TOR VERGATA
UNIVERSITÀ DEGLI STUDI DI ROMA

Macroarea di Ingegneria
Dipartimento di Ingegneria Civile e Ingegneria Informatica

LEZ. 13/11/2024 part 2

Elective exercise using Go and RPC

Corso di Sistemi Distribuiti e Cloud Computing
A.A. 2024/25

Valeria Cardellini

Laurea Magistrale in Ingegneria Informatica

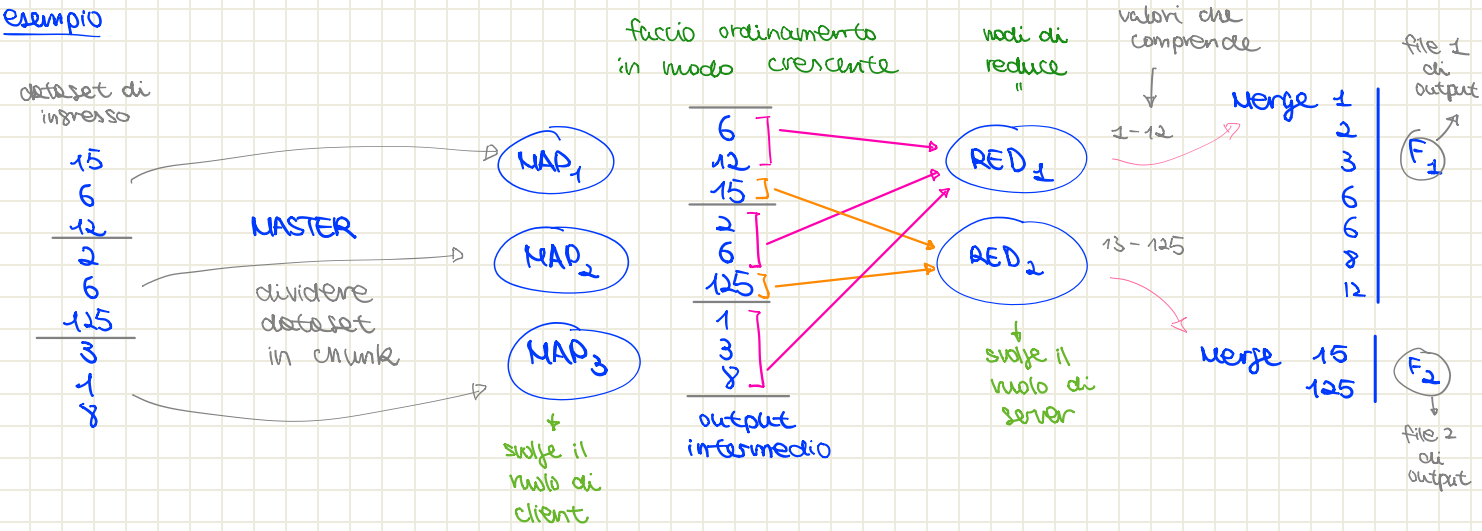
Elective exercise using Go and RPC

- Realize a distributed application that solves the sorting problem using MapReduce

> *The beauty of MapReduce is that any programmer can understand it, and its power comes from being able to harness thousands of computers behind that simple interface.*
>
> David Patterson

- Requirements:
  - Use either Go and RPC or Go and gRPC
  - Organize properly your code into separate files
  - 1 student per team (2 students only if also the optional part is implemented)

