definition

- A **transaction** can be executed by multiple networked computers in a unified manner.
- A **distributed database** (DDB) is a collection of multiple logically related database **distributed over a computer network**
- **A distributed database management system** (DDBMS) is a software system that manages a distributed database while making the distribution **transparent** to the user.

Distributed Database System

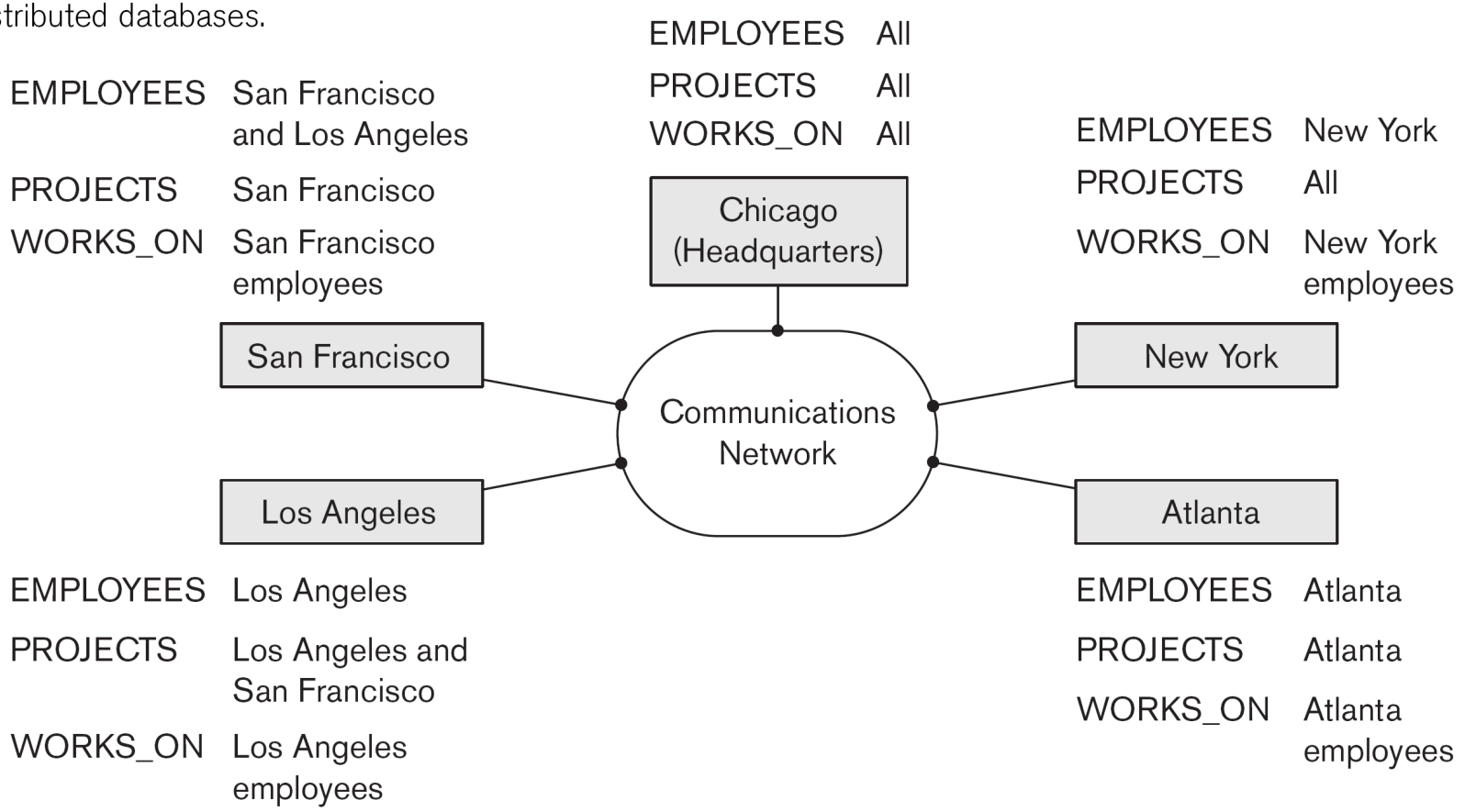
- Management of distributed data with different levels of transparency:
  - This refers to the physical placement of data (files, relations, etc.) which is not known to the user (distribution transparency).



