Intro to some Advanced Topics

15-213 / 18-213: Introduction to Computer Systems
27th Lecture, Dec. 6, 2011

Instructors:

Dave O'Hallaron, Greg Ganger, and Greg Kesden

Today

- **Parallel computing building blocks**
  - Map-reduce programming
- **Virtual Machines**
- **Cloud Computing**

Parallel programming building blocks

- **Not usually done fully "by hand"**
  - Major parallel programming exploits building blocks
  - For programming efficiency and portability
- **Example: OpenMP**
  - API and framework for parallel execution
  - for "shared memory" parallel programming
    - such as many-core systems
- **Example: MPI (Message Passing Interface)**
  - API and middleware for multi-machine parallel execution
- **Example: OpenGL**
  - API and framework for high-performance graphics
  - includes mapping to popular graphics accelerators and "GPUs"
- **Example: Map-Reduce…**

Map-Reduce Programming

- **Easy-to-use API for data-parallel programs**
  - "data-parallel" means that different data processed in parallel
    - by the same sub-program
  - partial results can then be combined
- **Programmer writes two functions**
  - Map(k1, v1): outputs a list of [k2, v2] pairs
    - common (but not required) for map functions to filter the input
  - Reduce(k2, list of v2 values): outputs a list of values (call it v3)
- **Easy to make parallel**
  - Map instances can execute in any order
  - Reduce instances can execute in any order (after all maps finish)
- **Described by a 2004 Google paper**
  - Used extensively by Google, Facebook, Twitter, etc.
  - Most use the open source (Apache) implementation called Hadoop

M-R Example: word frequency in web pages

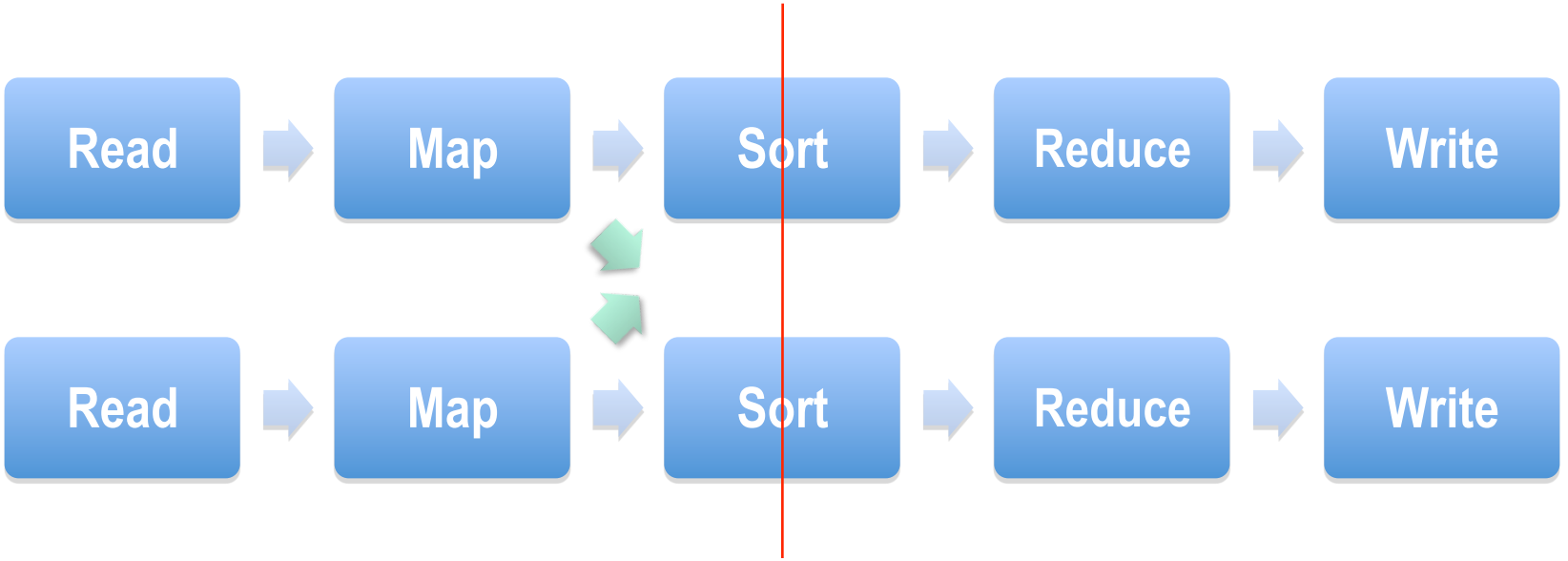
```
void map(String name, String document):
  // name: document name
  // document: document contents
  for each word w in document:
     EmitIntermediate(w, "1");
```

- **Input and output Strings**
  - Java pseudo-code here
- **Map breaks out each word**
- **Reduce counts occurrences**
  - Iterator provides the value list

```
void reduce(String word, Iterator partialCounts):
  // word: a word
  // partialCounts: a list of aggregated partial counts
   int sum = 0;
   for each pc in partialCounts:
      sum += ParseInt(pc);
   Emit(word, AsString(sum));
```