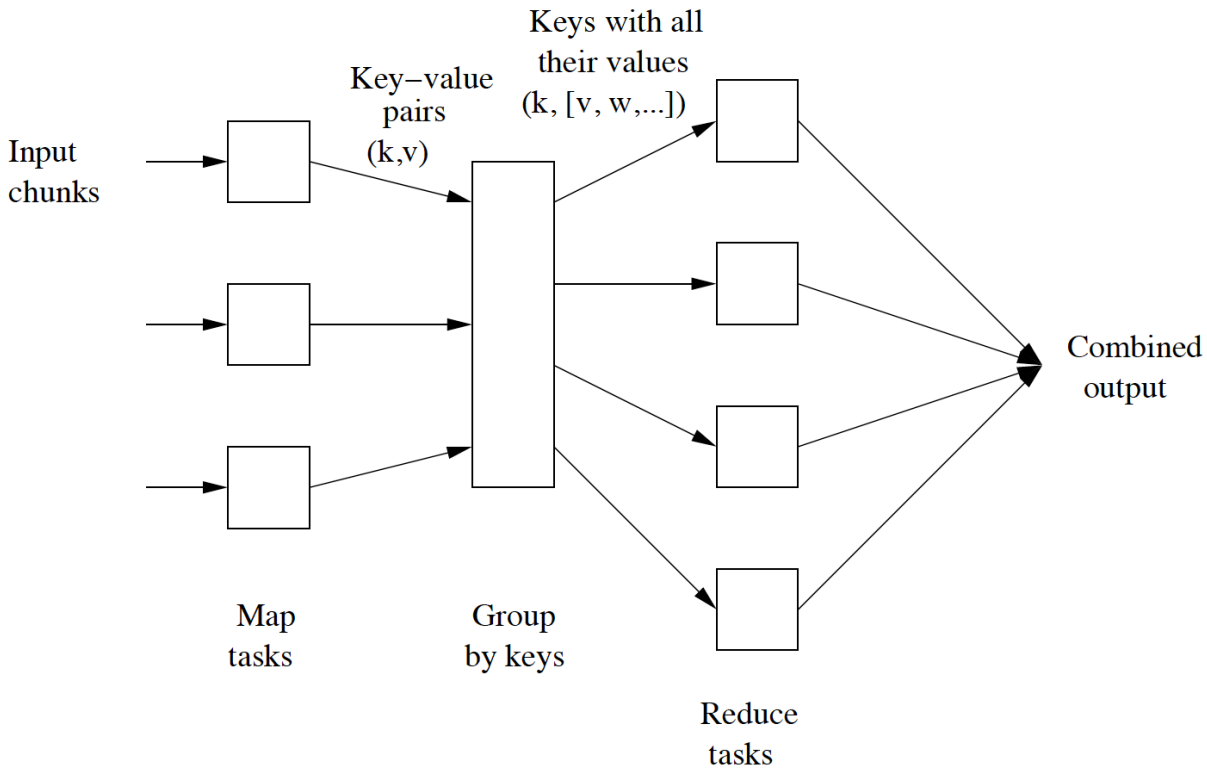
esempio



reduce → richiamato Terasor

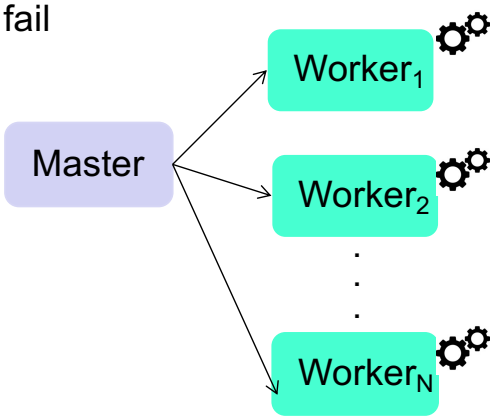
salva il risultato finale in un file

MapReduce paradigm



Overview on architecture

- Exploit master-worker architecture
  - Distribute work among workers using RPC or gRPC
- Master assigns map and reduce tasks to workers
- Workers execute map and reduce tasks
- Do not consider failures of master and workers
  - Set of workers is known and does not change during computation; no worker fails
  - Master does not fail



Overview on architecture

- 3 phases
  - **Map**: sort the chunk received from master
  - **Shuffle**: map tasks send their intermediate results to reduce by partitioning result
  - **Reduce**: merge intermediate results received from map
- Distribute work among parallel workers
- Need a synchronization point (i.e., barrier) between map and reduce tasks
  - No reduce task can start until all the map tasks have finished their processing
- Each reduce task writes its results on a file

Some simplifying assumptions

- Master and workers not fail during computation → nessun fallimento durante computaz.
- The set of workers is known to the ster when the system starts and is defined into a configuration file
- The ports are defined into a configuration file → indirizzo e porta

progetto: almeno 2 nodi di reduce (ma dovrebbe funzionare anche con un singolo nodo)

Optional

- Sample input data to optimize partitioning among reduce task
- You can containerized your distributed application
  - To build the image, see Go official image hub.docker.com/_/golang
  - A Docker container per application component and then use Docker Compose to orchestrate the multiple containers on your computer

Delivery

- When
  - By December 13, 2024
- What
  - Your code, including a README with instructions to run it
  - Optional: very short report describing the architecture of the distributed application and its main features
- How
  - By email
  - Use as mail subject: [SDCC] consegna esercizio in Go