CS 378 – Big Data Programming

Lecture 20

MetaPatterns
Review

• Assignment 9 – Job Chaining
  – Filter and bin sessions (same as assignment 8)
  – 3 jobs that process submitter, sharer, clicker bins
    • Can use the same map class
    • Compute stats for click types – over all sessions, not just sessions containing the click
  – Fourth job – aggregate click stats
    • Across the 3 session types
    • Across all click types (extra credit)
MetaPatterns

• We’ve discussed: Job chaining
  – Multiple jobs solving a multi-stage problem
  – When processing cannot be done in one job
  – When one output is input to multiple jobs

• Implemented in the run() method
Job Chaining

• Data pipelines often produce temporary files

• Output from one job that is input to another
  – As part of the pipeline, these files should be cleaned up
  – But you may want to keep them until the pipeline completes
  – Once complete, temp files can be deleted
Job Chaining - Scripting

• Another approach to managing job flow
  – Scripting languages
  – Shell scripts, python, ...

• Benefits
  – Changing the job flow does not require compilation
  – Script can use services and systems that are not Java
  – Easy to build flows between existing jobs
Chain Folding

- Basic patterns that can be “folded”:
  - Each record is submitted to multiple mappers
    - Combine these multiple map phases
  - Or to a reducer, then to a mapper
    - Push the map logic “upstream”

- Major benefit – reduce the amount of data moving through a data pipeline
  - Reduce disk I/O
  - Reduce data transfer (shuffle) over the network
Chain Folding

• Patterns that can benefit from folding

• In the data pipeline
  – Adjacent map phases might be merged

• Example:
  – Map only job, like a replicated join
  – Followed by map and reduce job

• Avoid writing the output of job one by joining the map logic of job one and two
Chain Folding

• Patterns that can benefit from folding

• A data pipeline ends with a map-only job

• Avoid reading the output of the penultimate job by merging the map logic of the final job into the previous reduce step
Chain Folding

• Split map phases between operations that
  – Decrease the amount of data (filtering)
  – Increase the amount of data (enrichment)

• Push the minimizing operation into previous reducer
  – This can reduce the amount of data transferred

• Generally, try to filter (minimize) data early
Chain Folding

Map: Filter out Teenager comments

Map: Tokenize remove stop words
Reduce: Word count

Map: Filter out Teenager comments, tokenize, remove stop words
Reduce: Word count
Chain Folding
Classes for Chaining

• ChainMapper
  – Specify a sequence of mappers
  – Output of one is input to the next
  – Arbitrary number can be “chained”

• ChainReducer
  – Specify the reducer
  – Specify a sequence of mappers
  – Arbitrary number of mappers can be “chained”