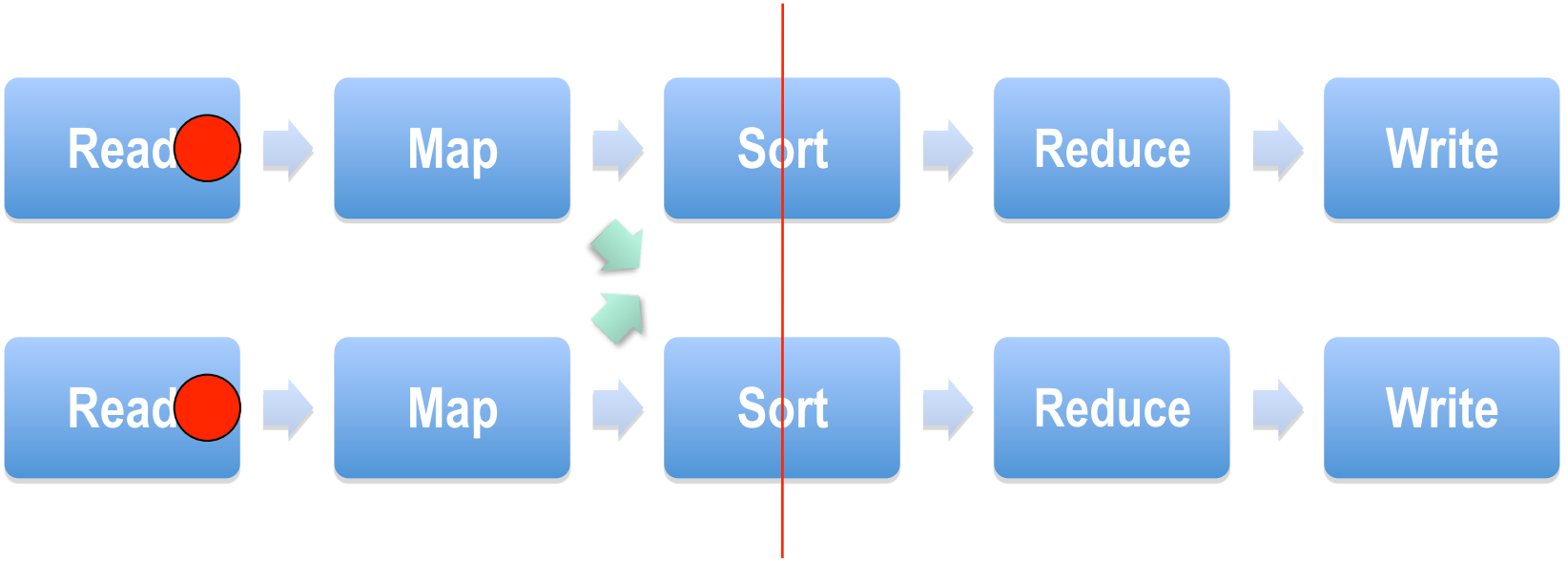
Visual of a map-reduce dataflow

Phase 1: read, map and shuffle data



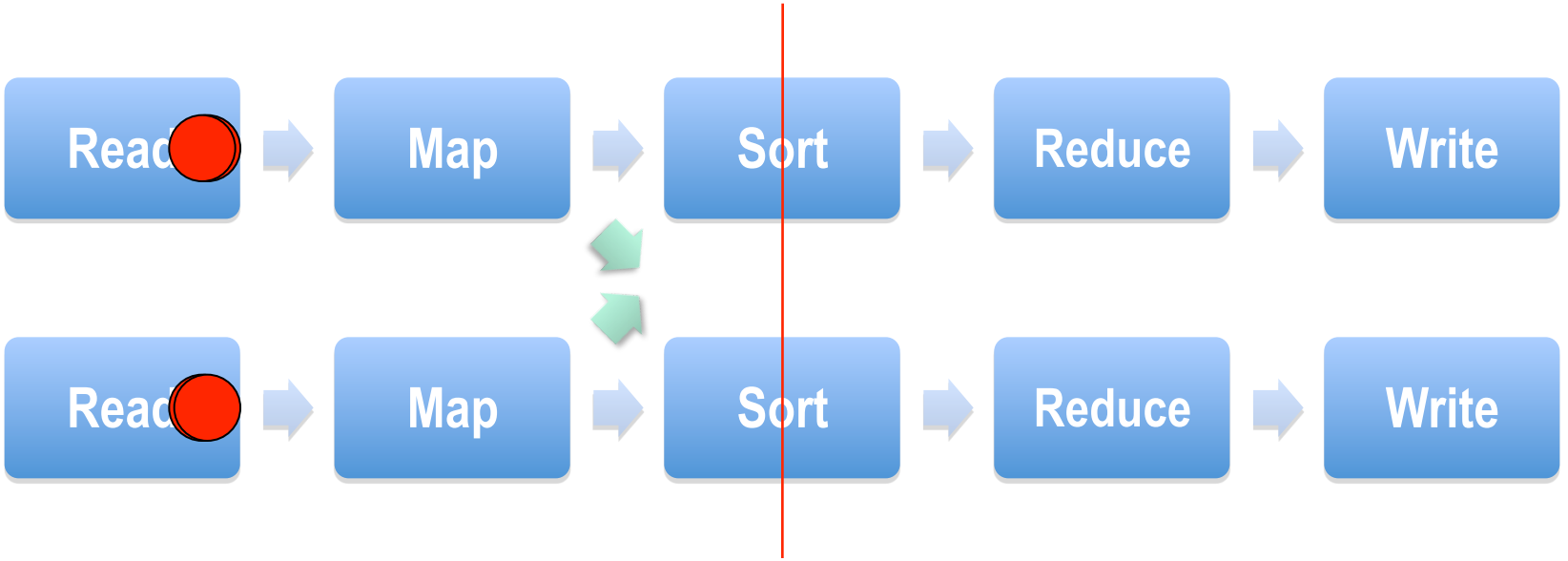
Visual of a map-reduce dataflow

Phase 1: read, map and shuffle data



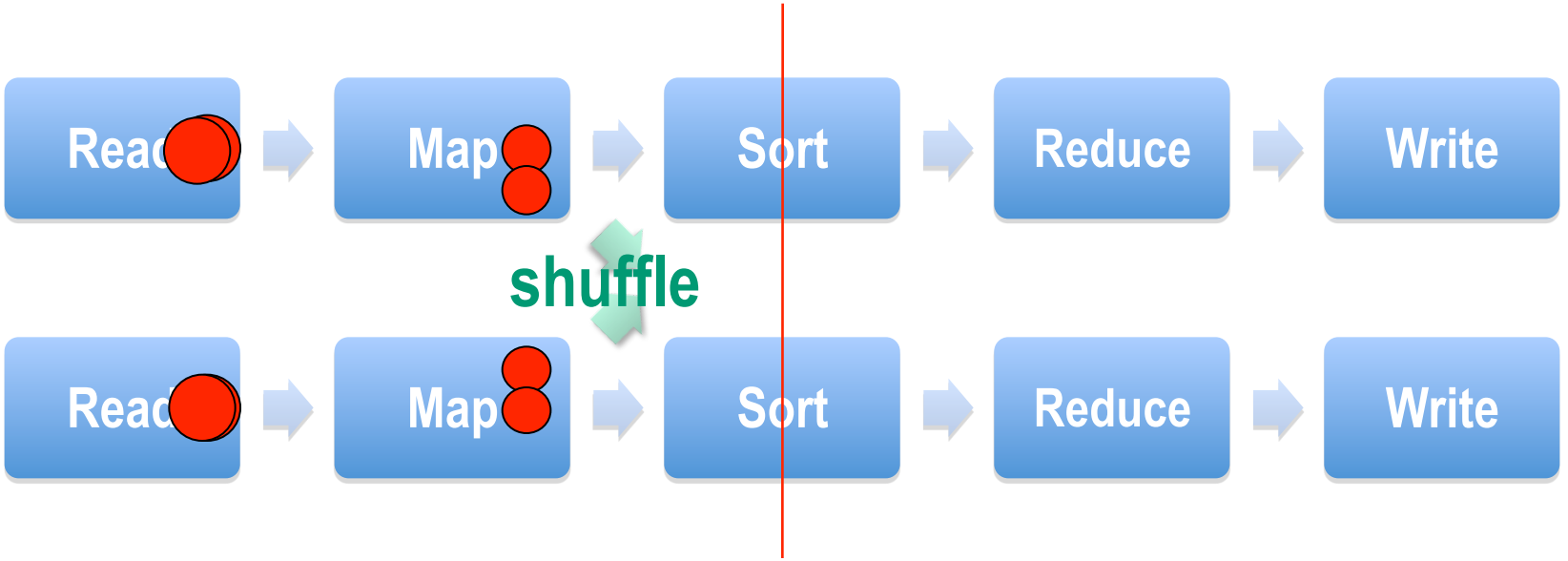
Visual of a map-reduce dataflow

