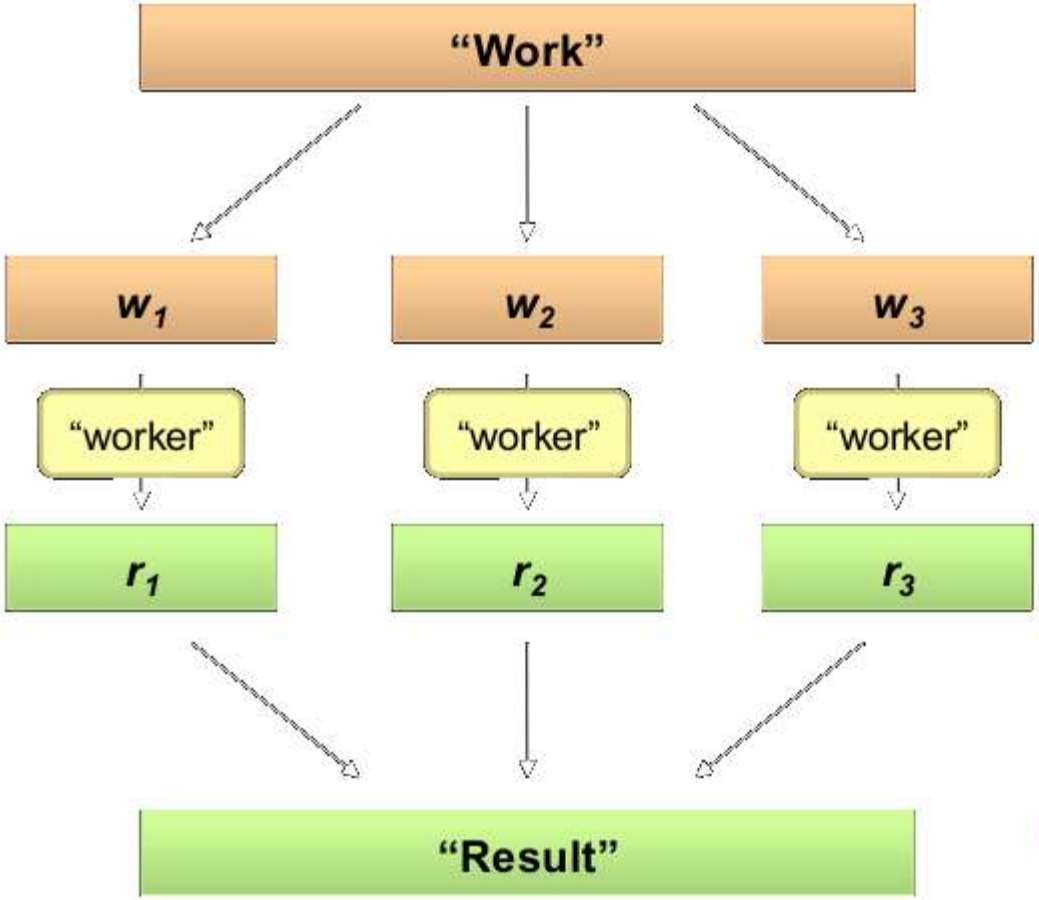
IRIT
CNRS - INPT - UPS - UT1 - UT2J

Map Reduce

dacosta@irit.fr

cnrs INP UNIVERSITÉ TOULOUSE III PAUL SABATIER UNIVERSITÉ TOULOUSE 1 CAPITOLE UNIVERSITÉ TOULOUSE Jean Jaurès

Divide and conquer at PaaS



Partition

100 % //

Combine

- Troisième niveau

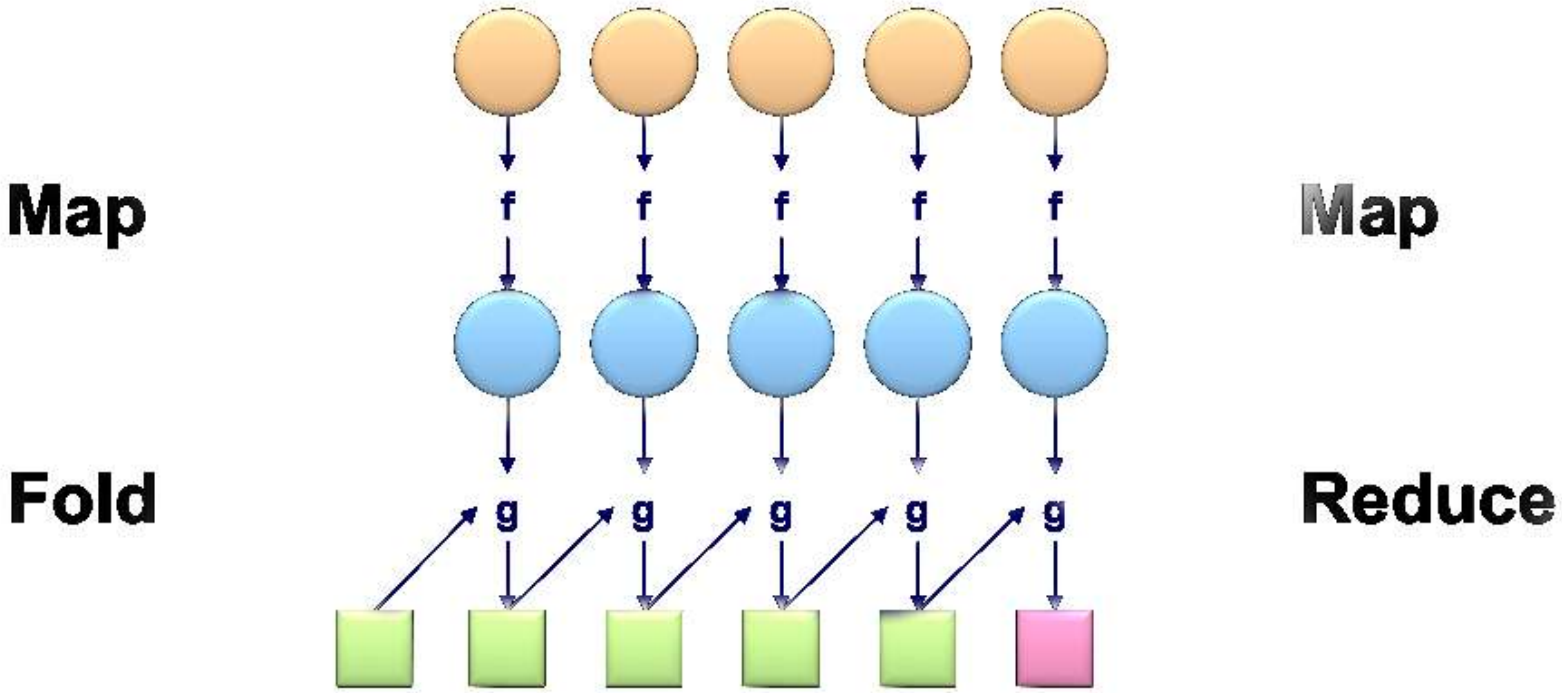
Typical problem

- Iterate over a large number of records
- Extract something of interest from each MAP
- Shuffle and sort intermediate results
- Aggregate intermediate results Reduce
- Generate final output

Key idea: functional abstraction for these two operations

- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
- Seventh Outline LevelCliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

Folding



- Premier niveau
  - Deuxième niveau
    - Troisième niveau

Difficulties ?

