CLOUD COMPUTING CONCEPTS

with Indranil Gupta (Indy)

MAPREDUCE

Lecture C

MAPREDUCE SCHEDULING

Programming MapReduce

Externally: For user

1. Write a Map program (short), write a Reduce program (short)
2. Submit job; wait for result
3. Need to know nothing about parallel/distributed programming!

Internally: For the Paradigm and Scheduler

1. Parallelize Map
2. Transfer data from Map to Reduce
3. Parallelize Reduce
4. Implement Storage for Map input, Map output, Reduce input, and Reduce output

(Ensure that no Reduce starts before all Maps are finished. That is, ensure the ***barrier*** between the Map phase and Reduce phase)

Inside MapReduce

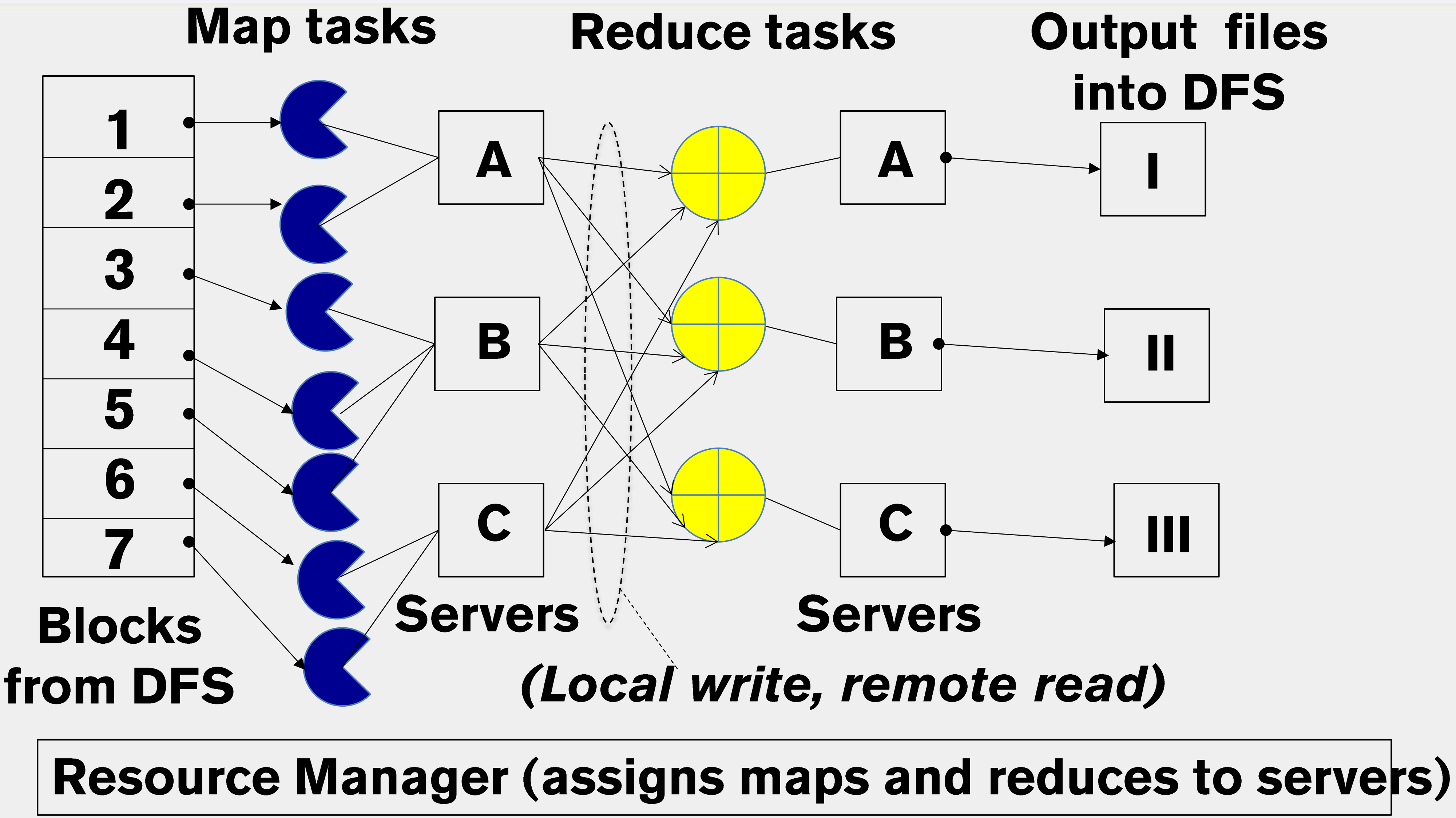
For the cloud:

1. Parallelize Map: easy! each map task is independent of the other!
   - All Map output records with same key assigned to same Reduce
2. Transfer data from Map to Reduce:
   - All Map output records with same key assigned to same Reduce task
   - Use partitioning function, e.g., hash(key)%number of reducers
3. Parallelize Reduce: easy! Each reduce task is independent of the other!
4. Implement Storage for Map input, Map output, Reduce input, and Reduce output
   - Map input: from distributed file system
   - Map output: to local disk (at Map node); uses local file system
   - Reduce input: from (multiple) remote disks; uses local file systems
   - Reduce output: to distributed file system

   local file system = Linux FS, etc.

   distributed file system = GFS (Google File System), HDFS (Hadoop Distributed File System)

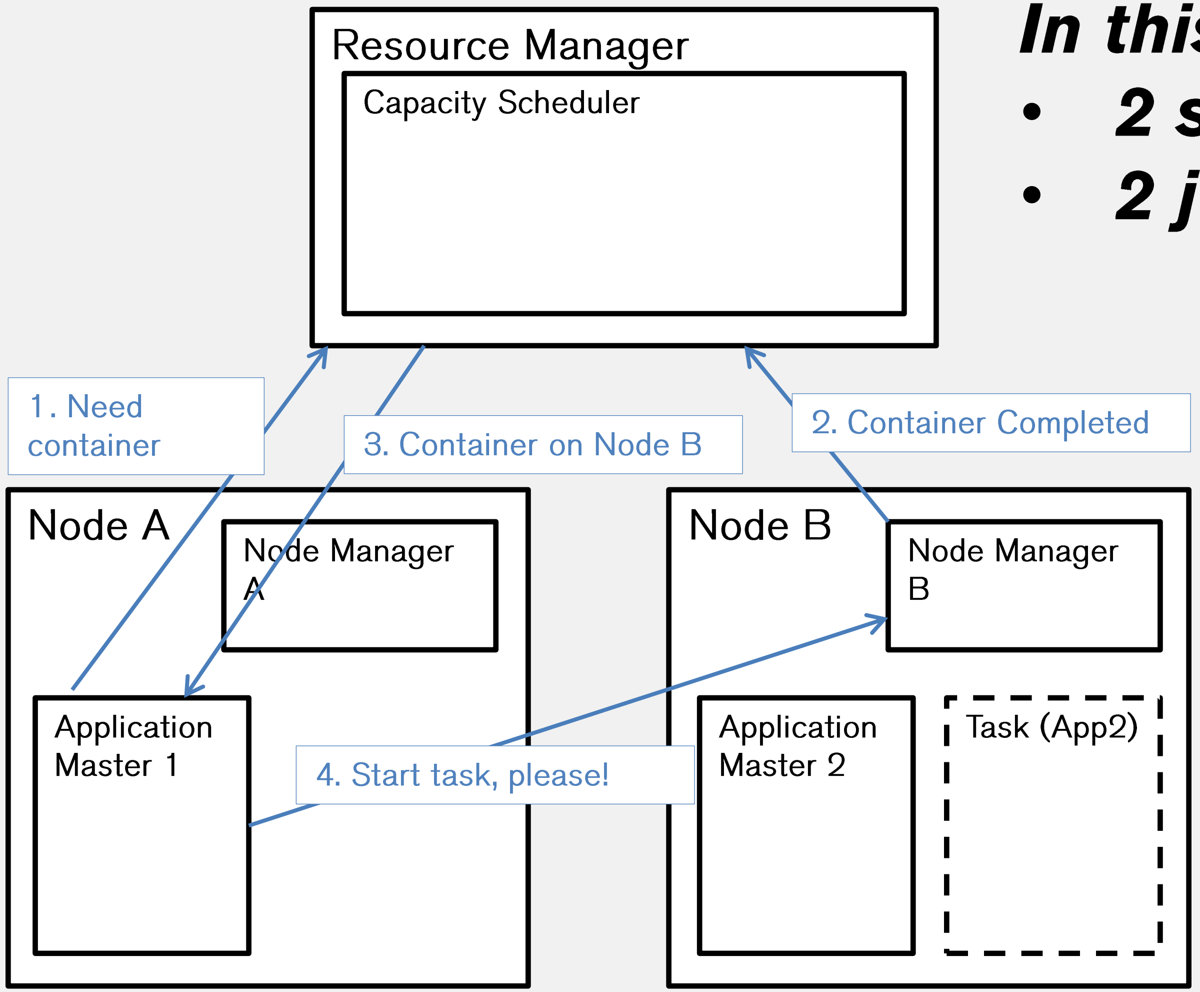
Internal Workings of MapReduce



The YARN Scheduler

- Used in Hadoop 2.x +
- YARN = Yet Another Resource Negotiator
- Treats each server as a collection of *containers*
    - Container = some CPU + some memory
- Has 3 main components
    - Global *Resource Manager (RM)*
        - Scheduling
    - Per-server *Node Manager (NM)*
        - Daemon and server-specific functions
    - Per-application (job) *Application Master (AM)*
        - Container negotiation with RM and NMs
        - Detecting task failures of that job

YARN: How a job gets a container



In this figure

- ***2 servers (A, B)***
- ***2 jobs (1, 2)***