Phase 1: read, map and shuffle data



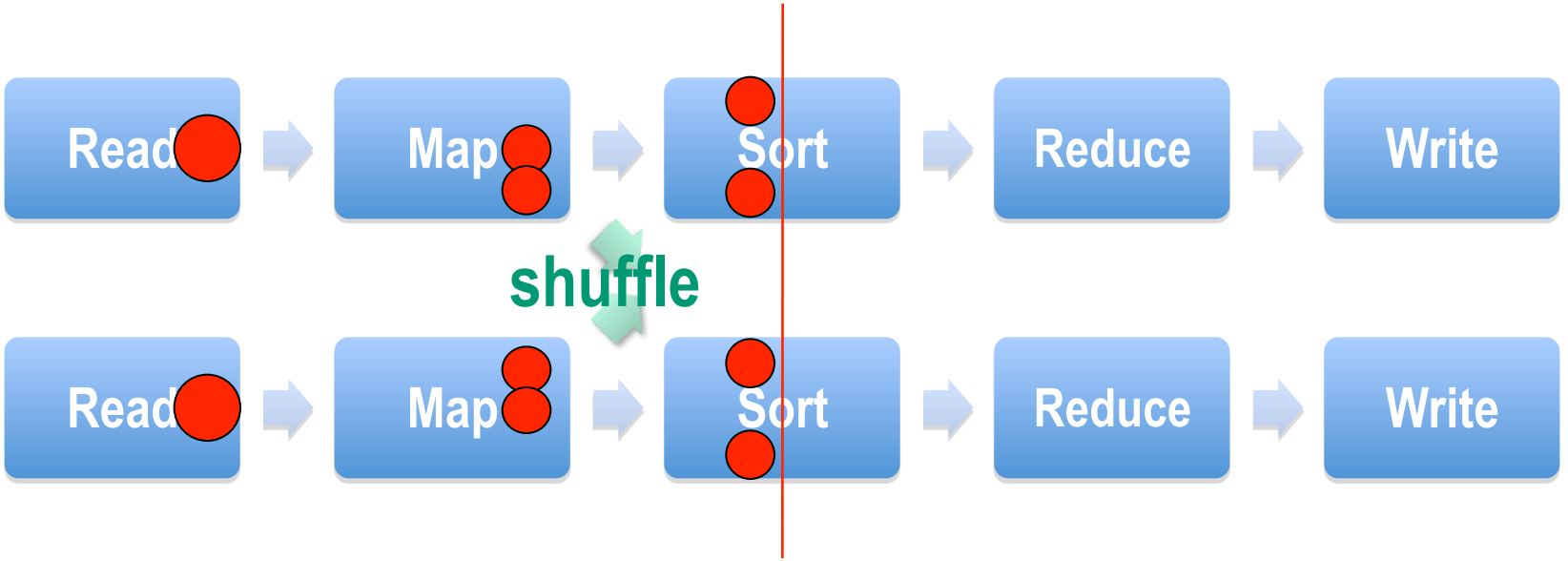
Visual of a map-reduce dataflow

Phase 1: read, map and shuffle data



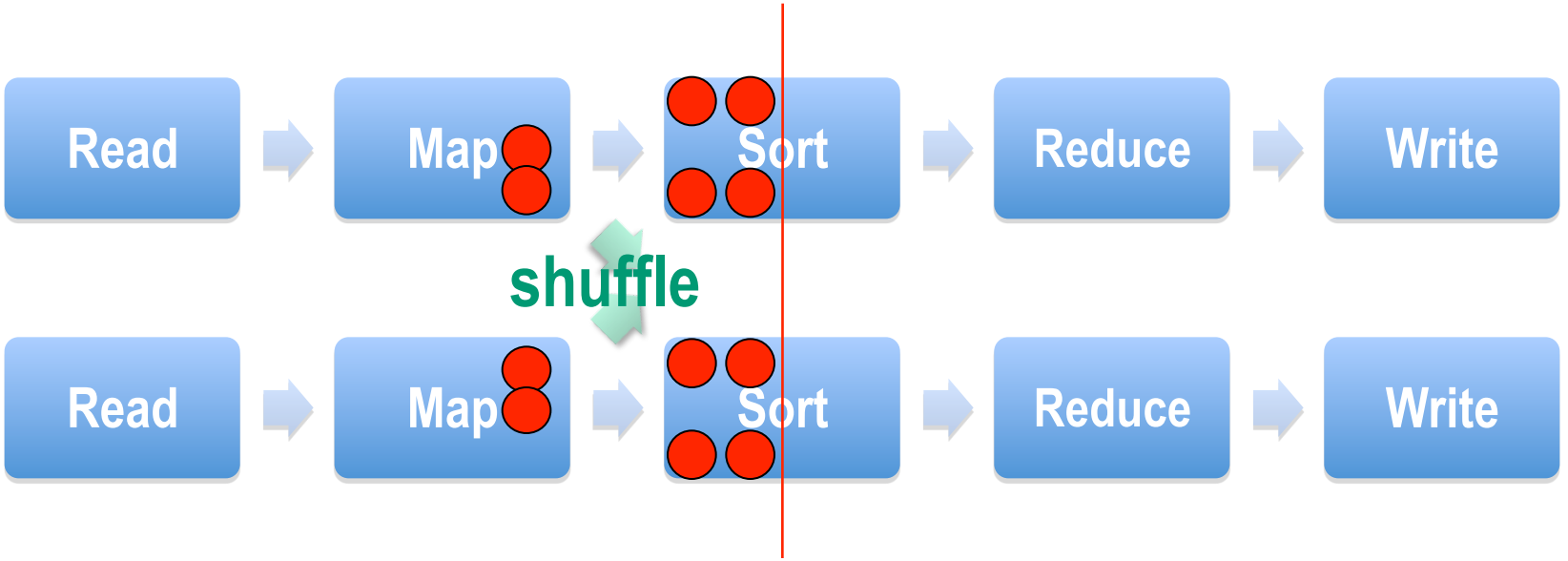
Visual of a map-reduce dataflow

Phase 1: read, map and shuffle data



Visual of a map-reduce dataflow

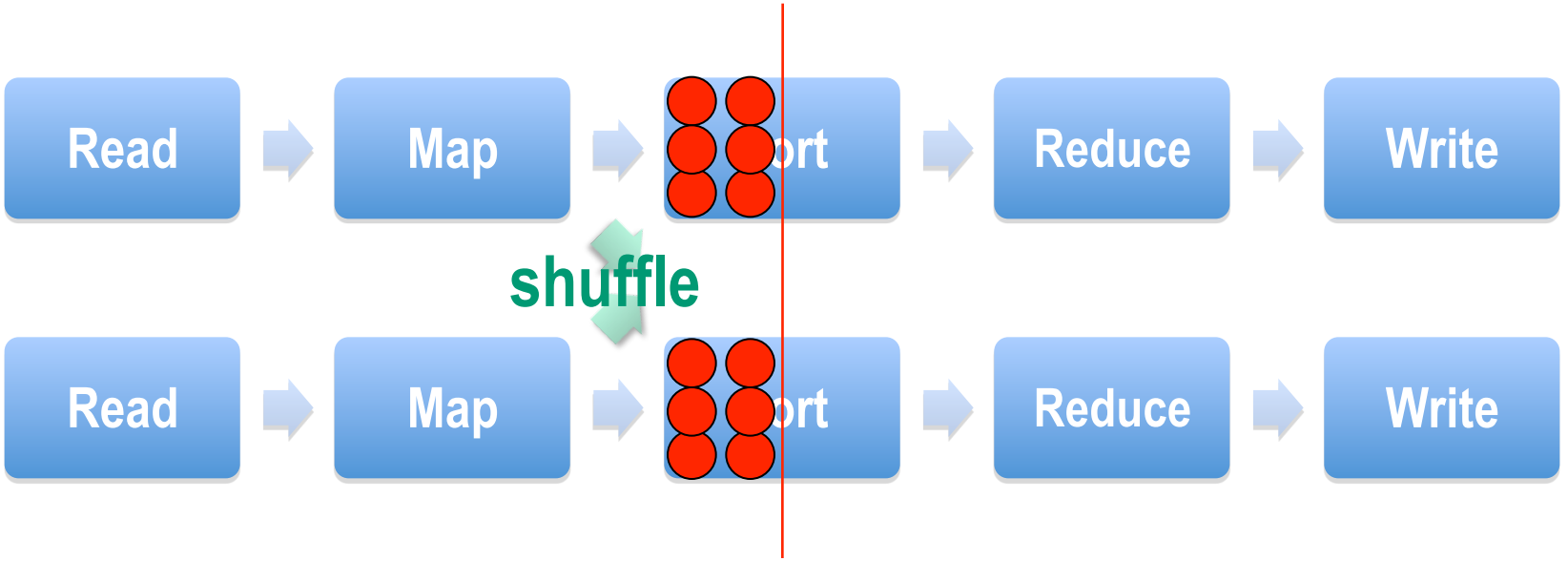
