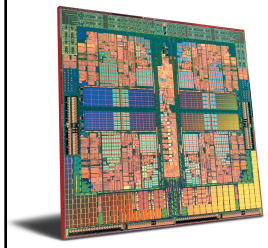


Some material dapted from slides by Steve Freund and Dan Grossman

Parallelism vs. Concurrency

Key concerns

Parallelism:

Use extra resources to solve a problem faster

Concurrency:

Correctly and efficiently manage access to shared resources

May overlap

2

An analogy

CS1 idea: A program is like a recipe for a cook

- One cook who does one thing at a time! (*Sequential*)

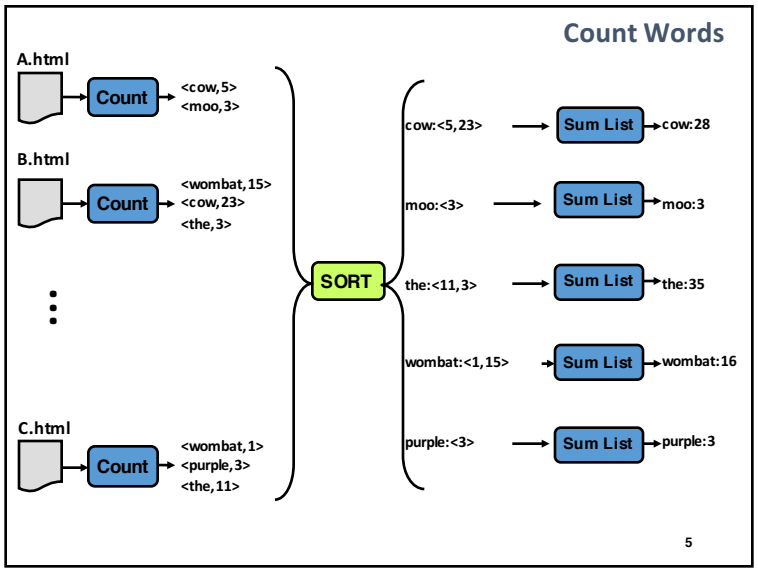
Parallelism:

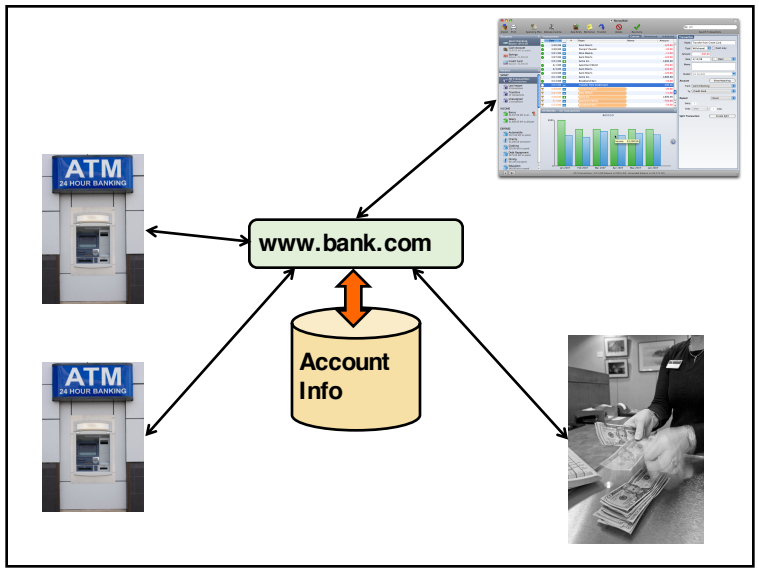
- Have lots of potatoes to slice?
- Hire helpers, hand out potatoes and knives
- But too many chefs and you spend all your time coordinating

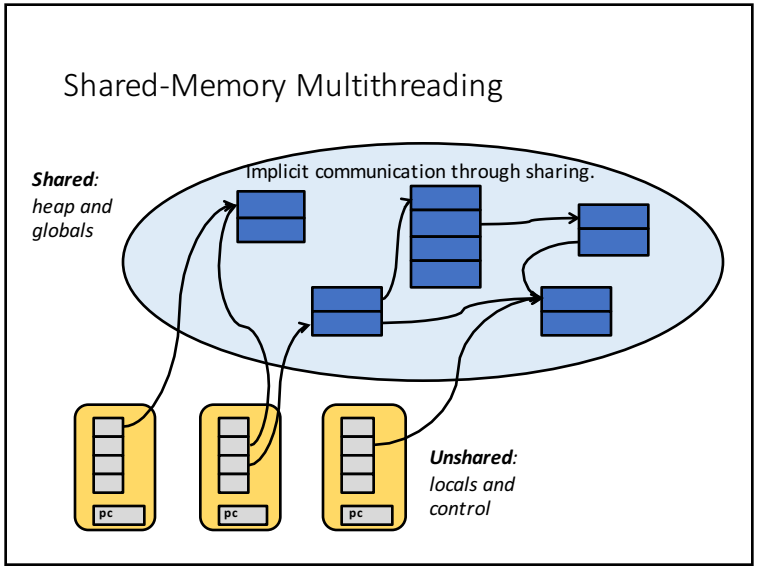
Concurrency:

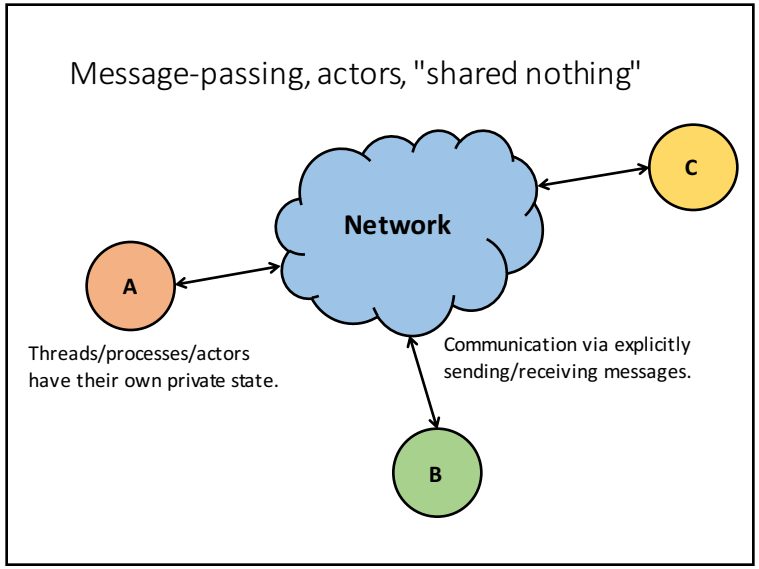
- Lots of cooks making different things, but only 4 stove burners
- Want to allow access to all 4 burners, but not cause spills or incorrect burner settings

3

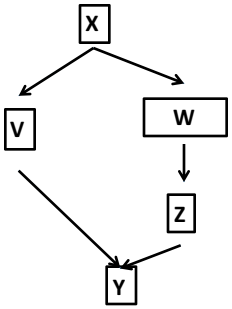

Count Words
A.html
Count
<cow,5>
<moo,3>
B.html
Count
<wombat,15>
<cow,23>
<the,3>
C.html
Count
<wombat,1>
<purple,3>
<the,11>
SORT
cow:<5,23>
moo:<3>
the:<11,3>
wombat:<1,15>
purple:<3>
Sum List
cow:28
moo:3
the:35
wombat:16
purple:3
5


ATM
24 HOUR BANKING
www.bank.com
Account Info


Shared-Memory Multithreading
Shared: heap and globals
Implicit communication through sharing.
pc
pc
pc
Unshared: locals and control


Message-passing, actors, "shared nothing"
Network
A
B
C
Threads/processes/actors have their own private state.
Communication via explicitly sending/receiving messages.

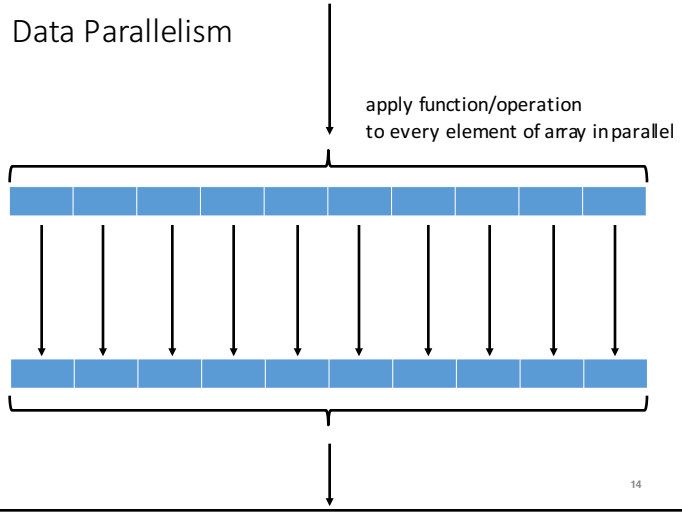
Dataflow



Programs are DAGs.
Nodes are functions/operations.
Node executes after predecessors.

13

Data Parallelism



14

```
do i=1, n
      z(i) = x(i) + y(i)
enddo
```

```
z(1) = 1
do i=2, n
      z(i) = z(i-1)*2
enddo
```

```
z(1) = 1
do i=2, n
      z(i) = z(1)*2**(i-1)
enddo
```

```
cobegin
  x = sumRange(0,n/2)  ||  y = sumRange(n/2,n)
end

z = x + y
```