CS61A Lecture 30

MapReduce

Jom Magrotker
UC Berkeley EECS
August 8, 2012
COMPUTER SCIENCE IN THE NEWS

Touch Your Philodendron and Control Your Computer: Technology Turns Any Plant Into an Interactive Device

http://www.sciencedaily.com/releases/2012/08/120806094053.htm
http://www.disneyresearch.com/research/projects/hci_botanicus_drp.htm
TODAY

• MapReduce
  – Description
  – Examples

• Social implications of Computing
  – Therac-25 case study
RECAP: REDUCE

\[
\text{reduce(lambda so\_far, elt: so\_far * elt, [1, 3, 8, 7], 2)}
\]

\[
((((2 * 1) * 3) * 8) * 7)
\]
MAP AND REDUCE

Many problems can be solved by mapping a function over a sequence, and then reducing the new items.

To find the sum of squares of items in a list:

```python
from functools import reduce
from operator import add

def sum_squares(list):
    return reduce(add, map(lambda num: num ** 2, list), 0)
```

1. Square each item in a list...
2. ... and add the squared items together.
MAP AND REDUCE: EXAMPLE

Count the number of letters in the words of a list

def count_letters(wordlist):
    return \n    reduce(add, 
    map(lambda word: len(word), wordlist),
    0)

1. Find the length of each word in a list...
2. ... and add the lengths together.
MAP AND REDUCE: EXAMPLE

Count the number of words in a list

def count_words(wordlist):
    return \n    reduce(____,
            map(______________,
                 wordlist),
             0)
**Map and Reduce: Example**

*Count the number of words in a list*

```python
def count_words(wordlist):
    return reduce(add,
                 map(lambda word: 1, wordlist),
                 0)
```
MAP AND REDUCE: EXAMPLE

Count the number of words in a list of even length

def count_words(wordlist):
    return \n    reduce(___,
           reduce(___,
                   map(______________________________,
                        wordlist),
                   0)
MAP AND REDUCE: EXAMPLE

Count the number of words in a list of even length

def count_words(wordlist):
    return \n    reduce(add,
    map(lambda word: 1 if len(word)%2 == 0 else 0, wordlist),
    0)
MAP AND REDUCE: EXAMPLE

Count the number of items in a list that satisfy a given predicate

def count_pred(pred, list):
    return \n    reduce(___,
        reduce(___,
            map(___________________________________,
                list),
            0)
**Map and Reduce: Example**

**Count the number of items in a list that satisfy a given predicate**

```python
def count_pred(pred, list):
    return \n    reduce(add,
    reduce(lambda item: 1 if pred(item) else 0, list),
    0)
```
MAP AND REDUCE: EXAMPLE

Filter out the items in a list that satisfy a given predicate

def filter(pred, list):
    return \
    reduce(________________________________________, 
        map(lambda item: _____ if pred(item) \ 
            else ___, 
            list), 
    __)

MAP AND REDUCE: EXAMPLE

Filter out the items in a list that satisfy a given predicate

def filter(pred, list):
    return \
    reduce(lambda so_far, sublist: so_far + sublist,
            map(lambda item: [item] if pred(item) \n                 else [],
                 list),
            [])
ANNOUNCEMENTS

• You are now done with all projects and homework assignments! Congratulations! 😊

• Final lecture: You choose!
  Piazza poll will be sent out today asking for suggestions for tomorrow’s lecture.
ANNOUNCEMENTS: FINAL

• Final is **Thursday, August 9**.
  – *Where?* 1 Pimentel.
  – *When?* 6PM to 9PM.
  – *How much?* *All* of the material in the course, from June 18 to August 8, will be tested.

• Closed book and closed electronic devices.

• One 8.5” x 11” ‘cheat sheet’ allowed.

• No group portion.

• Post-final laser tag?
ANNOUNCEMENTS: FINAL

http://berkeley.edu/map/maps/BCS6.html
As of 2007, Google processes 20 petabytes of data per day. Now, they probably process thousands of petabytes per day.