Phase 1: read, map and shuffle data



- Sort introduces barrier that disrupts pipeline

Visual of a map-reduce dataflow

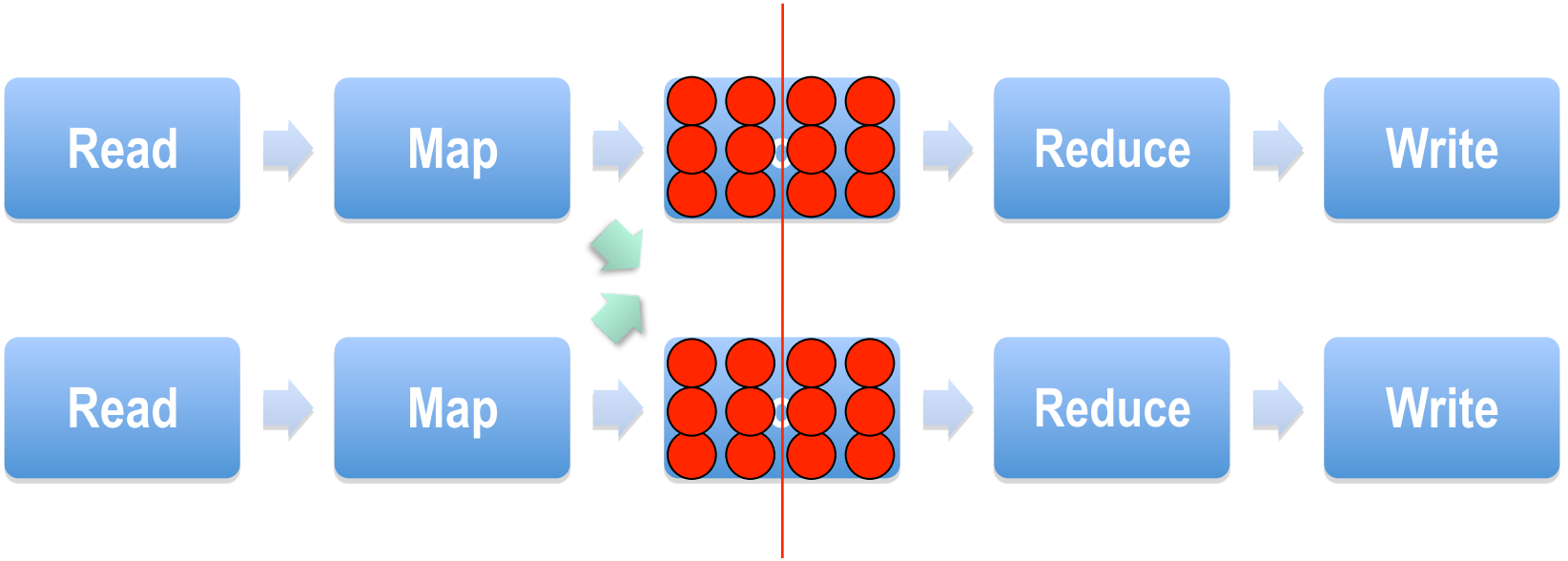
Phase 1: read, map and shuffle data



- **Sort introduces barrier that disrupts pipeline**

Visual of a map-reduce dataflow

