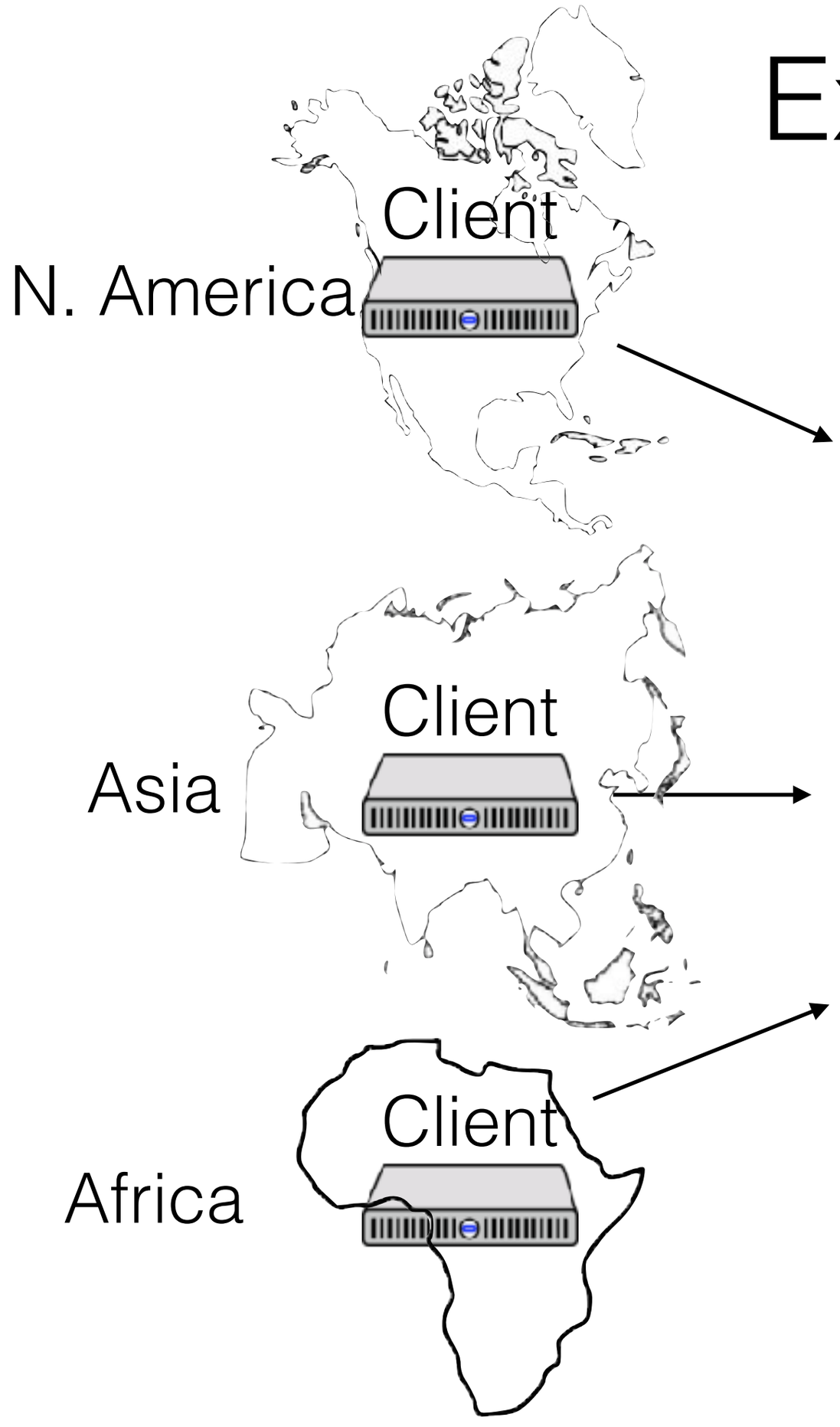
Implementing caches

Doug Woos

Logistics notes

Textbook chapter for Friday—no (graded) discussion

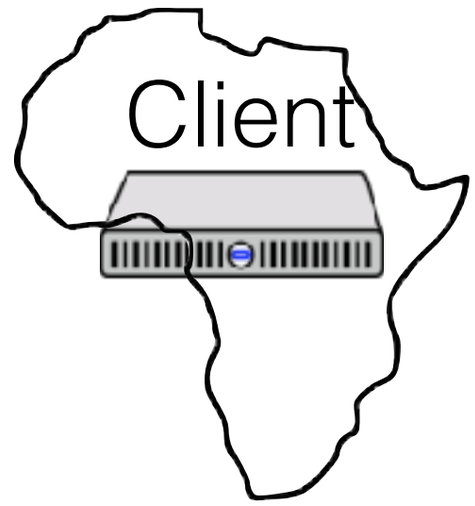
Example



System
+
Caches


```
put (k1, f(data))
put (done1, true)
```

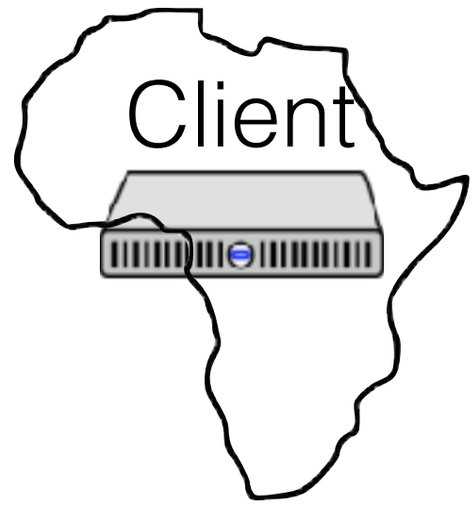

```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```