- Huge amount of data
  - Do not fit into memory
  - Access patterns are broad
  - Most data not accessed frequently
- Complex data
  - links between data or treatment
  - Same data can be treated in different ways
  - No pre-processing
- Seventh Outline Level Cliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

Click to edit the outline text format

Second Outline Level

Third Outline Level

Fourth Outline Level

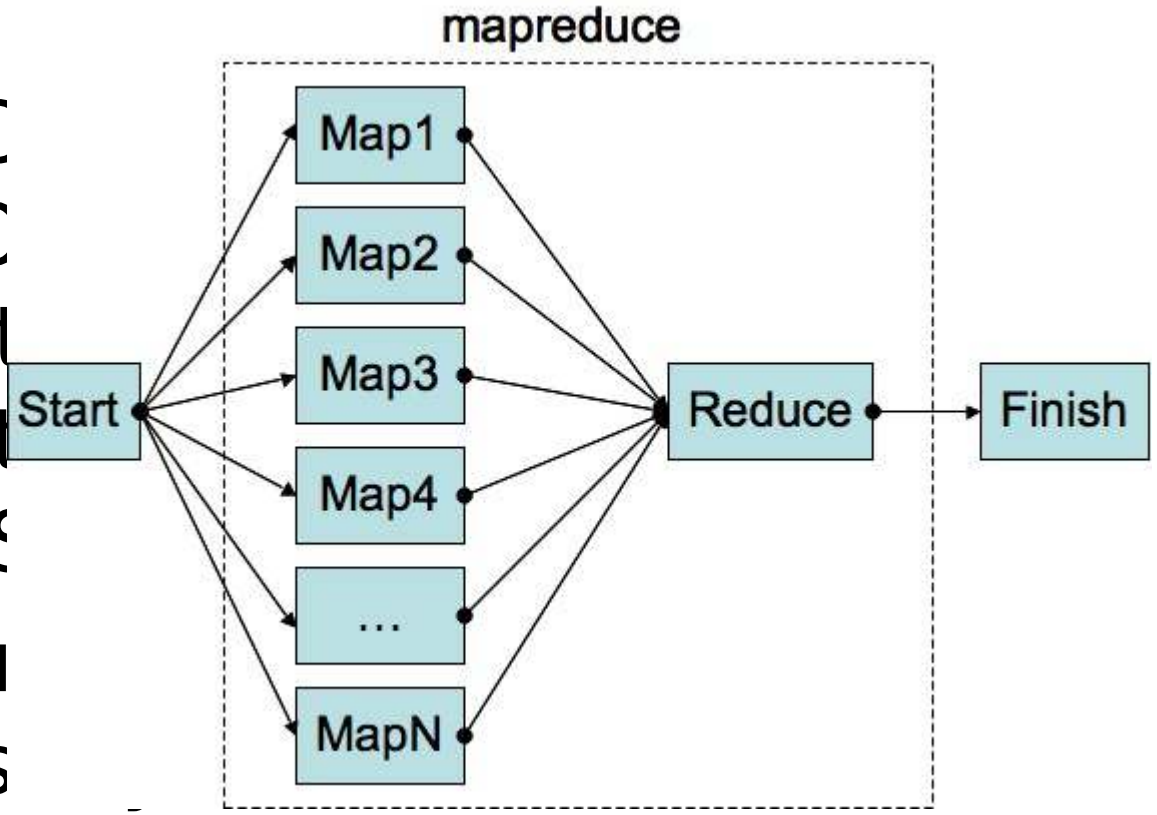
Fifth Outline Level

Sixth Outline Level

Example : crawling through internet data

Principle

- "Map" step: The master node takes the input, divides it into smaller sub-problems, and distributes them to worker nodes
- "Reduce" step: The master node then collects the answers to all the sub-problems and combines them in some way to form the output



- Click to edi
  - Second C
    - Third C
      - Fourt
        - Fift
          - S
            - Seventh Ou
              - modifier les
                - masque
  - Deuxième niveau
    - Troisième niveau