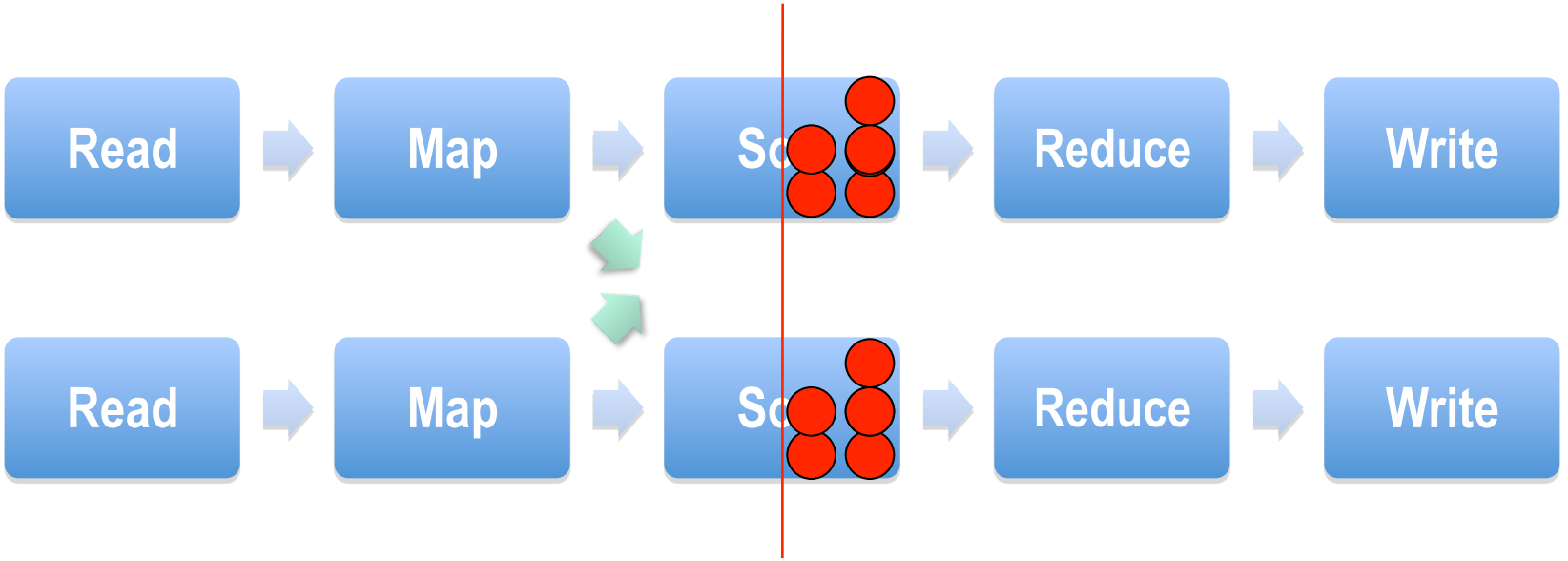
Phase 2: sort, reduce, and write data



- Sort introduces barrier that disrupts pipeline

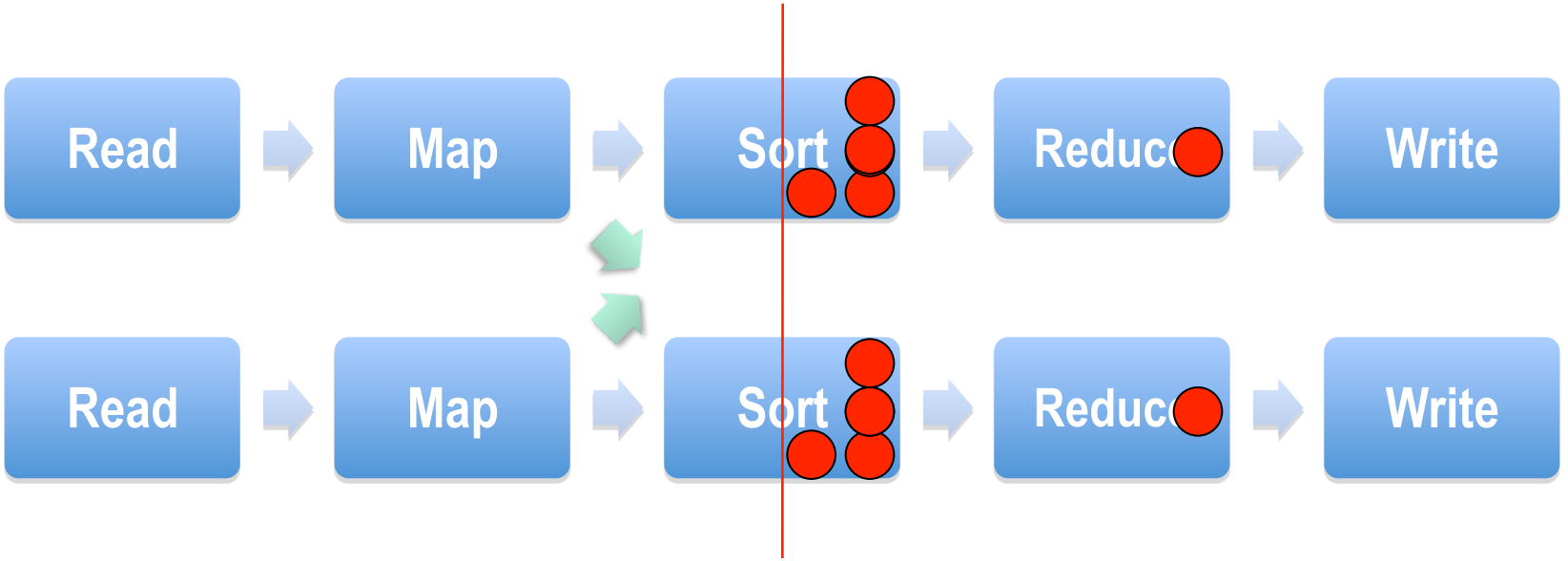
Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



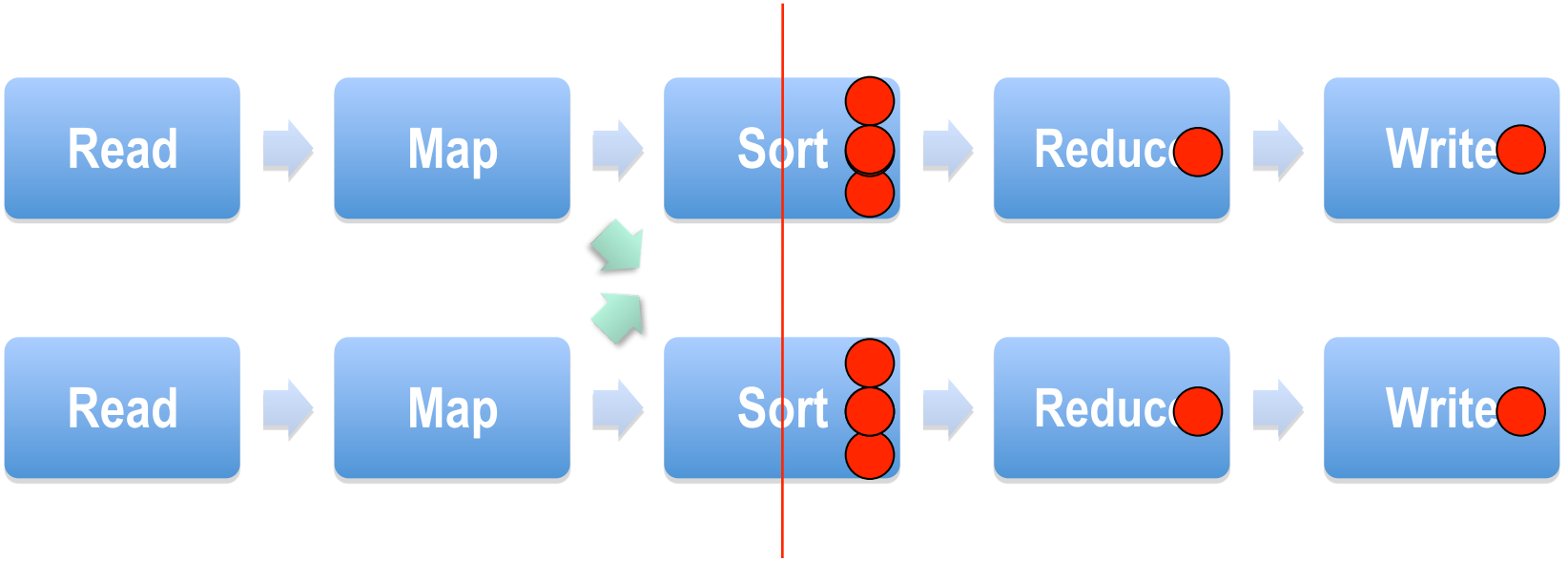
Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