Transparency

The EMPLOYEE, PROJECT, and WORKS_ON tables may be fragmented horizontally and stored with possible replication as shown below.

Figure 25.1
Data distribution and replication among distributed databases.



Advantages (transparency, contd.)

- Distribution and Network transparency:
  - Users do not have to worry about operational details of the network.
  - There is **Location transparency**, which refers to freedom of issuing command from any location without affecting its working.
  - Then there is **Naming transparency**, which allows access to any names object (files, relations, etc.) from any location.

Advantages (transparency, contd.)

- Replication transparency:
  - It allows to store copies of a data at multiple sites.
  - This is done to minimize access time to the required data.
- Fragmentation transparency:
  - Allows to fragment a relation horizontally (create a subset of tuples of a relation) or vertically (create a subset of columns of a relation).

Advantages (transparency, contd.)

- Increased reliability and availability:
  - **Reliability** refers to **system live time**, that is, system is running efficiently most of the time. Reliability is often characterized in terms of mean time between failures (MTBF).
  - **Availability** is the **probability that the system is continuously available** during a time interval. Availability is given as a percentage of the time a system is expected to be available, e.g., 99.999 percent ("five nines").
- A distributed database system has multiple nodes (computers) and if one fails then others are available to do the job.

Advantages (transparency, contd.)

- Improved performance:
  - A distributed DBMS fragments the database to **keep data closer to where it is needed** most.
  - This reduces data management (access and modification) time significantly.
- Easier expansion (scalability):
  - Allows new nodes (computers) to be added anytime without changing the entire configuration.

Data Fragmentation, Replication and Allocation

- Data Fragmentation
  - Split a relation into logically related and correct parts. A relation can be fragmented in two ways:
- Horizontal Fragmentation
- Vertical Fragmentation

Horizontal fragmentation

- It is a horizontal subset of a relation which contain those of tuples which satisfy selection conditions.
- Consider the Employee relation with selection condition (DNO = 5). All tuples satisfy this condition will create a subset which will be a horizontal fragment of Employee relation.
- A selection condition may be composed of several conditions connected by AND or OR.

Horizontal fragmentation

- **Complete relation:**

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
2	Kmart	Ottawa	671.05
3	Eatons	Toronto	301.00
4	The Bay	Ottawa	162.99

- **Horizontally fragmented relation (two sites):**

Site 1 (Ottawa site)

Vno	Vname	City	Vbal
2	Kmart	Ottawa	671.05
4	The Bay	Ottawa	162.99

Site 2 (Toronto site)

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
3	Eatons	Toronto	301.00

Vertical fragmentation

- It is a subset of a relation which is created by a subset of columns. Thus a vertical fragment of a relation will contain values of selected columns.
- Consider the Employee relation. A vertical fragment of can be created by keeping the values of Name, Bdate, Sex, and Address.
- Because there is no condition for creating a vertical fragment, each fragment must include the primary key attribute of the parent relation Employee.

Vertical fragmentation

- Complete relation:

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
2	Kmart	Ottawa	671.05
3	Eatons	Toronto	301.00
4	The Bay	Ottawa	162.99

- Vertically fragmented relation (two sites):

Site 1

Vno	Vbal
1	200.00
2	671.05
3	301.00
4	162.99

Site 2

Vno	Vname	City
1	Sears	Toronto
2	Kmart	Ottawa
3	Eatons	Toronto
4	The Bay	Ottawa

Representation - Horizontal fragmentation

- Each horizontal fragment on a relation can be specified by a σ_{Ci} (R) operation in the relational algebra.
- Complete horizontal fragmentation: A set of horizontal fragments whose conditions C1, C2, …, Cn include all the tuples in R- that is, every tuple in R satisfies (C1 OR C2 OR … OR Cn).
- Disjoint complete horizontal fragmentation: No tuple in R satisfies (Ci AND Cj) where $i \neq j$.