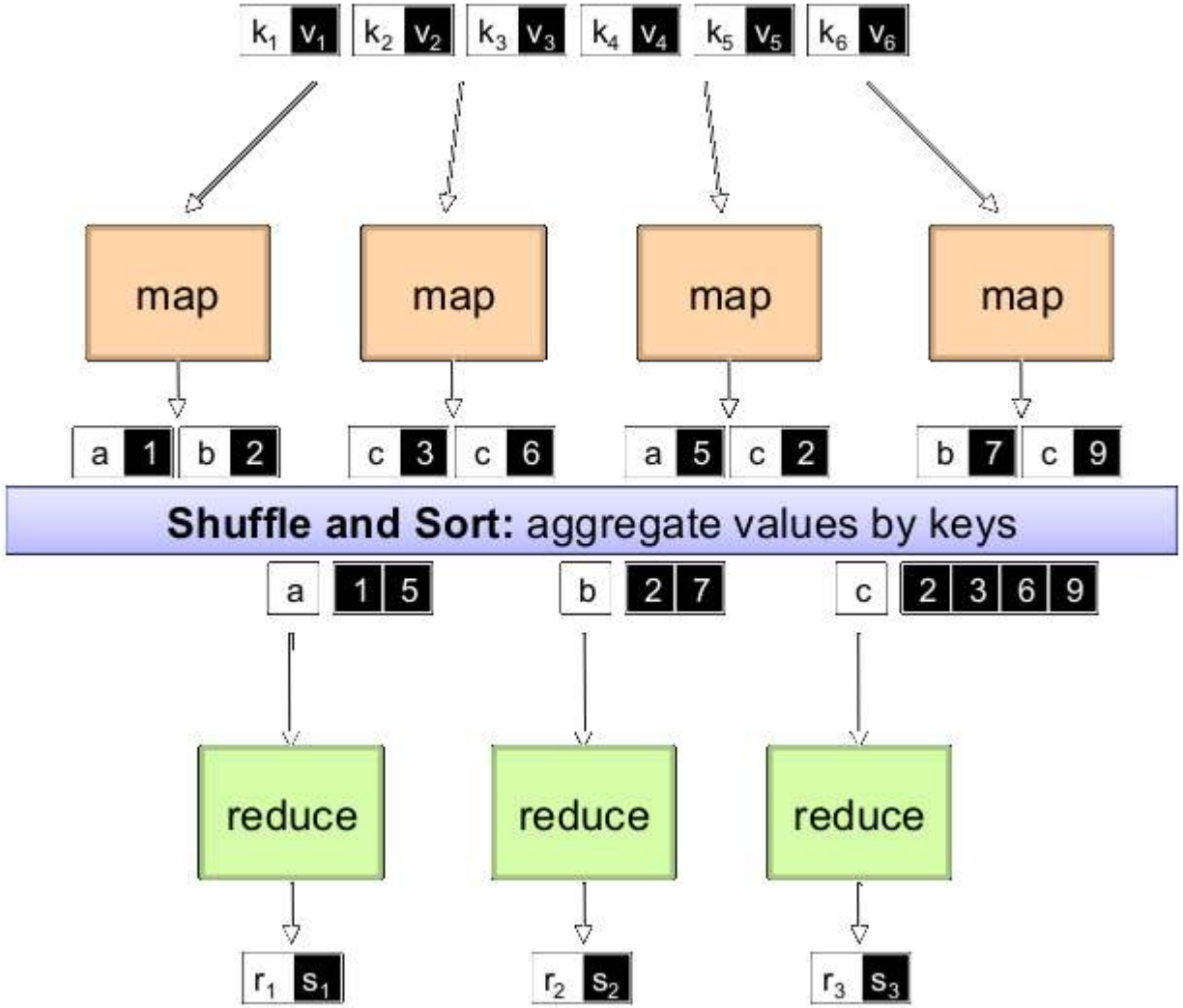
MapReduce

- Programmers specify two functions:
  - map (k,v) → <k’, v’>*
  - reduce (k’, v’) → <k’, v’>*
  - All values with the same key are reduced together
- Usually, programmers also specify:
  - partition (k’, number of partitions) → partition for k’
    - Often a simple hash of the key, e.g. hash(k’) mod n
    - Allows reduce operations for different keys in parallel
  - combine(k’,v’) → <k’,v’>
    - Mini-reducers that run in memory after the map phase
    - Optimizes to reduce network traffic & disk writes
- Implementations:
  - Google has a proprietary implementation in C++
  - Hadoop is an open source implementation in Java

• Click to edit the outline text format
• Second Outline Level
• Third Outline Level
• Fourth Outline Level
• Fifth Outline Level
• Sixth Outline Level
• Seventh Outline Level Cliquez pour modifier les styles du texte du masque
– Deuxième niveau
• Troisième niveau

- (

- s
  r
  r



  - Deuxième niveau
    - Troisième niveau

Word count

-

```
function map(String name, String document):
  // name: document name
  // document: document contents
  for each word w in document:
    emit (w, 1)

function reduce(String word, Iterator partialCounts):
  // word: a word
  // partialCounts: a list of aggregated partial counts
  sum = 0
  for each pc in partialCounts:
    sum += ParseInt(pc)
  emit (word, sum)
```

- modifier les styles du texte du masque
  - Deuxième niveau

Exemple : Average number of contract by Age

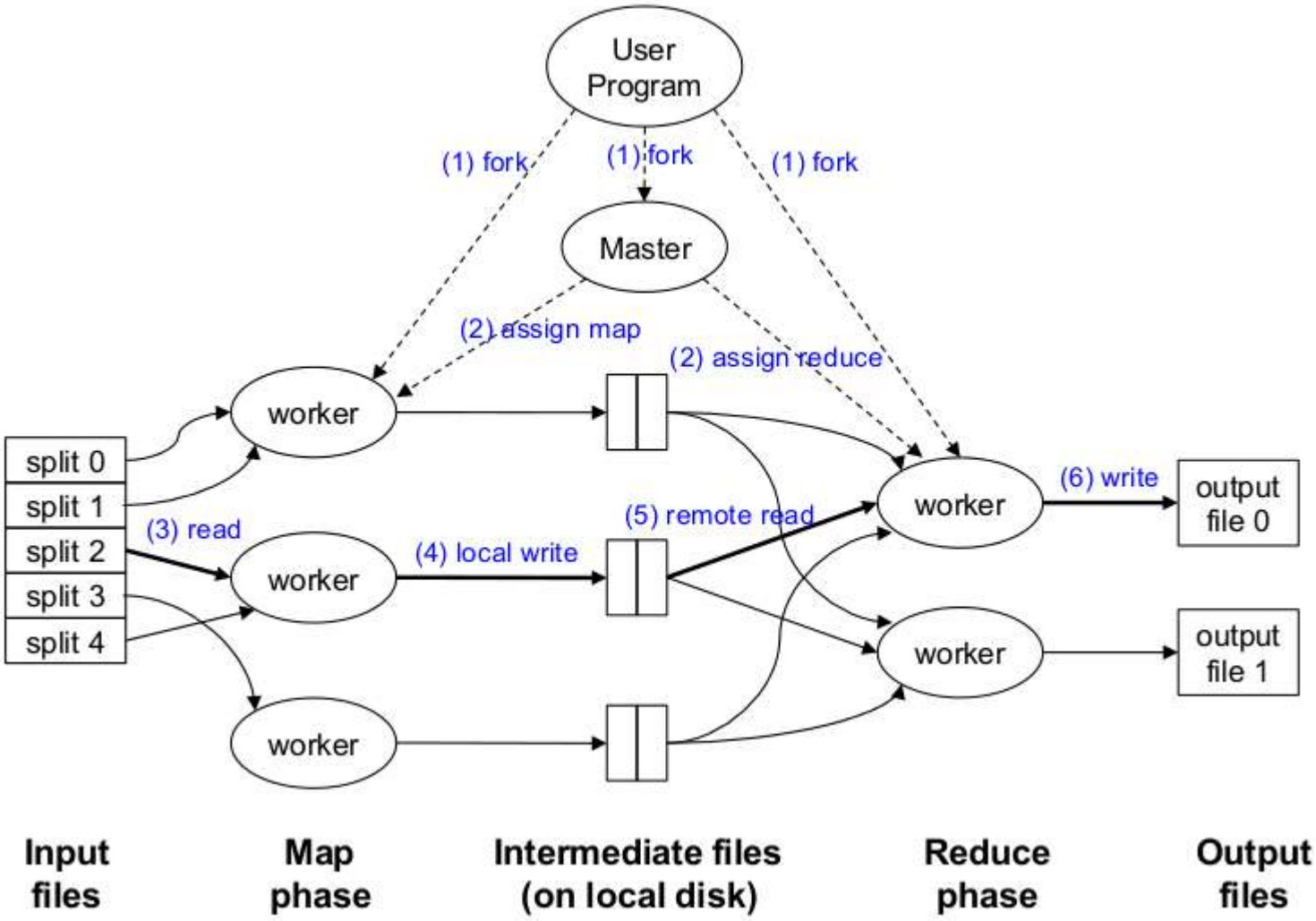
- For 1 million entry
  - Batch of 1000
  - 1100 of them
- Output of Map
  - Range 8-110
- Reduce :
  - Batch of 1 Y
  - 102 of them
  - Treat 1000
- Output
  - 102 values

```
function Map is
    input: integer K1 between 1 and 1100
    for each social.person record in the K1 batch do
        let Y be the person's age
        let N be the number of contacts the person has
        produce one output record <Y,N>
    repeat
end function

function Reduce is
    input: age (in years) Y
    for each input record <Y,N> do
        Accumulate in S the sum of N
        Accumulate in C the count of records so far
    repeat
    let A be S/C
    produce one output record <Y,A>
end function
```

