Leveraging the MapReduce Application Model to Run Text Analytics in HPC Clusters

Cyril Briquet, McMaster University
Overview

1. Text Analytics with Voyeur Tools
2. HPC burst computing
3. MapReduce
4. Challenges
Text analytics with Voyeur Tools

Voyeur Tools

* interactive web app

* toolbox (word frequencies, collocates, concordances, ...)

* for text corpora (primary scope: Digital Humanities)

http://voyeurtools.org (you can try now)
Text analytics with Voyeur Tools
Overview

1. Text Analytics with Voyeur Tools
2. HPC burst computing
3. MapReduce
4. Challenges
HPC burst computing

* next generation of Voyeur Tools: 
  * **scale** to process large text corpora upon user request, 
    so-called “burst computing”

* use of **HPC resources** to process **bursts of user requests** 
  so that interactive web app remains **responsive**

* **SHARCNET-sponsored** project & Postdoctoral Fellowship 
  (started: January 2010)
Burst computing opportunities

3 opportunities to use HPC resources in Voyeur Tools:

* initial data **importing**

* data **indexing**

* data **analysis**

=> multithreading “OK”, but limited data parallelism
Overview

1. Text Analytics with Voyeur Tools
2. HPC burst computing
3. MapReduce
4. Challenges
MapReduce

Dean et al. 2004

Diagram showing the process of MapReduce with steps:
1. User Program forks
2. Master assigns map
3. Read from input files
4. Local write to intermediate files
5. Remote read by worker
6. Write to output files
MapReduce features [Dean et al. 2004]

* automatic parallelization and distribution

* I/O scheduling (+ data-aware scheduling)

* fault-tolerance

* status and monitoring
Using MapReduce in Voyeur Tools

* initial data importing:
  
  import text corpora into MapReduce filesystem

* data indexing: online or (preferably) batch indexing jobs

* data analysis: online analysis jobs
Overview

1. Text Analytics with Voyeur Tools
2. HPC burst computing
3. MapReduce
4. Challenges
Challenge: data model

impact of data model

* current Voyeur Tools: file-based
  (O.S.-level, “local” filesystem)
vs.

* MapReduce: record-based
  (application-level, “distributed” filesystem)
Challenge: computation model

impact of computation model

* current Voyeur Tools: multiple nested loops

vs.

* MapReduce: 2-phase record-based processing
Challenge: cluster job queue

impact of cluster job queue

* middleware daemons (for data distribution, computation) should ideally be always-on (to reduce web app latency)

* application malleability (to # available cores): supplementary challenge...
Challenge: global filesystem

impact of cluster-level global filesystem

* very convenient for many applications...

but we'd rather preposition data to local disks,

in order to maximize parallelism of data access
Challenge: firewalls

impact of firewalls

* can communicate <=> SHARCNET nodes, clusters

* cannot download data from the web
  (= limited parallelism of initial data import from web servers)
Challenge: public web front-end

requirement for public web front-end

* scalable servlet container

* DNS entry
Conclusion

* project **getting started**

* will rely on Open Source software:
  . Apache **Hadoop**
  . Apache Tomcat (maybe Eclipse Jetty)
  . and of course Voyeur Tools

* importance of **flexible data model**
Leveraging the MapReduce Application Model to Run Text Analytics in HPC Clusters

Cyril Briquet, McMaster University