Representation - Vertical fragmentation

- A vertical fragment on a relation can be specified by a ΠLi(R) operation in the relational algebra.
- Complete vertical fragmentation: A set of vertical fragments whose projection lists L1, L2, …, Ln include all the attributes in R but share only the primary key of R. In this case the projection lists satisfy the following two conditions:
- L1 U L2 U ... U Ln = ATTRS (R)
- Li ∩ Lj = PK(R) for any i j, where ATTRS (R) is the set of attributes of R and PK(R) is the primary key of R.

Data Fragmentation, Replication and Allocation

- Fragmentation schema
  - A definition of a set of fragments (horizontal or vertical or horizontal and vertical) that includes all attributes and tuples in the database that satisfies the condition that the whole database can be reconstructed from the fragments.
- Allocation schema
  - It describes the distribution of fragments to sites of distributed databases. It can be fully or partially replicated or can be partitioned.

Replication and Allocation

- Data Replication
  - In full replication the entire database is replicated and in partial replication some selected part is replicated to some of the sites.
  - Data replication is achieved through a replication schema.
- Data Distribution (Data Allocation)
  - This is relevant only in the case of partial replication or partition.
  - The selected portion of the database is distributed to the database sites.

Exercício 3

- Considere a relação R(a,b,c). Quais operações da álgebra relacional são necessárias para recompor a tabela em caso de fragmentação horizontal? E para fragmentação vertical?

Vertical

Horizontal

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
2	Kmart	Ottawa	671.05
3	Eatons	Toronto	301.00
4	The Bay	Ottawa	162.99

Concurrency Control and Recovery

- Dealing with multiple copies of data items
- Failure of individual sites
- Communication link failure
- Distributed commit
- Distributed deadlock