MapReduce Runtime

- Handles scheduling
  - Assigns workers to map and reduce tasks
- Handles "data distribution"
  - Moves the process to the data
- Handles synchronization
  - Gathers, sorts, and shuffles intermediate data
- Handles faults
  - Detects worker failures and restarts
- Everything happens on top of a distributed FS (later)



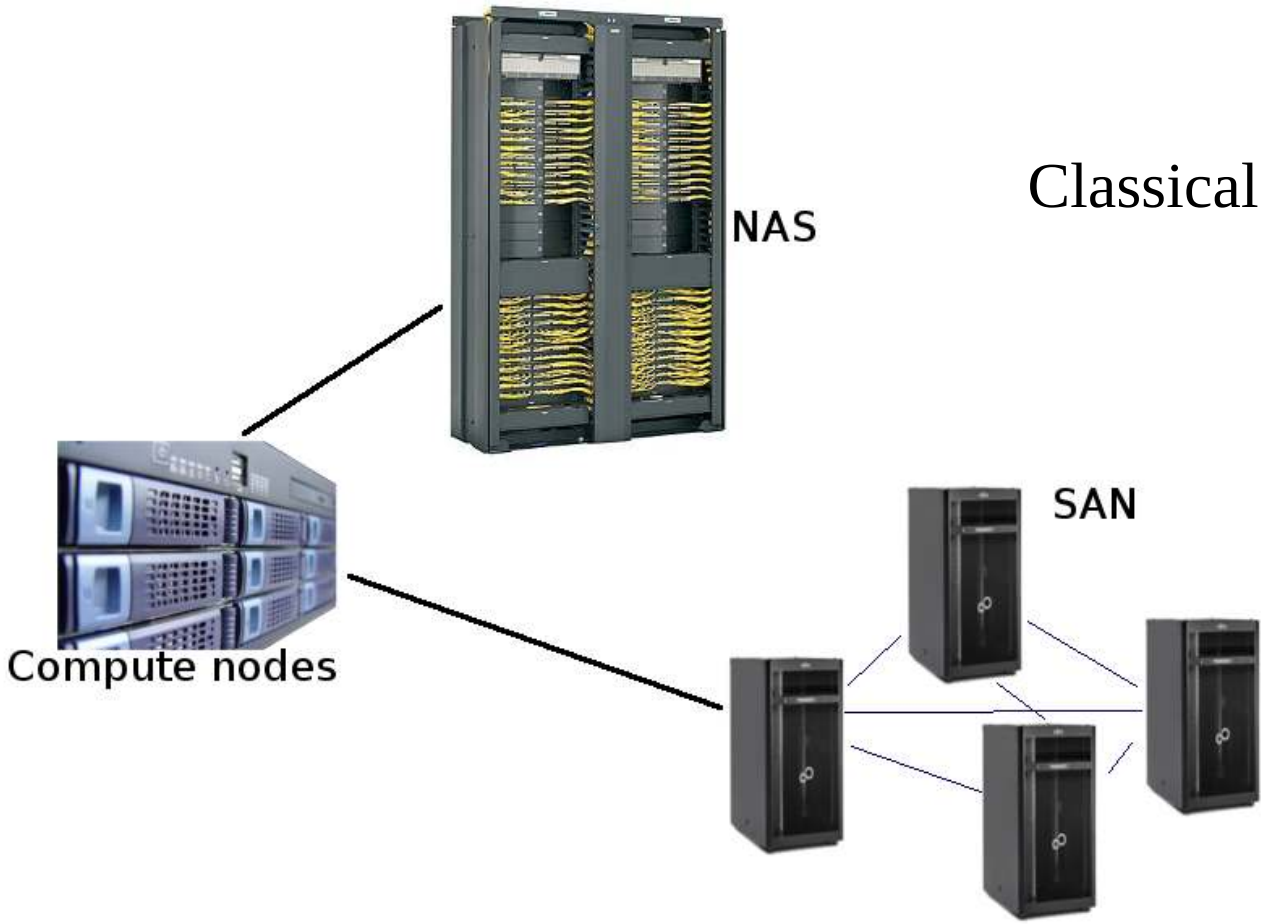
-
-

  - Deuxième niveau
    - Troisième niveau

How do we get data to the workers

- t

Classical cluster vision



-

What's the problem here ?

– Deuxième niveau

- Don't move data to workers… Move workers to the data!
  - Store data on the local disks for nodes in the cluster
  - Start up the workers on the node that has the data local
- Why ?
  - Not enough RAM to hold all the data in memory
  - Disk access is slow, disk throughput is good
- A distributed file system is the answer
  - GFS (Google File System)
  - HDFS for Hadoop (= GFS clone)

- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
- Seventh Outline LevelCliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