How much is a petabyte?

**How Much Data?**

Assume that every 0 or 1 in a file is as large as a pixel on a screen (approx. 100 $\mu$m$^2$).

Then, a homework file, of size (approximately) 5 *kilo*bytes, is about as big as

-
A music file has (approximately) 6 megabytes. (1 megabyte = 1024 kilobytes)
HOW MUCH DATA?

One DVD-R has (approximately) 5 gigabytes.  
(1 gigabyte = 1024 megabytes)

This is about as big as the total area of all pages of the paper copy of a homework.
HOW MUCH DATA?

One terabyte is 1024 gigabytes.
One petabyte is 1024 terabytes.

This is as much information as would fit in:
6 billion photos on Facebook, or around 13 years of HD video.

HOW MUCH DATA?

One petabyte is an extremely large amount of data. We cannot process this on one computer alone.

My computer would take around 7 years to process all of 20 petabytes exactly once, and that assumes that all it ever does is process data and never fails.

Companies like Google do not have this luxury: they need data to be processed as quickly as possible.

**PARALLELISM**

**Idea:** Have a lot of computers working at once on different chunks of the same data, independently of other chunks and computers.
MAPREDUCE

All data is available at the outset; results are not consumed until processing completes

Framework for batch processing of Big Data

System used by programmers to build applications

Very large and complex data sets
**MapReduce: History**

- Developed at Google in 2003.
- Programmers had to consider a lot of details as part of their code: what to do when a computer fails, and how to divide the work up among different computers.
- However, many operations fit neatly into a map step, followed by a reduce step.
The MapReduce framework is an *abstraction*: it allows programmers to code, while taking care of the underlying messy details, such as dealing with failure and distributing work across computers.

All the programmers had to do was create:

- a *mapper* (the function to map), and
- a *reducer* (the function to combine mapped results).
MAPREDUCE: EXECUTION MODEL

MAPPER

Intermediate: k1:v k1:v k2:v k1:v k3:v k4:v k4:v k5:v k4:v k1:v k3:v

GROUP BY KEY

GROUPED: k1:v,v,v,v k2:v k3:v,v,v k4:v,v,v,v k5:v

REDUCER

Output

**MapReduce**

**Map Phase**
Apply a *mapper* over inputs, and emit intermediate *key-value pairs*.

**Sort Phase**
Group the intermediate *key-value pairs* by *keys*.

**Reduce Phase**
For each intermediate *key*, apply a *reducer* to “accumulate” all values associated with that key.
**MapReduce**

Running example
Count all the vowels in Shakespeare’s works.

We need a mapper and a reducer.

**Idea:**
The mapper will count the number of vowels in each line, for each vowel.
The reducer will take the counts for each vowel from each line and total all the counts.
“To be or not to be”

MAPPER

e: 2, o: 4

“Brevity is the soul of wit”

MAPPER

e: 2, i: 3, o: 2, u: 1

“What is past is prologue”

MAPPER

a: 2, e: 1, i: 2, o: 2, u: 1

SORTER

a: [2], e: [2, 2, 1], i: [3, 2], o: [4, 2, 2], u: [1, 1]

REDUCER

a: 2, e: 5, i: 5, o: 8, u: 2
import sys
from ucb import main
from mr import emit

def emit_vowels(line):
    for vowel in 'aeiou':
        count = line.count(vowel)
        if count > 0:
            emit(vowel, count)

@main
def run():
    for line in sys.stdin:
        emit_vowels(line)

1. For every line in a file, fed through the standard input of this program...

2. Check if the line has a vowel.

3. If it does, “emit” – or send to the standard output – a key-value pair that associates the vowel with its count.
import sys
from ucb import main
from mr import emit, values_by_key

@main
def run():
    for key, value_iterator in values_by_key(sys.stdin):
        emit(key, sum(value_iterator))
STANDARD INPUT AND OUTPUT

The *standard input* is where the program receives its input from external sources, such as the user or a file.

