DATA 301, Big Data Computing and Systems

Lecture 1: Introduction 1: Big Data

Dr. James Atlas

Senior Lecturer
Computer Science and Software Engineering
University of Canterbury
james.atlas@canterbury.ac.nz

Today

- Course background, mechanics
- 5 V Challenges: Variety, Velocity, Volume, Veracity, Value
- 3 Perspectives: Architecture, Algorithms, Programming
- Algorithms: parallelism
- Programming: MapReduce in Spark

In class exercises

- Simple / short form today uses 3 roles:
  - Manager – ensures team is on task and on time
  - Recorder – records in written or diagram form team answers and discussion
  - Reporter – summarizes findings from the team orally to instructor or other teams
- Participant – All! Discussion, sharing ideas, solving problems is a joint responsibility.

Who are you? 3 minutes

Introduce self to your group

year ?

curriculum / program ?

What interests you about big data? Computing vs systems?

Who am I?

- 5th year at University of Canterbury
  - 9 years as faculty at University of Delaware
- Research background: Artificial Intelligence (multi-agent system coordination)
- Taught 14+ different courses to > 5000 students

Current research efforts in

- AI/ML + …
  - Medical Data
  - Geospatial Data
  - Space Exploration
  - Physics integrated ML

Recent efforts in

- Computer Science Education
- Parallel / Scientific Computing

Course Background

A 3rd/4th year course level in a highly technical computational field, both in theory and practice

Students

- Computer Science, Data Science, Software Engineering

Prerequisites

- **Algorithms** (COSC 122 and 262)
  - Dynamic programming, basic data structures
- **Basic linear algebra and probability** (MATH 102 and 120)
  - Distributions, matrix analysis, …
- **Programming** (COSC 121 and COSC 122)
  - Python

Acknowledgements

- University of Delaware CISC372
- Stanford University CS246
  http://www.mmds.org
- University of Utah CS 5965

Course Mechanics (see syllabus)

- **Weekly lab assignments: 30%**
  - Short e-quizzes on Learn
  - Longer programming lab each of the first 8 weeks (8 lab assignments total)
- **Project: 40%**
  - We'll talk more about this next week
- **Final exam: 30%**

- **It's going to be fun and hard work. ☺**

Week One Learning Goals

- Explain five challenges for big data computing: **Variety, Velocity, Volume, Veracity, Value**
- Recognize three perspectives in **Big Data Computing**: architecture, algorithms, programming
- Recognize the hierarchical relationship between a **cluster computing architecture** and a single node architecture
- Investigate how the **MapReduce** model addresses architecture, algorithms, and programming

Big Data Computing and Systems

The recognition that data is at the center of our digital world and that there are big challenges in collecting, storing, processing, analyzing, and making use of such data.

Data Mining ≈ Big Data Computing ≈ Predictive Analytics ≈ Data Science

(and recently, < ≈ Artificial Intelligence/Machine Learning)

Big Data Systems include and are closely related to Parallel Computing, Cloud Computing, GPU/TPU Computing

VOLUME

Huge amount of data

VERACITY

Inconsistencies and uncertainty in data

VARIETY

Different formats of data from various sources

VELOCITY

High speed of accumulation of data

VALUE

Extract useful data

Perspectives for Week One

- Architecture
  - Von Neumann architecture
  - Large Scale Cluster Computing
- Algorithms
  - **Parallelism**
  - Scalability
  - Distributed Data (File Systems)
- Programming
  - **Map Reduce**

Algorithms Perspective

Algorithms Perspective

- Exercise: Parallelism

compute $accum = \sum_1^n \sqrt{A_i}$

https://goo.gl/ds2RnW

Programming Perspective

Programming Model: MapReduce

- **Challenge: Productivity**
  - **How can we make it easy to write distributed programs?**

- **Provide a functional programming interface that wraps the architecture and algorithm details**
  - MapReduce (2004, Google)
  - Spark (2014, UCB now Apache)

MapReduce: Overview

- Load a lot of data into a structure
- **Map:**
  - Extract something you care about
- **Group by key:** Sort and Shuffle
- **Reduce:**
  - Aggregate, summarize, filter or transform
- Write the result

Outline stays the same, **Map** and **Reduce** change to fit the problem

MapReduce: Our sum of square roots

- Load a lot of data into a structure
  `A = [1, 2, 3, 4, 5, 6, 7, 8]`
- **Map:**
  - Extract something you care about

  `items = map(sqrt, A)`
- **Group by key:** Sort and Shuffle

  not needed in this example
- **Reduce:**
  - Aggregate, summarize, filter or transform