GFS: Assumptions

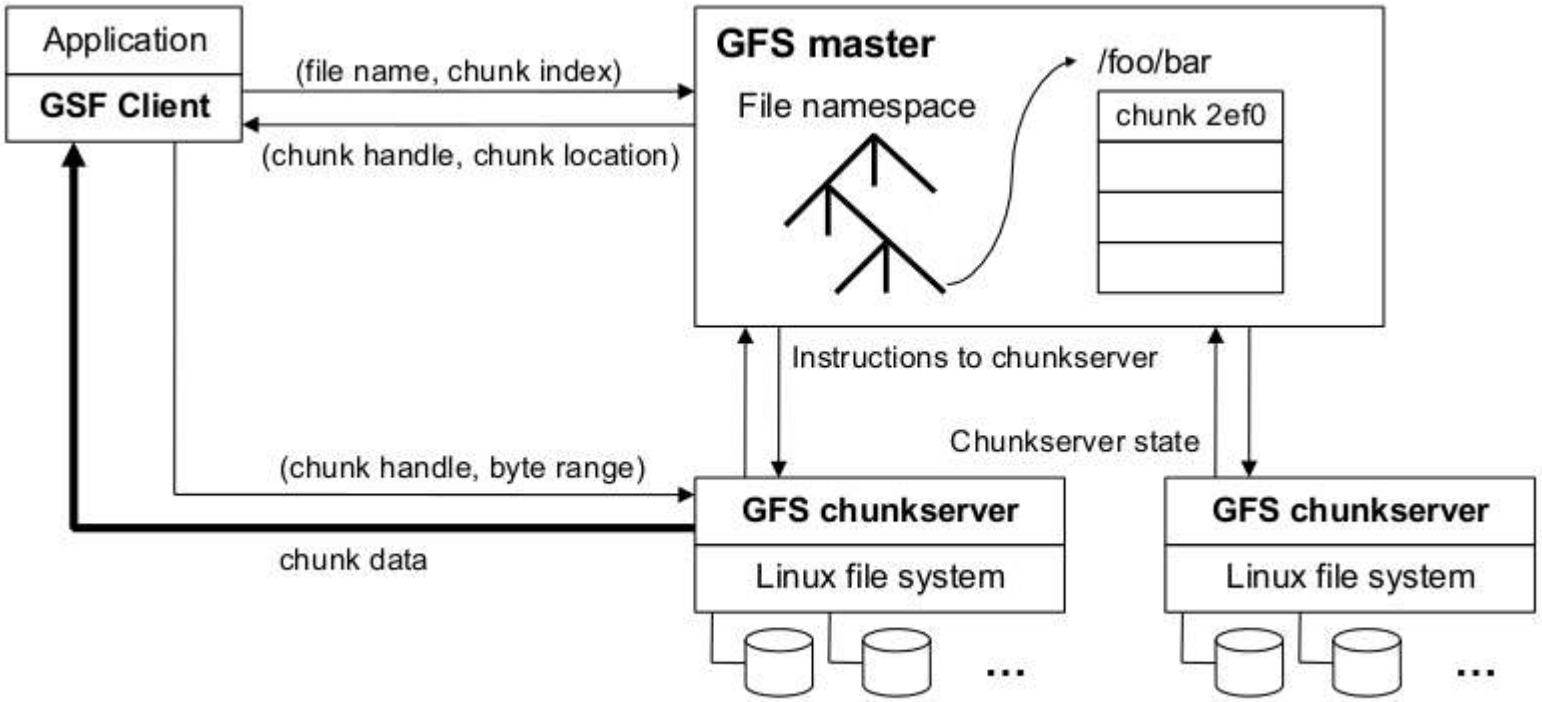
- Commodity hardware over “exotic” hardware
  - High component failure rates
    - Inexpensive commodity components fail all the time
- “Modest” number of HUGE files
- Files are write-once, mostly appended to
  - Perhaps concurrently
- Large streaming reads over random access
- High sustained throughput over low latency

Click to edit the outline text format
Second Outline Level
Third Outline Level
Fourth Outline Level
Fifth Outline Level
Sixth Outline Level
Seventh Outline Level Cliquez pour modifier les styles du texte du masque
Deuxième niveau
Troisième niveau

GFS: Design Decisions

- Files stored as chunks
  - Fixed size (64MB)
- Reliability through replication
  - Each chunk replicated across 3+ chunkservers
- Single master to coordinate access, keep metadata
  - Simple centralized management
- No data caching
  - Little benefit due to large data sets, streaming reads
- Simplify the API
  - Push some of the issues onto the client

- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
            - Seventh Outline LevelCliquez pour modifier les styles du texte du masque
              - Deuxième niveau
                - Troisième niveau



- modifier les styles du texte du masque
  - Deuxième niveau

- Metadata storage
- Namespace management/locking
- Periodic communication with chunkservers
- Chunk creation, replication, rebalancing
- Garbage collection

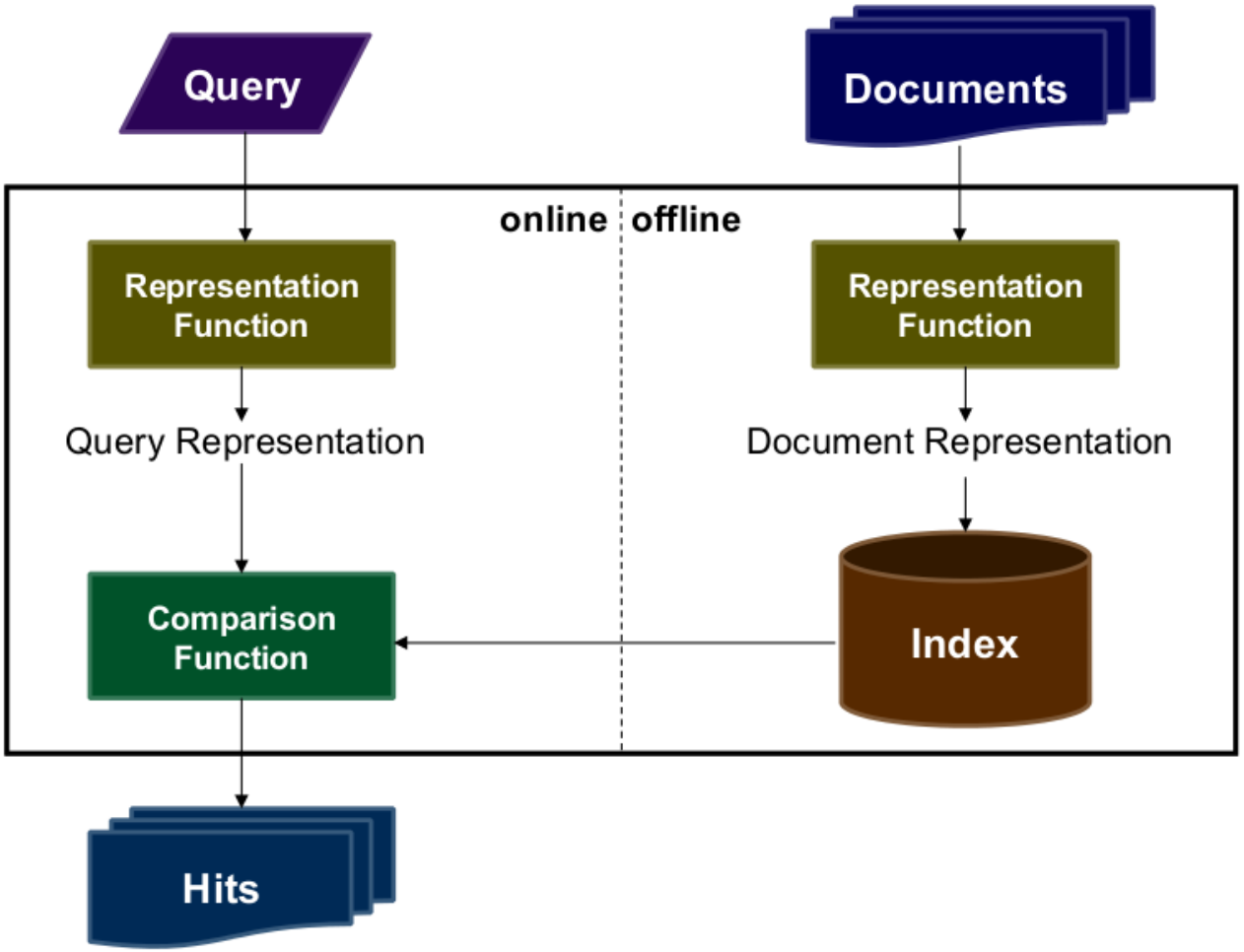
Click to edit the outline text format
- Second Outline Level
  - Third Outline Level
    - Fourth Outline Level
      - Fifth Outline Level
        - Sixth Outline Level
          - Seventh Outline Level