The *standard output* is where the program will send its output, such as the screen or another file.
**MapReduce: Under-the-Hood**

Different computers have copies of the mapper, and work on pieces of the data simultaneously.

Key-value pairs are sent to different machines based on keys.

[Diagram showing the process of MapReduce](http://research.google.com/archive/mapreduce-osdi04-slides/index-auto-0008.html)
What is a mapper and reducer that we can use to count the number of times particular words appear in a file?

**Mapper**: For every line in a file, emit...

**Reducer**: For every key,
**MAPREDUCE: PRACTICE**

What is a mapper and reducer that we can use to count the number of times particular words appear in a file?

**Mapper:** For every line in a file, emit a key-value pair that associates a “special” word with the number of times it appears in that line.

**Reducer:** For every key, sum up all the values associated with that key.
**MAPREDUCE: PRACTICE**

What is a **mapper** and **reducer** that we can use to count the number of lines in a file?

**Mapper**: For every line in a file, emit...

**Reducer**: For every key,
**MapReduce: Practice**

What is a **mapper** and **reducer** that we can use to count the number of lines in a file?

**Mapper**: For every line in a file, emit the key-value pair (“line”, 1).

**Reducer**: For every key, sum up all the values associated with that key.
**MapReduce**

What does the framework provide us?

- **Fault Tolerance**: A machine or hard drive might crash.
  The framework will automatically re-run failed tasks.
- **Speed**: Some machine might be slow because it is overloaded.
  The framework can run multiple copies and keep the result of the one that finishes first.
What does the framework provide us?

• **Network locality**: Data transfer is expensive.
  The framework tries to schedule map tasks on the machines that hold the data to be processed.

• **Monitoring**: When will my job finish?
  The framework provides a web-based interface describing jobs.
**BREAK**

There's been a lot of confusion over 1024 vs 1000, kbyte vs kbit, and the capitalization for each. Here, at last, is a single, definitive standard:

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>NAME</th>
<th>SIZE</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>kB</td>
<td>KILOBYTE</td>
<td>1024 BYTES OR 1000 BYTES</td>
<td>1000 BYTES DURING LEAP YEARS, 1024 OTHERWISE</td>
</tr>
<tr>
<td>KB</td>
<td>KELLY-BOOTLE STANDARD UNIT</td>
<td>1012 BYTES</td>
<td>COMPROMISE BETWEEN 1000 AND 1024 BYTES</td>
</tr>
<tr>
<td>KiB</td>
<td>IMAGINARY KILOBYTE</td>
<td>1024.51 BYTES</td>
<td>USED IN QUANTUM COMPUTING</td>
</tr>
<tr>
<td>kb</td>
<td>INTEL KILOBYTE</td>
<td>1023.937528 BYTES</td>
<td>CALCULATED ON PENTIUM FPU.</td>
</tr>
<tr>
<td>Kb</td>
<td>DRIVEMAKER'S KILOBYTE</td>
<td>CURRENTLY 908 BYTES</td>
<td>SHRINKS BY 4 BYTES EACH YEAR FOR MARKETING REASONS</td>
</tr>
<tr>
<td>KBa</td>
<td>BAKER'S KILOBYTE</td>
<td>1152 BYTES</td>
<td>9 BITS TO THE BYTE SINCE YOU'RE SUCH A GOOD CUSTOMER</td>
</tr>
</tbody>
</table>

---

Spelling of Shakespeare's name

From Wikipedia, the free encyclopedia

The spelling of William Shakespeare's name has varied over time. It was not consistently spelled any single way during his lifetime, in manuscript or in printed form. After his death the name was spelled variously by editors of his work and the spelling was not fixed until well into the 20th century.

The standard spelling of the surname as "Shakespeare" was the [lefthandedtoons.com](http://www.lefthandedtoons.com/1271/)

http://xkcd.com/394/

http://www.cartoonstock.com/newscartoons/cartoonists/jco/lowres/jcon1499l.jpg

---

http://www.cartoonstock.com/newscartoons/cartoonists/jco/lowres/jcon1499l.jpg
The web is full of links, but there’s no way I’m going to visit them all. So who’s to say those pages actually exist? Maybe they’re just there to take me out and give me the IMPRESSION we built a dang ol’ world wide web!