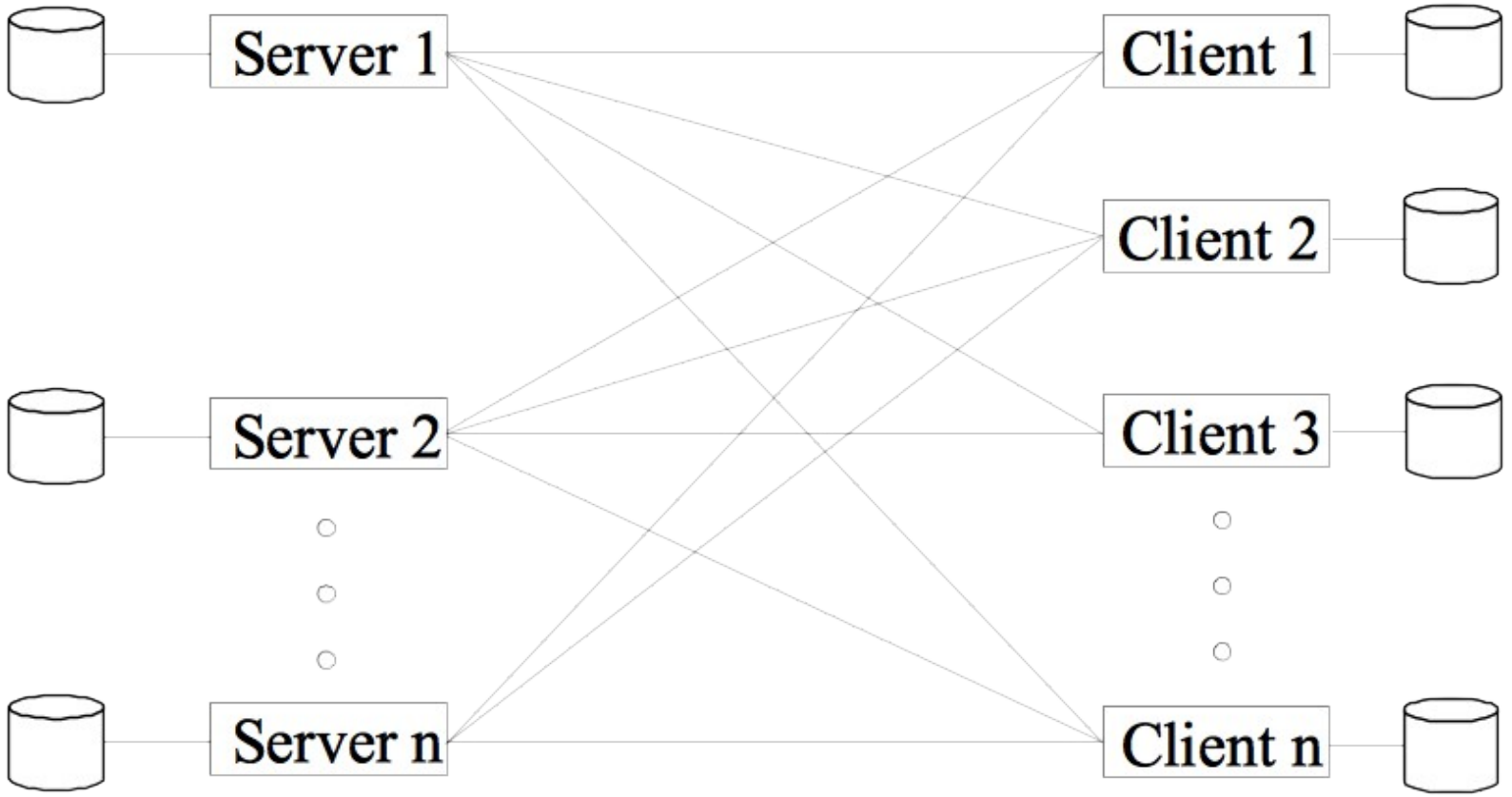
Parallel vs distributed servers

- parallel database server:
  - servers in physical proximity to each other
  - fast, high-bandwidth communication between servers, usually via a LAN
  - most queries processed cooperatively by all servers
- distributed database server:
  - servers may be widely separated
  - server-to-server communication may be slower, possibly via a WAN
  - queries often processed by a single server

Client-Server Database Architecture

Client-Server DB Architecture

- It consists of clients running client software, a set of servers which provide all database functionalities and a reliable communication infrastructure.
- 3-Tier Architecture



Client-Server DB Architecture

- Clients reach server for desired service, but server does reach clients.
- The server software is responsible for local data management at a site, much like centralized DBMS software.
- The client software is responsible for most of the distribution function.