Cliquez pour modifier les styles du texte du masque
- Deuxième niveau
  - Troisième niveau

- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
- Seventh Outline LevelCliquez pour modifier les styles du texte du masque
  - Deuxième niveau

Exemple : Inverted Indexing

- rmat
- our



  - Deuxième niveau
    - Troisième niveau

How do we represent text?

- "Bag of words"
  - Treat all the words in a document as index terms for that document
  - Assign a weight to each term based on "importance"
  - Disregard order, structure, meaning, etc. of the words
  - Simple, yet effective!
- Assumptions
  - Term occurrence is independent
  - Document relevance is independent
  - "Words" are well-defined

Cliquez pour modifier les styles du texte du masque

- Deuxième niveau
  - Troisième niveau
    - Quatrième niveau
      - Cinquième niveau
        - Sixième niveau
          - Septième niveau

Click to edit the outline text format

- Second Outline Level
  - Third Outline Level
    - Fourth Outline Level
      - Fifth Outline Level
        - Sixth Outline Level
          - Seventh Outline Level

Sample Document

- McDonald's slims down spuds
- Fast-food chain to reduce certain types of fat in its french fries with new cooking oil. “Bag of Words”
- NEW YORK (CNN/Money) - McDonald's Corp. is cutting the amount of "bad" fat in its french fries nearly in half, the fast-food chain said Tuesday as it moves to make all its fried menu items healthier.
- But does that mean the popular shoestring frieswon't taste the same? The company says no. "It's a win-win for our customers because they are getting the same great french-fry taste along with an even healthier nutrition profile," said Mike Roberts, president of McDonald's USA.
- But others are not so sure. McDonald's will not specifically discuss the kind of oil it plans to use, but at least one nutrition expert says playing with the formula could mean a different taste.
- …

- “Bag of Words”
  - 16 × said
  - 14 × McDonalds
  - 12 × fat
  - 11 × fries
  - 8 × new
  - 6 × company, french, nutrition
  - 5 × food, oil, percent, reduce, taste, Tuesday
  - …

- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
            - Seventh Outline Level
- Cliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

Representing Documents

Document 1

The quick brown fox jumped over the lazy dog's back.

Document 2

Now is the time for all good men to come to the aid of their party.

Term	Document 1	Document 2
aid	0	1
all	0	1
back	1	0
brown	1	0
come	0	1
dog	1	0
fox	1	0
good	0	1
jump	1	0
lazy	1	0
men	0	1
now	0	1
over	1	0
party	0	1
quick	1	0
their	0	1
time	0	1

Stopword List

for
is
of
the
to

- ormat
- pour
  - Deuxième niveau
    - Troisième niveau

Inverted Index

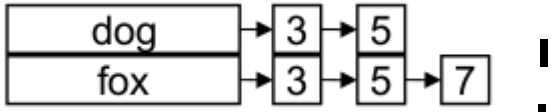
Term	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8
aid	0	0	0	1	0	0	0	1
all	0	1	0	1	0	1	0	0
back	1	0	1	0	0	0	1	0
brown	1	0	1	0	1	0	1	0
come	0	1	0	1	0	1	0	1
dog	0	0	1	0	1	0	0	0
fox	0	0	1	0	1	0	1	0
good	0	1	0	1	0	1	0	1
jump	0	0	1	0	0	0	0	0
lazy	1	0	1	0	1	0	1	0
men	0	1	0	1	0	0	0	1
now	0	1	0	0	0	1	0	1
over	1	0	1	0	1	0	1	1
party	0	0	0	0	0	1	0	1
quick	1	0	1	0	0	0	0	0
their	1	0	0	0	1	0	1	0
time	0	1	0	1	0	1	0	0

Term	Postings
aid	4 → 8
all	2 → 4 → 6
back	1 → 3 → 7
brown	1 → 3 → 5 → 7
come	2 → 4 → 6 → 8
dog	3 → 5
fox	3 → 5 → 7
good	2 → 4 → 6 → 8
jump	3
lazy	1 → 3 → 5 → 7
men	2 → 4 → 8
now	2 → 6 → 8
over	1 → 3 → 5 → 7 → 8
party	6 → 8
quick	1 → 3
their	1 → 5 → 7
time	2 → 4 → 6

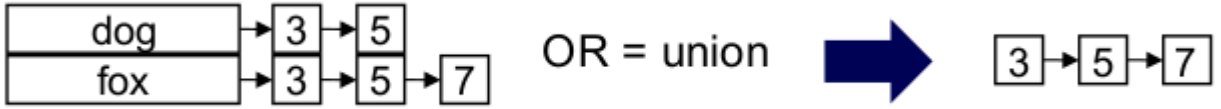
- Deuxième niveau
  - Troisième niveau

- To execute a Boolean query:
  - Build query syntax tree

- For each clause, look up postings



- Traverse postings and apply Boolean operator



- Efficiency analysis
  - Postings traversal is linear (assuming sorted postings)
  - Start with shortest posting first

Click to edit Master text styles
Second level
Third level
Fourth level
Fifth level
Seventh Outline LevelCliquez pour modifier les styles du texte du masque
Deuxième niveau
Troisième niveau

- Click to edit the outline text format
  - $$w_{i,j} = \mathrm{tf}_{i,j} \cdot \log \frac{N}{n_i}$$