Who’s to say I’m not the only person using the Internet??

I’m serious! It’s possible that I’m just getting lucky and hitting the few pages that people created for me in advance!

Or maybe... we’re simply creating them on the fly when you request them?

Oh my gosh that finally explains why my internet is so slow!!

You guys take like forever to write this stuff out!

This is ridiculous, dude!

Is it? Is it REALLY?

Take wikipedia: how can I get mad at the errors when articles are being pulled together in like two seconds, which I guess isn’t that bad? And when the page is vandalized to read “butts lol ;)

That’s just a desperate stalling tactic for more time!

Well, if this paranoia helps you calm down about wikipedia, I’m all for it!

Paranoia?? Man, I’m proposing a vast worldwide conspiracy where people work to help me out on demand and let me learn about boats and pokémon.

If this be paranoia, I would like a double-down refill please!

(C) 2012 Ryan North

THERAC-25

Figure 1. Typical Therac-25 facility

http://computingcases.org/images/therac/therac_facility.jpg
THERAC-25

• Built by AECL (Atomic Energy of Canada Limited)
• Dual-mode machine:
  – Treat using low-energy electron beams.
  – Treat using high-energy X-ray beams.
• Turntable with items for each mode placed on it: rotates to the correct position before a beam is started up.
THERAC-25

What happened?

• High-power beam activated instead of the low-power beam.
• High-power beam had approximately 100 times the intended dose of radiation.
• Six accidents between 1985 and 1987.
• Three deaths from radiation poisoning.

Sources:
THERAC-25

Why? Software bugs and user interface issues.

• A one-byte counter frequently “overflowed”, or got too big for the memory allocated to it. If an operator provided manual input at the same time as the overflow, the interlock would fail.

• Failure only occurred during a nonstandard sequence of keystrokes: this went unnoticed for a long time.

Sources:
THERAC-25

Why? Software bugs and user interface issues.

• Therac-25 did not have the hardware interlocks that its predecessors had, to prevent the wrong beam from activating.

• The developers relied on software interlocks, which were incorrectly coded.

• The code reused software from older models, where the hardware interlocks could not report that they were triggered.

Sources:
Why? Software bugs and user interface issues.

• The combination of hardware and software was never tested until at the hospital.
• If the system noticed that something was wrong, it would halt the X-ray beam and display the word “MALFUNCTION”, followed by an unhelpful error code.
  The operator could override the warning and proceed anyway.
• Complaints were not believed initially.

Sources:
THERAC-25

Thus, for any nondeterministic Turing machine $M$ that runs in some polynomial time $p(m)$, we can devise an algorithm that takes an input $w$ of length $n$ and produces $E_{n,w}$. The running time is $O(p^2(m))$ on a multitape deterministic Turing machine and...

WTF, MAN. I JUST WANTED TO LEARN HOW TO PROGRAM VIDEO GAMES.

SIPSER CH 7

$N_1 = (A_1 \lor B_1) \land (A_1 \lor B_1) \land \cdots$

$N = N_0$
THERAC-25

What can we learn?

• Computers are everywhere: this means that the code you write can potentially be used in a life-or-death situation.

• In a complex codebase, software bugs can be really hard to catch and fix.

• This can be ameliorated significantly with proper design and frequent testing.
What can we learn?

- Test not just the software, but how it interacts with hardware and even the user.
- The user is not your enemy.
  - A significantly large project will account for many possible erroneous inputs from the user.
  - The user should be provided useful output to understand the cause for error.
CONCLUSION

• MapReduce is a framework that allows programmers to process large amounts of data in parallel, without having to concern themselves with details such as computer failure and task assignment.

• Proper design and test of software is becoming increasingly essential, especially in a world where computing is ubiquitous.