CS61A Lecture 9
MapReduce

2011-07-05
Eric Tzeng

iClicker activity!
Vote for whichever letter you think will have the
least votes by the time class starts!
Feel free to ask each other what you’re voting
for (but obviously you don’t have to tell the
truth!)

Reminder: accumulate

STk> (trace accumulate)
STk> (accumulate + 0 '(1 2 3 4))
(+ 1 (accumulate + 0 '(2 3 4)))
(+ 1 (+ 2 (accumulate + 0 '(3 4))))
(+ 1 (+ 2 (+ 3 (accumulate + 0 '(4)))))
(+ 1 (+ 2 (+ 3 (+ 4 (accumulate + 0 '())))
(+ 1 (+ 2 (+ 3 4))))
(+ 1 (+ 2 7))
(+ 1 9)
10

Reminder: accumulate (con’t)

STk> (trace accumulate)
STk> (accumulate cons '(a b c))
(cons 'a (accumulate cons '(b c)))
(cons 'a (cons 'b (accumulate cons '(c))))
(cons 'a (cons 'b (cons 'c (accumulate cons '()))))
(cons 'a (cons 'b 'c))
(cons 'a 'b c)
'(a b c)

Reminder: accumulate (con’t)

• Potential asymmetry in arguments:

(accumulate
(lambd (next so-far) (+ next so-far))
0
'(1 2 3))
always a
number
always a
number

Arguments are
not the same
type!

Reminder: accumulate (con’t)

• Potential asymmetry in arguments:

(accumulate
(lambd (next so-far) (cons next so-far))
'()
'(1 2 3))
always a
number
always a
list
Higher Order Function Review!

For these questions, use only higher order functions!

1. Write a procedure `product-of-squares` that takes a list and returns the product of the squares of each element.
   STk> (product-of-squares '(1 2 3))
   36

2. Write a procedure `letter-count` that counts the number of letters in a list of words.
   STk> (letter-count '(hello there))
   10

Intro to MapReduce

Google

2005 – 200 terabytes of information indexed

200 Terabytes?

one homework file – 5 kb
(5 x 10^6 bytes)

one music file – 6 MB
(1228.8 homeworks)

200 Terabytes?

one mp3 file – 6 MB
(1228.8 homeworks)

one blu ray disc – 25 GB
(5.2 x 10^9 homeworks)
(4267 mp3 files)

Tangent: how big is Wikipedia?

Wikipedia is only 16 gigabytes!
200 Terabytes?
- one blu ray disc – 25 GB
  (5.2 x 10^9 homeworks)
  (4267 mp3 files)
- Google’s index size as of 2005 – 200 TB
  (4 x 10^9 homeworks)
  (155 x 10^9 mp3 files)
  (8192 blu ray discs)

“Between the birth of the world and 2003 there were 5 exabytes of information created [...] [now] we create 5 exabytes in two days.”
- Eric Schmidt, Former Google CEO, 2008

The normal way

(why so serious)

12

count-letters

The parallel way

(why so serious)

12

Splitting up and managing the work is hard!

Let’s take a look at that again...

(define (product-of-squares ls)
  (accumulate * 1
   /(map square ls))))

(define (letter-count ls)
  (accumulate + 0
   /(map count ls))))

The basic idea

(accumulate + 0
  (map count '(why so serious)))

(kind of) MapReduce
The basic idea (map phase)

\[
\text{accumulate } + \emptyset \\
\text{(map count '(why so serious))}
\]

The "mapper"
• Procedure of one argument

The basic idea (reduce phase)

\[
\text{accumulate } + \emptyset \\
\text{(map count '(why so serious))}
\]

The "reducer"
• Procedure of two arguments
• Base-case

The "mapper"
• Procedure of one argument

The basic idea

\[
\text{define (kinda-mapreduce mapper reducer base input)} \\
\text{(accumulate reducer base (map mapper input))}
\]

Break time!

MapReduce

Key-Value Pairs

\[
\text{define make-kv-pair cons} \\
\text{(define kv-key car)} \\
\text{(define kv-value cdr)}
\]

Reminder: respect the abstraction!
mapreduce – map phase

mapper

key → MAP → (v1, v2, v3)

map phase - example

What mapper will perform this transformation?

’((hello goodbye) you say goodbye but I say hello)

map phase - example

’((you . 1) (say . 1) (goodbye . 1) (but . 1) (1 . 1) (say . 1) (hello . 1))

mapreduce – sort phase

(map phase) → (v1, v2, v3) → SORT → (v1, v2, v3)

sort phase - example

’((you . 1) (say . 1) (goodbye . 1) (but . 1) (1 . 1) (say . 1) (hello . 1))

sort phase - example

’((you . 1) (say . 1) (goodbye . 1) (but . 1) (1 . 1) (say . 1) (hello . 1))

mapreduce – reduce phase

key → reducer → base

reduce phase - example

say + φ (say 1 1 1) → (1 1 1) → REDUCE → 3 → (say . 3)
reduce phase – example 2

(mapreduce)

True (A) or false (B)?
1. The reducer is a procedure that takes a list of values and combines them.
2. The mapper can act as a filterer.
3. The kv-pairs that the mapper produces will have the same keys as the final output kv-pairs.

“Forsooth, a mapper!”
(define (forsooth-mapper kvp)
  (map
    (lambda (wd) (make-kv-pair (kv-key kvp) wd))
    (filter
      (lambda (wd) (equal? wd ‘forsooth))
      (kv-value kvp)))))

“Alack! A reducer!”
(define (forsooth-reducer next so-far)
  (+ next so-far))

;; alternatively...
(define forsooth-reducer +)

mapreduce example
Input:
- a set of kv-pairs of the form
  `((play title) . (line from that play))`

Output:
- the number of letters in each play, not counting the word “forsooth”

Go over all the parts one more time with your neighbor!
Putting it all together

• All that’s left to do is to make our final call:

```
STk> (mapreduce forsooth-mapper ; mapper
    forsooth-reducer ; reducer
    0 ; base-case
    "/gutenberg/shakespeare") ; input
```

Why mapreduce?

<table>
<thead>
<tr>
<th>mapper</th>
<th>reducer</th>
<th>base-case</th>
</tr>
</thead>
<tbody>
<tr>
<td>above the line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>below the line</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Splitting up work
- Process management
- Failure handling
- etc...

7/5/11