    $w_{i,j}$ **weight assigned to term *i* in document *j***

    $\mathrm{tf}_{i,j}$ **number of occurrence of term *i* in document *j***

    N **number of documents in entire collection**

    n_i **number of documents with term *i***

- Se… …our m… masque
  - Deuxième niveau

	tf				idf
	1	2	3	4	
complicated			5	2	0.301
contaminated	4	1	3		0.125
fallout	5		4	3	0.125
information	6	3	3	2	0.000
interesting		1			0.602
nuclear	3		7		0.301
retrieval		6	1	4	0.125
siberia	2				0.602

complicated	0.301	3,5 4,2
contaminated	0.125	1,4 2,1 3,3
fallout	0.125	1,5 3,4 4,3
information	0.000	1,6 2,3 3,3 4,2
interesting	0.602	2,1
nuclear	0.301	1,3 3,7
retrieval	0.125	2,6 3,1 4,4
siberia	0.602	1,2

masque

- Deuxième niveau
  - Troisième niveau

MapReduce it?

- The indexing problem
  - Must be relatively fast, but need not be real time
  - For Web, incremental updates are important
  - Crawling is a challenge itself
- The retrieval problem
  - Must have sub-second response
  - For Web, only need relatively few results

Click to edit the outline text format

- Second Outline Level
  - Third Outline Level
    - Fourth Outline Level
      - Fifth Outline Level
        - Sixth Outline Level
- Seventh Outline LevelCliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

- Fundamentally, a large sorting problem
  - Terms usually fit in memory
  - Postings usually don't
- How is it done on a single machine?
- How large is the inverted index?
  - Size of vocabulary
  - Size of postings

MapReduce: Index Construction

- Map over all documents
  - Emit *term* as key, *(docid, tf)* as value
  - Emit other information as necessary (e.g., term position)
- Reduce
  - Trivial: each value represents a posting!
  - Might want to sort the postings (e.g., by docid or tf)
- MapReduce does all the heavy lifting!

- MapReduce is meant for large data batch processing
  - Not suitable for lots of real time operations requiring low latency
- The solution: “the secret sauce”
  - Lots of system engineering: e.g., caching, load balancing, etc.

Click to edit the outline text format
Second Outline Level
Third Outline Level
Fourth Outline Level
Fifth Outline Level
Sixth Outline Level
Seventh Outline Level

- Cliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
- Seventh Outline LevelCliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

- Remember: Mappers run in isolation
  - You have no idea in what order the mappers run
  - You have no idea on what node the mappers run
  - You have no idea when each mapper finishes
- Tools for synchronization:
  - Ability to hold state in reducer across multiple key-value pairs
  - Sorting function for keys
  - Partitioner
  - Cleverly-constructed data structures

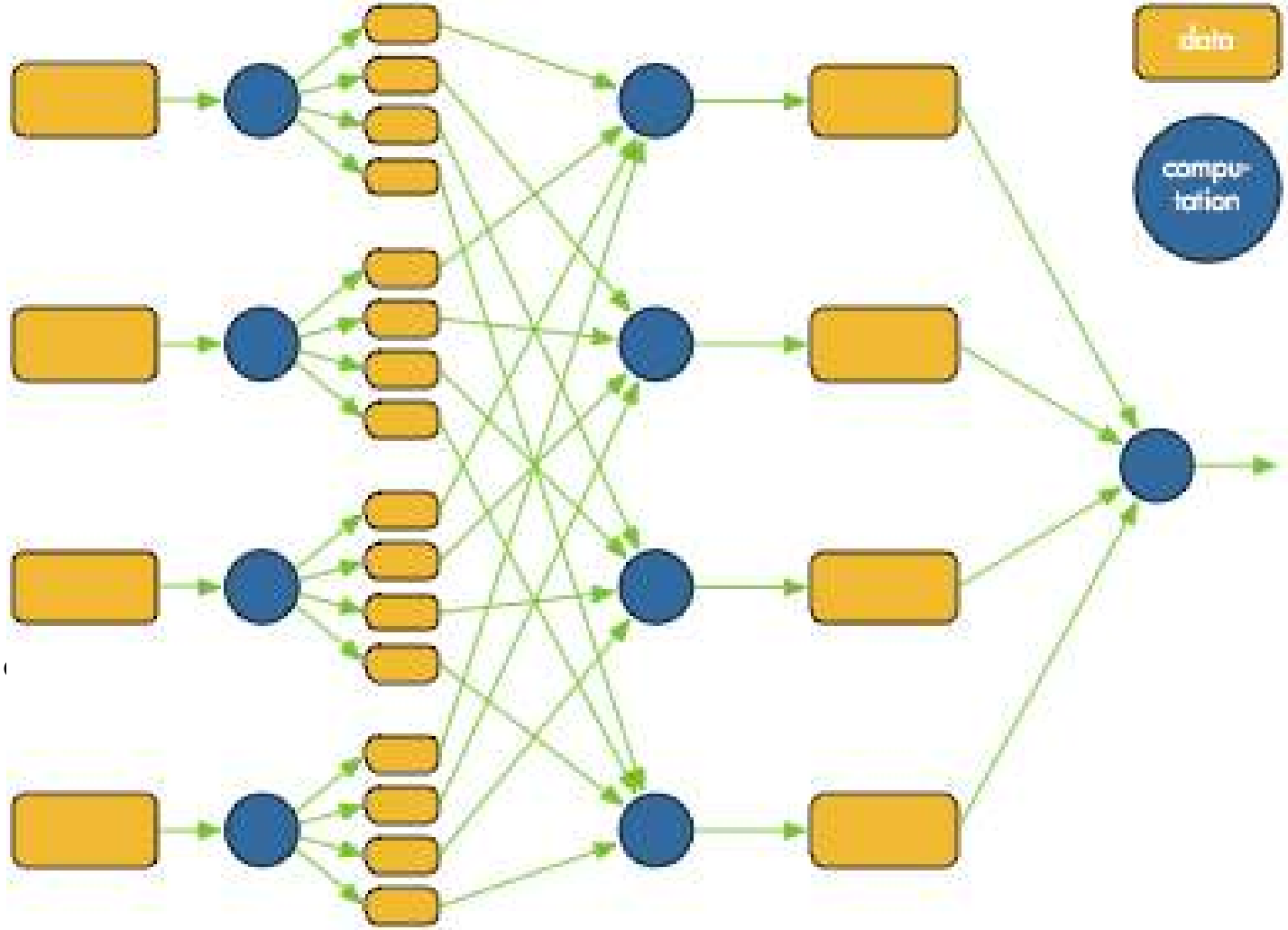
- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
            - Seventh Outline LevelCliquez pour modifier les styles du texte du masque
              - Deuxième niveau
                - Troisième niveau