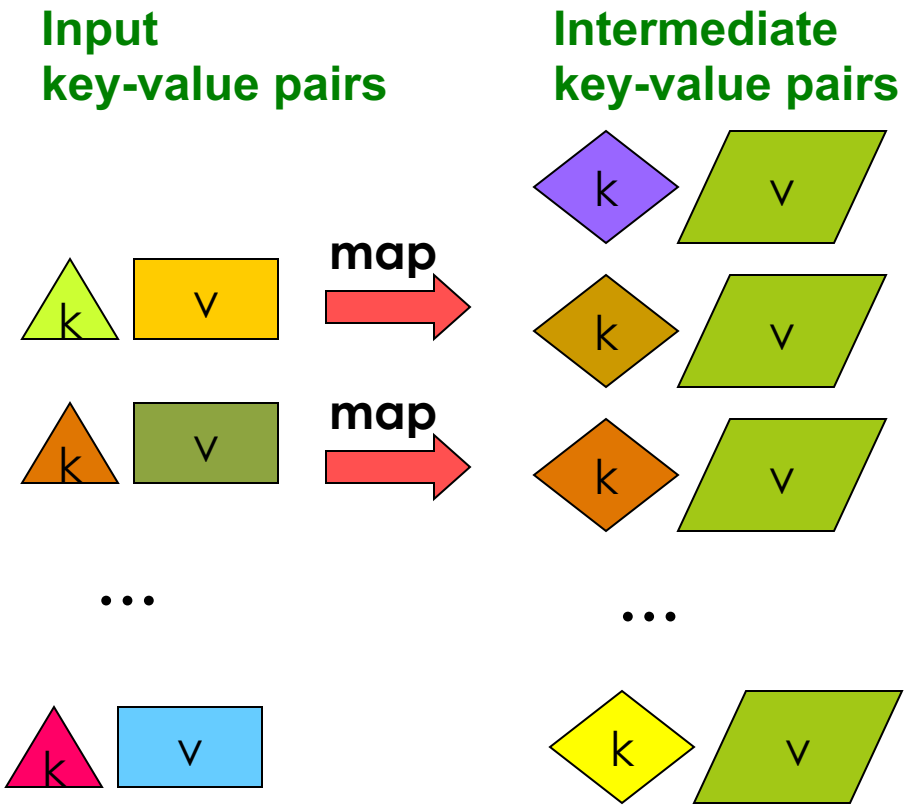
  `accum = reduce(lambda x, y: x+y, items)`
- Write the result
  `print(accum)`

Programming Model: MapReduce

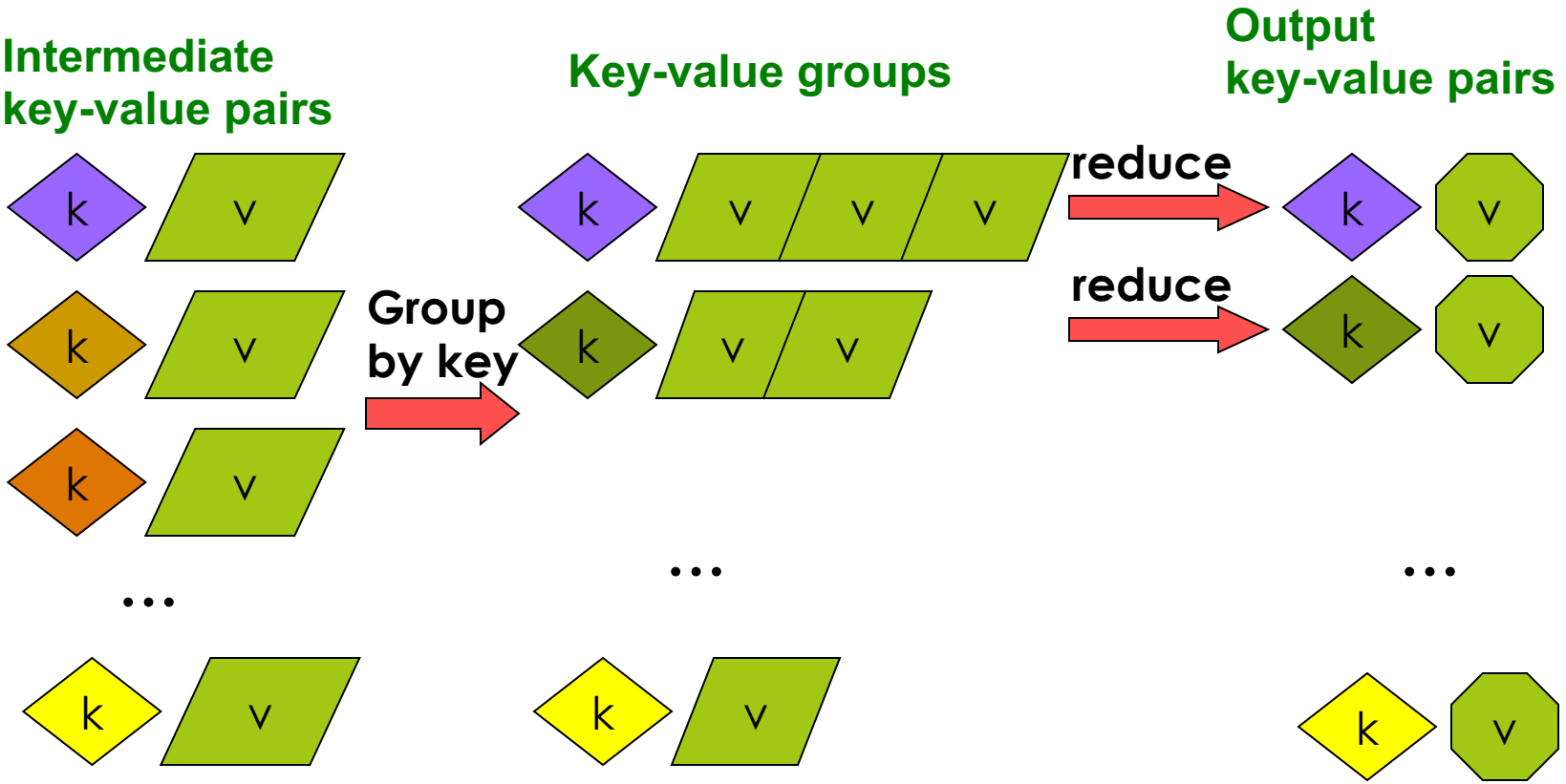
Canonical Example

- We have heaps of text documents
- How can we count the number of times each distinct word appears in the documents using MapReduce?
- **Sample application:**
  - Analyze web server logs to find popular URLs

MapReduce: The Map Step



MapReduce: The Reduce Step



MapReduce: Word Counting