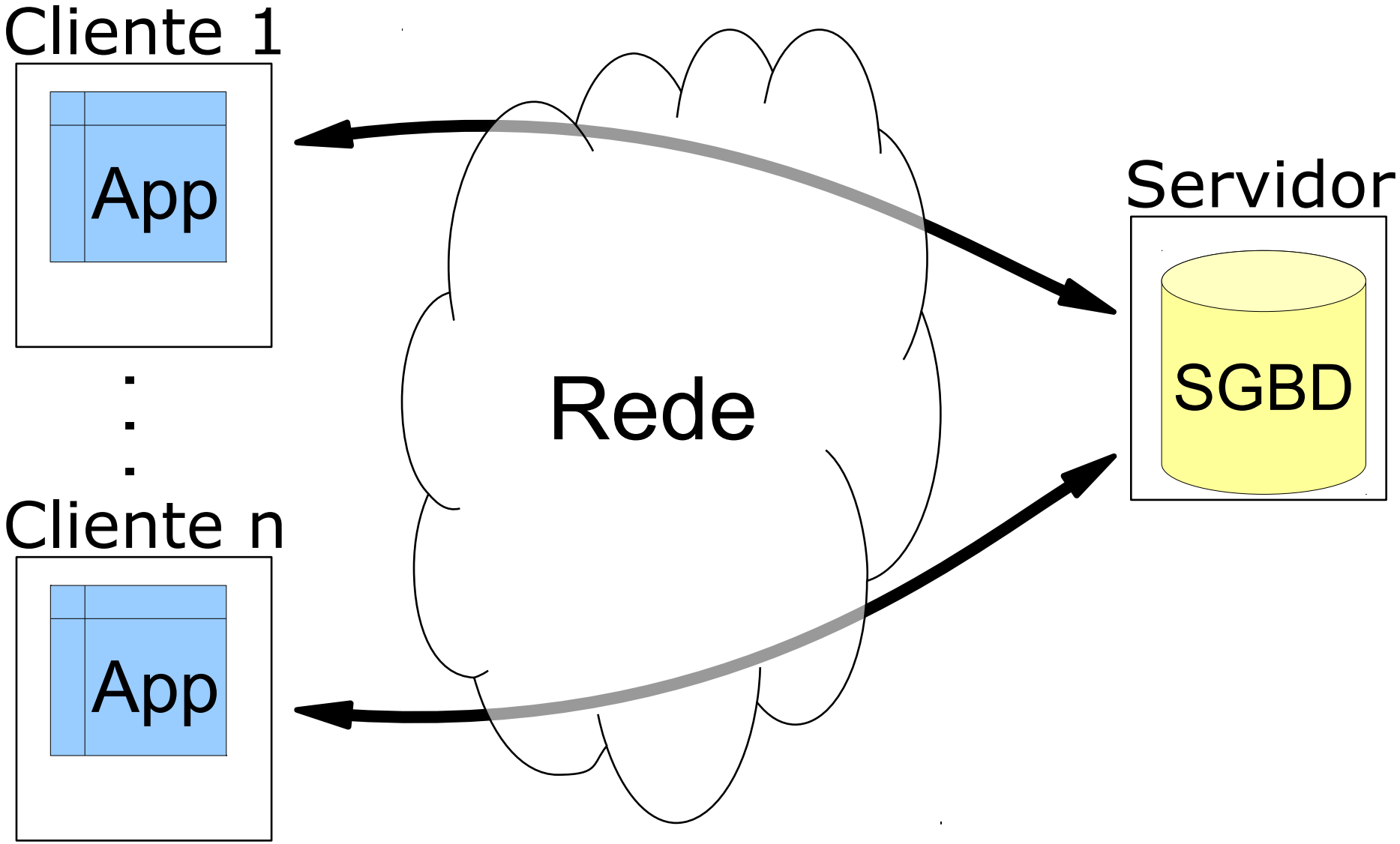
Processing of SQL queries

- Client parses a user query and decomposes it into a number of independent sub-queries. Each subquery is sent to appropriate site for execution.
- Each server processes its query and sends the result to the client.
- The client combines the results of subqueries and produces the final result.

Arquitetura Cliente-Servidor

- Usada na maioria das instituições
- Usuário acessa a aplicação por um dispositivo Cliente (desktop, laptop, celular...)
- Aplicação envia consultas para obter dados do SGBD (Servidor)
- SGBD processa consulta e retorna dados para serem exibidos no Cliente
- Exemplos: Folha de pagamentos, iTunes