For the programmer

- Input reader
  - The input reader reads data from stable storage and generates key/value pairs.
- Map function
  - Takes a series of key/value pairs, processes each, and generates zero or more output key/value pairs
- Partition function
  - Each Map function output is allocated to a particular reducer by the application's partition function
- Compare function
- Reduce function
  - The framework calls the application's Reduce function once for each unique key in the sorted order
- Output writer
  - writes the output of the Reduce to the stable storage

Click to edit the outline text format
- Second Outline Level
  - Third Outline Level
    - Fourth Outline Level
      - Fifth Outline Level
        - Sixth Outline Level
          - Seventh Outline LevelCliquez pour modifier les styles du texte du masque
            - Deuxième niveau
              - Troisième niveau

Input -> Map -> Copy/Sort -> Reduce -> Output



nat

ur

- Deuxième niveau
  - Troisième niveau

Use cases

- Word count
  - Before: 1 message per word in the text
  - Here: 1 message per word in the text
- Click to edit the outline text format
  - Second Outline Level
    - Third Outline Level
      - Fourth Outline Level
        - Fifth Outline Level
          - Sixth Outline Level
            - Seventh Outline Level

```
class Mapper
   method Map(docid id, doc d)
      H = new AssociativeArray
      for all term t in doc d do
          H{t} = H{t} + 1
      for all term t in H do
         Emit(term t, count H{t})
```

- Cliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

Co-occurence

- Count the number of co-occurence of n elements in sets
- Exemple
  - Words appears in same sentence
  - Customer who buy this also buy that
- If there are N elements
  - Report occurrence of NxN couples
- On a single node, quite simple
  - Foreach set
    - Foreach i in set
      - Foreach j in set
        - res[i][j]++

Map Reduce version ?

Click to edit the outline text format
- Second Outline Level
  - Third Outline Level
    - Fourth Outline Level
      - Fifth Outline Level
        - Sixth Outline Level
          - Seventh Outline LevelCliquez pour modifier les styles du texte du masque
            - Deuxième niveau
              - Troisième niveau

Pairs approach

- Cl rmat
  - ```
    class Mapper
        method Map(null, items [i1, i2,...] )
            for all item i in [i1, i2,...]
                for all item j in [i1, i2,...]
                    Emit(pair [i j], count 1)

    class Reducer
        method Reduce(pair [i j], counts [c1, c2,...])
            s = sum([c1, c2,...])
            Emit(pair[i j], count s)
    ```
    - Fifth Outline Level
      - Sixth Outline Level
        - Seventh Outline LevelCliquez pour modifier les styles du texte du masque
          - Deuxième niveau
            - Troisième niveau

- Too many intermediary keys
- Easy and strayforward implementation
- Optimize using local accumulation of counts of *[i,j]*
  - Easy optimization
  - Only few improvement (large space)
    ```

Stripes Approach

- ```
  class Mapper
     method Map(null, items [i1, i2,...] )
        for all item i in [i1, i2,...]
           H = new AssociativeArray : item -> counter
           for all item j in [i1, i2,...]
              H{j} = H{j} + 1
           Emit(item i, stripe H)

  class Reducer
     method Reduce(item i, stripes [H1, H2,...])
        H = new AssociativeArray : item -> counter
        H = merge-sum( [H1, H2,...] )
        for all item j in H.keys()
           Emit(pair [i j], H{j})
  ```

ormat

- Fewer, larger intermediate keys
- Can lead to memory problems
- More complex implementation
  - Deuxième niveau
    - Troisième niveau

Seventh Outline Level Cliquez pour modifier les styles du texte du masque
  ```

Other exemples

- Grep
  - 10^10 100-byte records
  - Seek a rare 3 letters word
  - 1800 machines
  - Peak performance : 30 GB/s with 1764 workers
  - 150s
    - 1 minute startup
- Sort
  - Same environment and dataset
  - 50 lines of code
  - 891 seconds

Click to edit the outline text format
- Second Outline Level
  - Third Outline Level
    - Fourth Outline Level
      - Fifth Outline Level
        - Sixth Outline Level
          - Seventh Outline LevelCliquez pour modifier les styles du texte du masque
            - Deuxième niveau
              - Troisième niveau

Characteristics

- Manage well failure
  - Just send the keys again
- Heavy on the file system
  - Need dedicated and adapted filesystem
- Scale well
  - In term of data, workflow
- Easy to use
  - Some translation tools from SQL are available
- Middleware manages data and computing-
  - Deuxième niveau

Click to edit the outline text format
Second Outline Level
Third Outline Level
Fourth Outline Level
Fifth Outline Level
Sixth Outline Level
Seventh Outline LevelCliquez pour modifier les styles du texte du masque
Indisq

Some users

- Google
  - They normalized it
  - They use it internally
    - large-scale machine learning problems,
    - clustering problems for the Google News and Froogle products,
    - extracting data to produce reports of popular queries (e.g. Google Zeitgeist and Google Trends),
    - extracting properties of Web pages for new experiments and products (e.g. extraction of geographical locations from a large corpus of Web pages for localized search),
    - processing of satellite imagery data,
    - language model processing for statistical machine translation, and
    - large-scale graph computations.

Click to edit the outline text format
- Second Outline Level
  - Third Outline Level
    - Fourth Outline Level
      - Fifth Outline Level
        - Sixth Outline Level
- Seventh Outline LevelCliquez pour modifier les styles du texte du masque
  - Deuxième niveau
    - Troisième niveau

Other users

- Facebook
  - Hadoop
  - Now use Corona (own implementation)
- Yahoo
  - More than 100,000 CPUs in more than 40,000 computers
  - Hadoop
- Linkedin
  - 5000 servers on hadoop
- Ebay
  - 532 nodes cluster (8 * 532 cores, 5.3PB)

Click to edit the outline text format

Second Outline Level

Third Outline Level

Fourth Outline Level

Fifth Outline Level

Sixth Outline Level

Seventh Outline LevelCliquez pour modifier les styles du texte du masque

- Deuxième niveau
  - Troisième niveau

Some links

- Google
  - **MapReduce: Simplified Data Processing on Large Clusters** by *Jeffrey Dean and Sanjay Ghemawat*
  - Technical report
- Apache
  - **Hadoop: The definitive guide**
  - Book
- Microsoft
  - **Google's MapReduce Programming Model – Revisited**
  - Technical report

Click to edit the outline text format
Second Outline Level
Third Outline Level
Fourth Outline Level
Fifth Outline Level
Sixth Outline Level
Seventh Outline LevelCliquez pour modifier les styles du texte du masque
Deuxième niveau
Troisième niveau