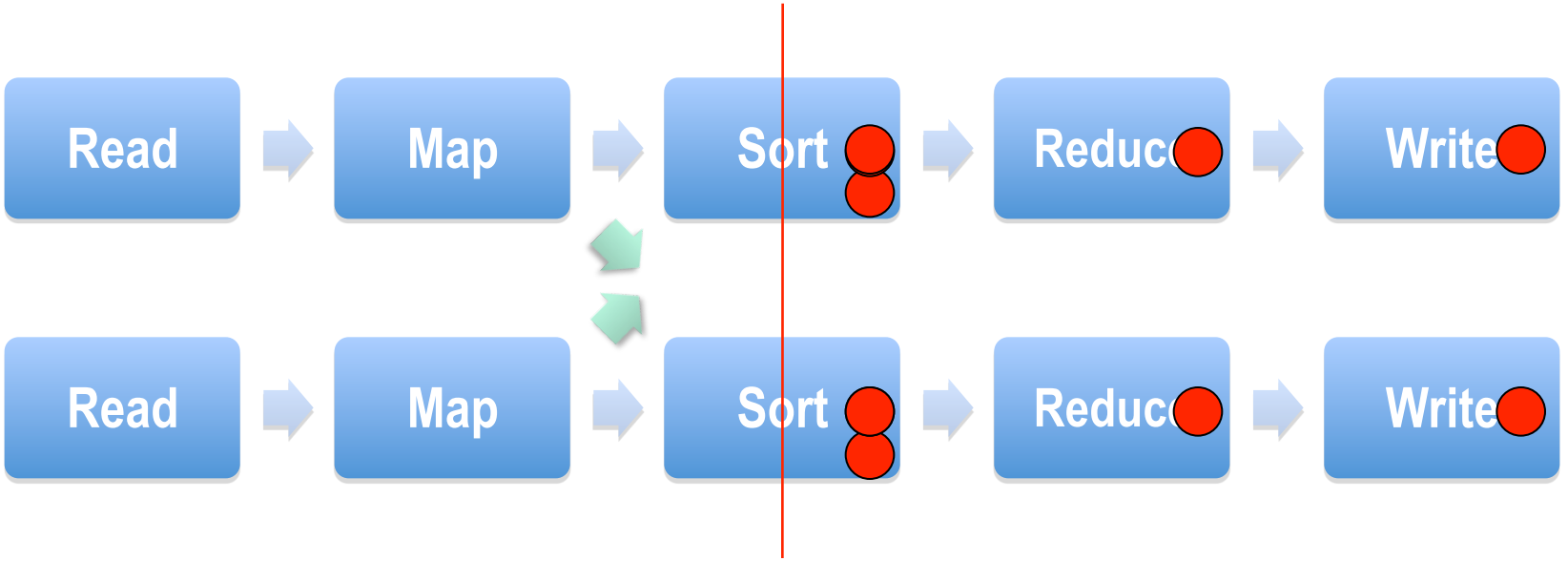
Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



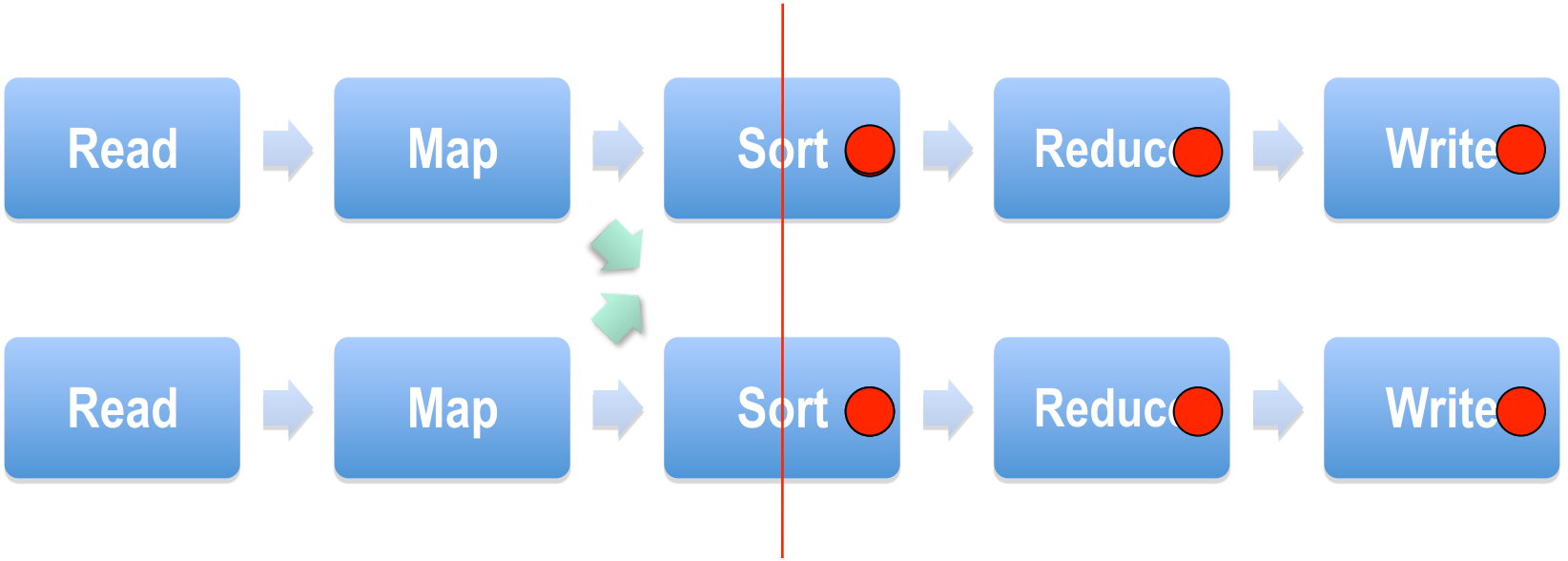
Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