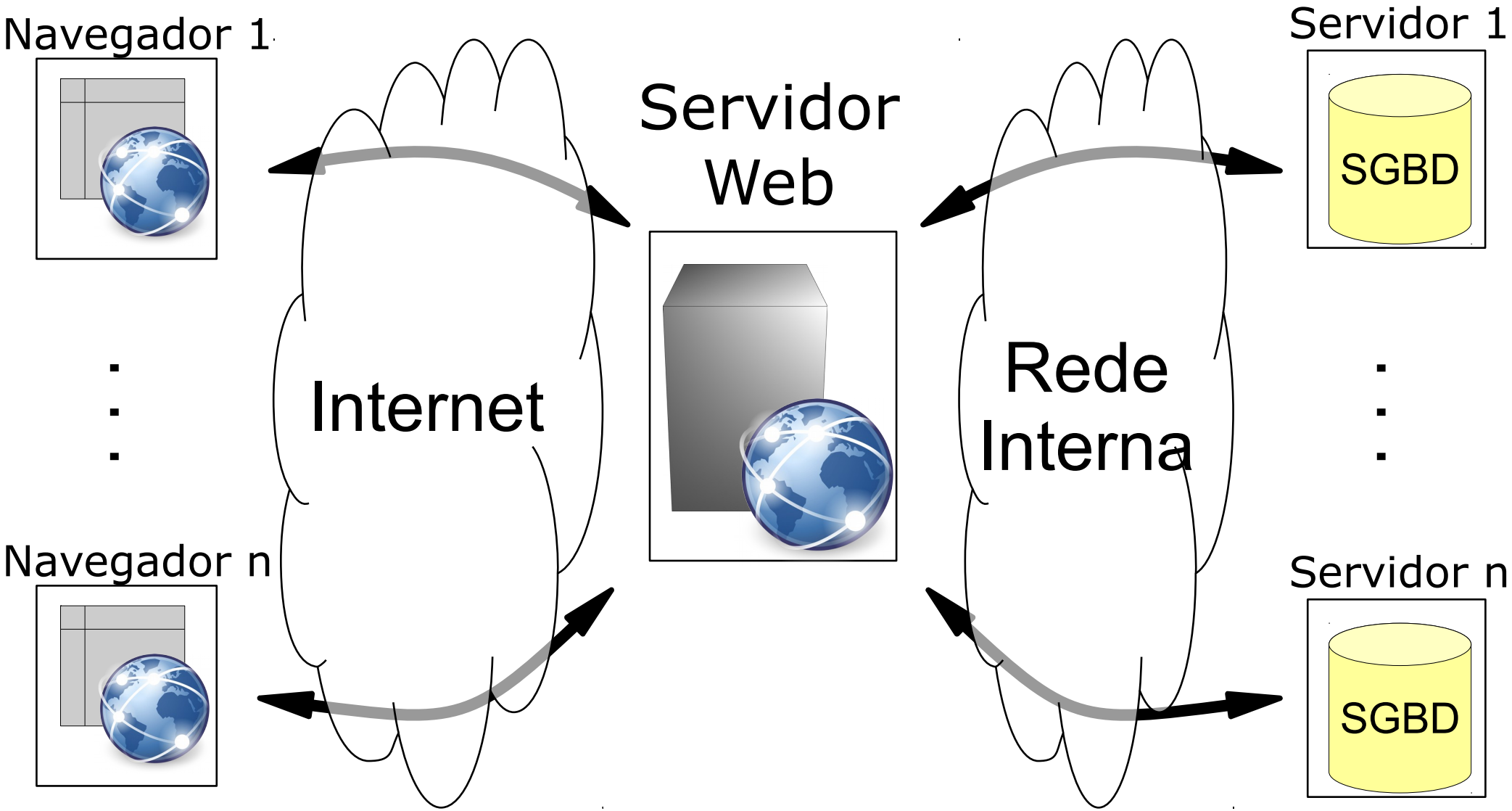
Arquitetura Cliente-Servidor



Arquitetura Web 1.0

- Usada na maioria dos sites “normais”
- Usuário usa o navegador para requisitar páginas para um Servidor Web
- Servidor Web envia consultas a um ou mais SGBDs para obter dados e montar a página
- Exemplos: bancos online, sites de empresas
- Muitas apps e sites como Facebook, Google precisam de arquiteturas mais complexas. Veremos estes casos no fim do curso.

Arquitetura Web 1.0



Exemplo: Facebook

1. Usuário abre o navegador e entra em facebook.com
2. Servidor Web do facebook recebe a requisição do usuário
3. Servidor Web do facebook obtém dados do mural de um SGBD interno
4. Servidor Web do facebook obtém dados de propaganda de um outro SGBD interno
5. Servidor Web do facebook monta a página e envia para o navegador exibir

Alternative Database Architectures

- In-Memory Databases
- SSD Databases
- GPU Databases
- Crowdsourced Databases

In-Memory Databases

- Becoming popular as RAM prices drop
- Offered by main vendors (MySQL offers in-memory storage engine)
- Durability (ACID) support?

In-Memory Databases - Durability

- Snapshot files: generated periodically - may lose recent information
- Transaction logging: as in RDBMS - disk may be bottleneck
- Non-Volatile DIMM: more expensive
- Non-volatile random access memory: usually RAM backed up with battery power
- Database replication

Crowdsourced Databases

- Ongoing research
- For task that are hard for computers to process
- e.g. interpreting images
- Uses crowdsourcing infrastructures such as Amazon Mechanical Turk

Crowdsourced Databases

SELECT * FROM images
WHERE isFlower(img)

TASK isFlower(Image img) RETURN BOOL:
TaskType: Question
Text: ``Does this image: <img src=`%s'>
contain a flower?'',URLify(img)
Response: Choice(``YES'',``NO'')

Referências