```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```




```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

Can we start done1 write before k1 write is complete?

Yes, provided we're talking to a single server

Sharding

Sharding

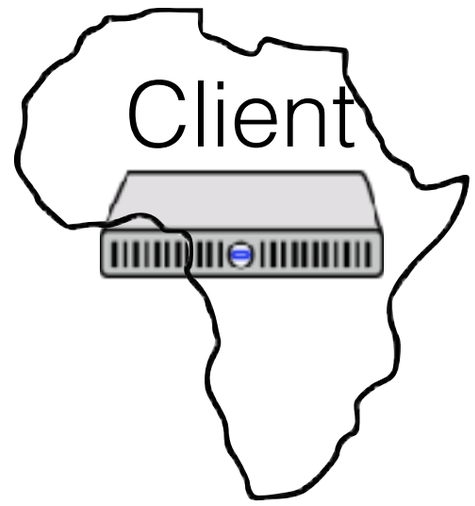
- Different data on different servers
- Partitioned via some function on keys
- Implement in Lab 4!

Another axis of scaling, cf. replication

- Usually, shard then replicate
- Each piece of data lives on one replicated shard


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```



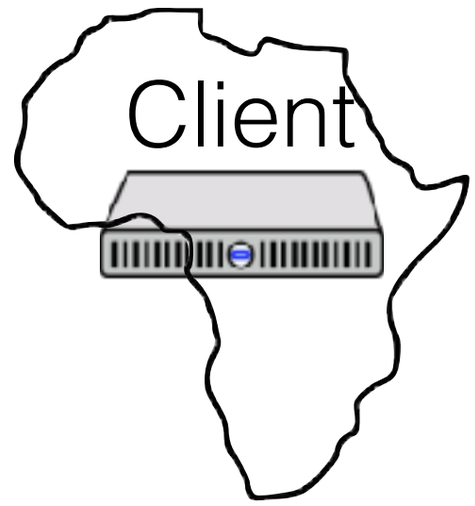

```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

Can we start done1 write before k1 write is complete, if done1 and k1 live on different servers?

Not if we want sequential consistency!


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```




```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

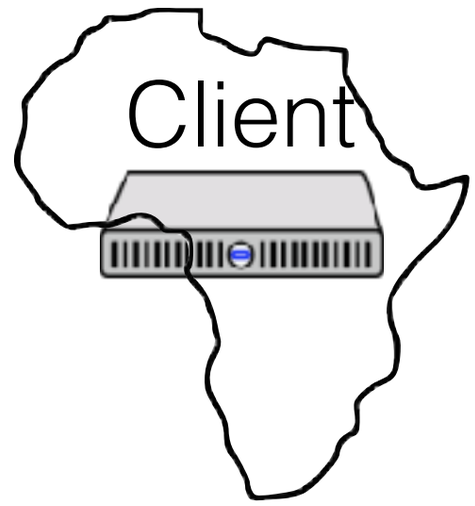
Can we start done1 write before k1 write is complete, if done1 and k1 live on different servers?

Not if we want sequential consistency!

- Must serialize writes


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```

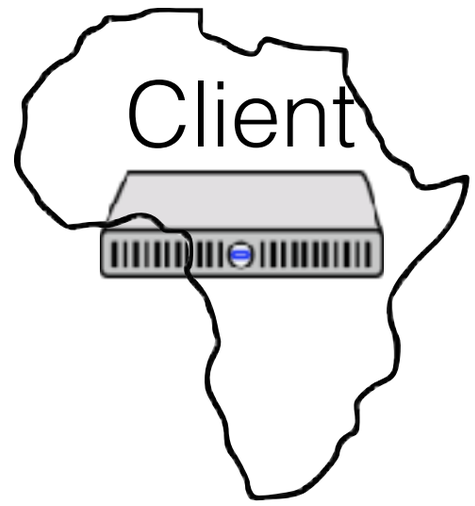



```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

What if there are caches? What might go wrong?