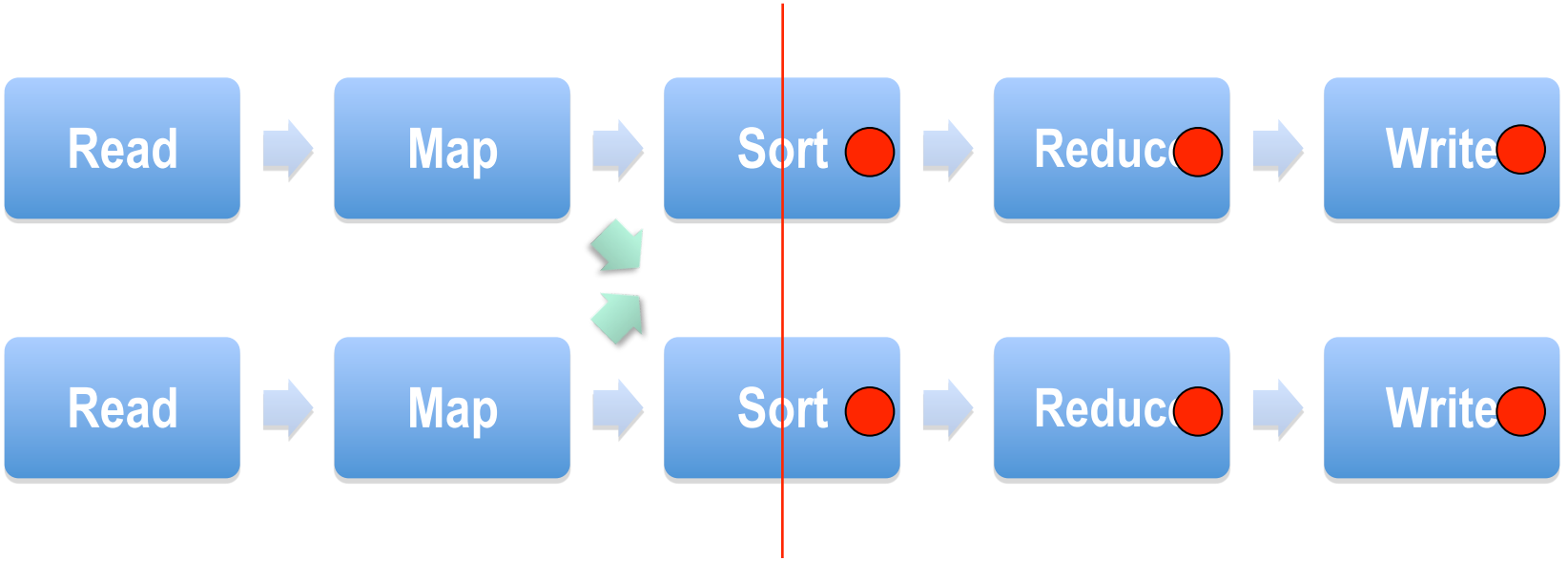
Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



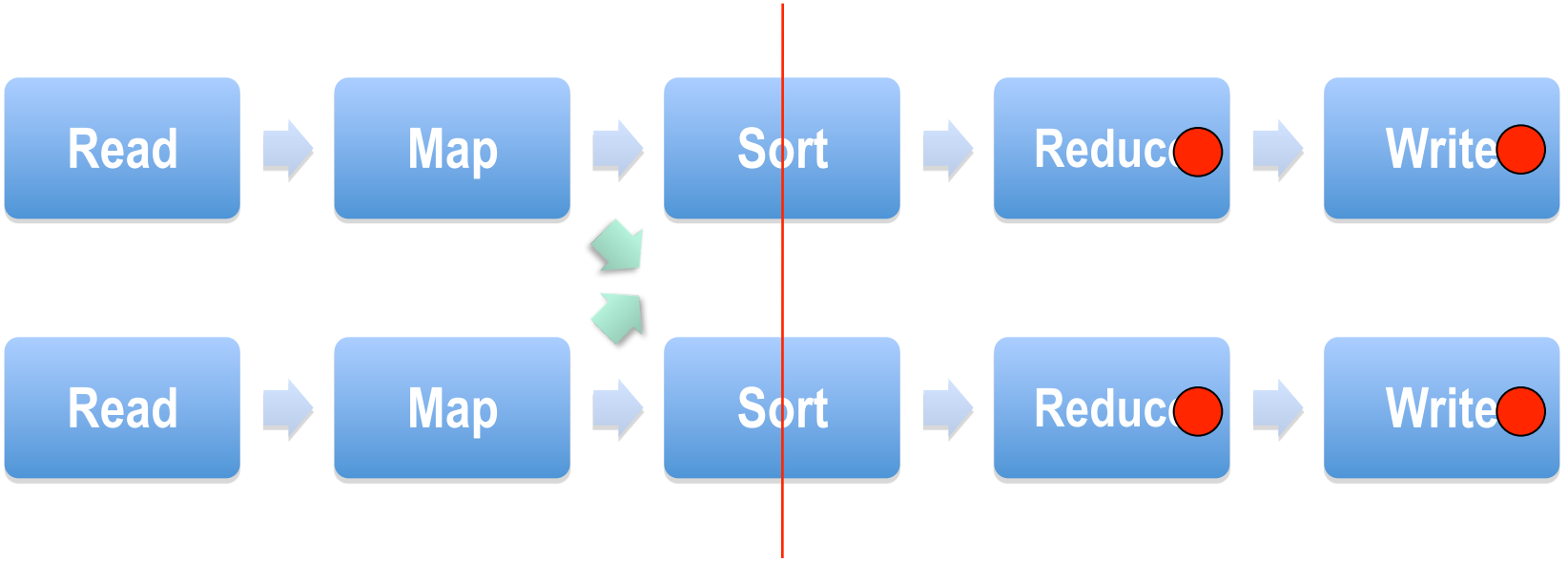
Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



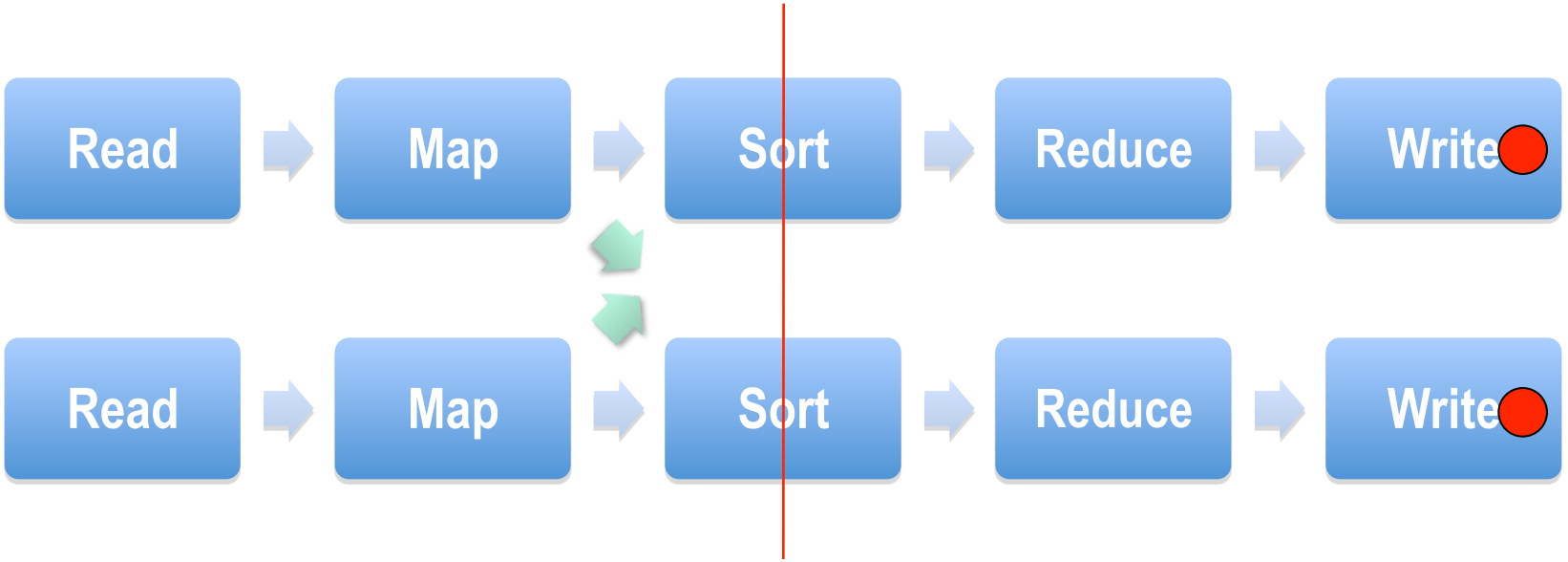
Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



Visual of a map-reduce dataflow

Phase 2: sort, reduce, and write data



Comments on map-reduce

- **Effective at large scale**
  - Google and others use it across 1000s of machines and PBs of data
    - to generate search indices, translate languages, and many other things
  - Used for setting sort benchmark records (e.g., TeraSort and PetaSort)
- **Indirectly helped spawn shift toward Data-Intensive Computing**
  - in which insights are mined from lots of observation data
  - Search for "Unreasonable Effectiveness of Data"
- **Not the "be all / end all" for parallel programming**
  - Great for relatively simple data-parallel activities
    - e.g., sifting through huge amounts of data
  - Not great for advanced machine learning algorithms
    - so, even newer APIs/frameworks being developed to support those

Today

- Parallel computing building blocks
  - Map-reduce programming
- **Virtual Machines**
- Cloud Computing

Virtualization

- **Decouple physical HW reality from exposed view**
  - We've seen "virtual memory" and processes
  - Apply same concept more generally
    - "virtual disks", "virtual networks", "virtual machines", etc.
- **Why?**
  - Efficiency and flexibility
    - Share HW resources, allow migration, etc.
- **Virtual Machines (VMs) are increasingly common**
  - A "virtual machine monitor" controls the HW resources
  - Each VM can look to the software within it as a machine
    - e.g., boot and execute an unmodified OS in a VM

What is cloud computing?

- **Short version:**
  - Using someone else's computers (and maybe software)
    - instead of buying/maintaining one's own
    - elastic and on-demand (pay for what need)
  - Sharing those computers with other "tenants"
    - instead of having them all-to-oneself
- **Longer version:**
  - See NIST's more complex definition (2 pages!)
    - a more technical and comprehensive statement
    - notes multiple styles, along multiple dimensions

Why cloud computing?

- **Huge potential benefits**
  - Consolidation
    - Higher server utilization (7-25% -> 70+%)
    - Economies of scale
    - E.g., HP went from 80+ data centers to 6
      - and saved $1B/year… over 60% of total annual expense
  - Aggregation
    - One set of experts doing it for many
      - Instead of each for themselves
- **Rapid deployment**
  - Rent when ready and scale as need
    - Rather than specify, buy, deploy, setup, then start

3 styles of Cloud Computing

- **IaaS – Infrastructure as a Service**
  - Data center rents VMs to users
    - Ex: Amazon EC2
  - User must install SW (platform & application)
- **PaaS – Platform as a Service**
  - Offer ready-to-run platform solutions
    - Ex: Google App Engine, Microsoft Azure
  - User develops/installs applications
- **SaaS – Software as a Service**
  - Complete application solutions are offered
  - Ex: Gmail, Salesforce.com, etc.

Cloud computing accessibility

- **Private vs. Public Clouds**
  - Private cloud: one organization
    - Multiple groups sharing a common infrastructure
    - Incredibly popular in business world, right now
  - Public cloud: many organizations
    - e.g., Internet offerings

Deeper: Operational costs out of control

Power and cooling

- Now on par with purchase costs
- Trends making it worse every year
  - Power/heat go up with speed
  - Cluster sizes increase due to commodity pricing

EPA report about 2011 data center power usage:

In 2006, 1.5% of total U.S. electricity consumption

"Under current efficiency trends, national energy consumption by servers and data centers could nearly double again in another five years (i.e., by 2011) to more than 100 billion kWh."

[i.e., 2-3% of total U.S. consumption]

A few "fun" data center energy facts

"Google's power consumption ... would incur an annual electricity bill of nearly $38 million"

[Qureshi:sigcomm09]

"Energy consumption by … data centers could nearly double ... (by 2011) to more than 100 billion kWh, representing a $7.4 billion annual electricity cost"

[EPA Report 2007]

Annual cost of energy for Google, Amazon, Microsoft

=

Annual cost of all first-year CS PhD Students

Deeper: Operational costs out of control

Power and cooling

- Now on par with purchase costs
- Trends making it worse every year
  - Power/heat go up with speed
  - Cluster sizes increase due to commodity pricing

Administration costs

- Often reported at 4-7X capital expenditures
- Trends making it worse every year
  - Complexity goes up with features, expectations and cluster size
  - Salaries go up while equipment costs go down