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```




```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

- Asia sees updated done1, cached k1
- Africa sees updated done2, cached k1 and k2
- Africa sees updated done2, k2, cached k1 (!)

Rule for caches and shards

Due to Sarita Adve

Suppose each process specifies ops in some order

Sequentially consistent if:

1. Ops applied in processor order, and

2. All ops to a single key are serialized (as if to a single copy)

So how do we ensure ops go to a single copy?

Invalidations vs. Leases

Invalidations

- Track where data is cached
- When doing a write, invalidate all (other) locations
  - Data can live in multiple caches for reading

Leases

- Permission to serve data for some time period
- Wait until lease expires before applying updates
- Must account for clock skew!

Write-through vs. write-back

Write-through

- Writes go to the server
- No modified caches

Write-back

- Writes go to cache
- Dirty cache written to server when necessary

Write-through vs. write-back

Mechanism Write policy	Invalidations	Leases
Write-through	AFS (Andrew FS)	DNS
Write-back	Sprite	NFS

Write-through invalidations

Track all reading caches

On a write:

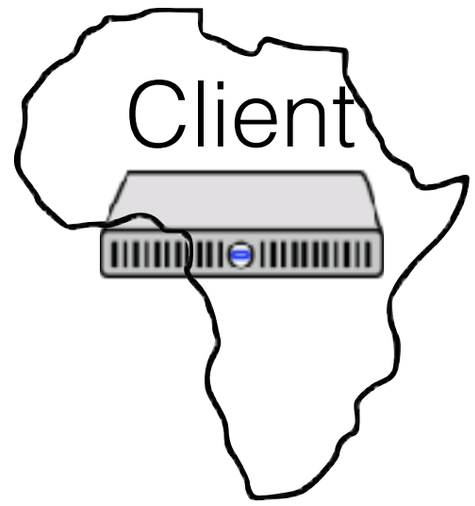
- Send invalidations to all caches
- Each cache invalidates, responds
- Wait for all invalidations, do update
- Return

Reads can proceed:

- If there is a cached copy
- If no write waiting at server


```
put (k1, f(data))
put (done1, true)
```

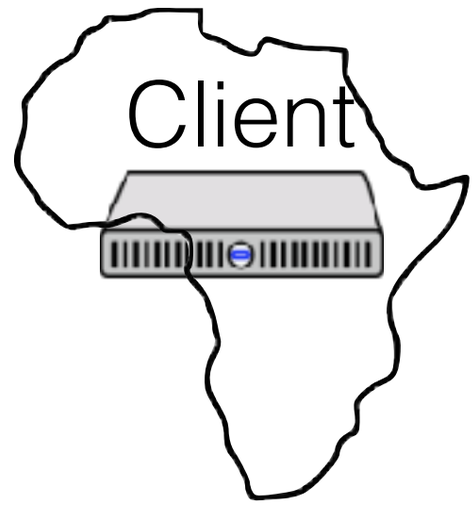

```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```




```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```



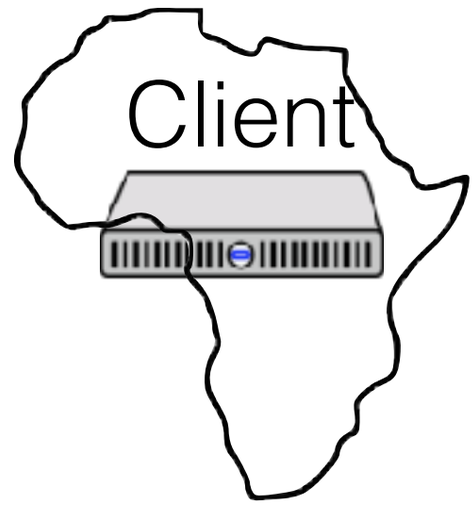

```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

done1=false


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```

done1=false



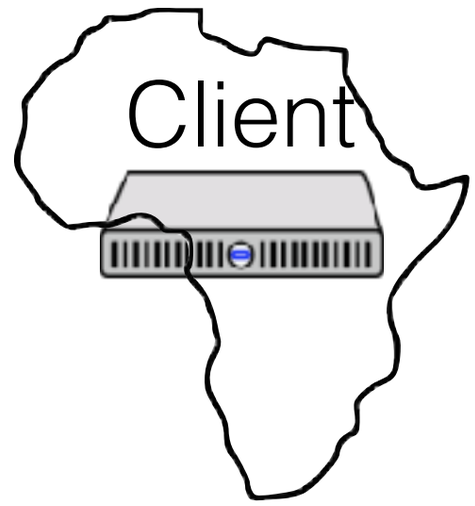

```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

done2=false


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```

done1=false



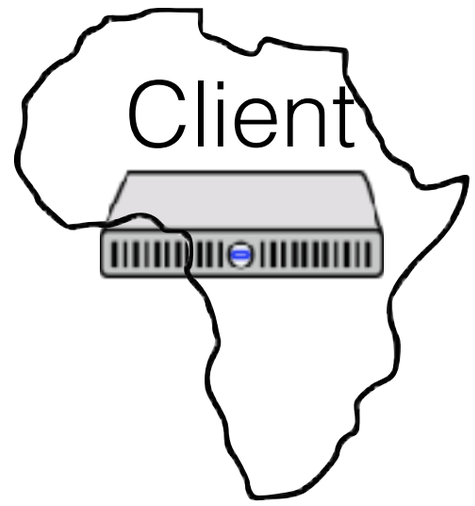

```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

done2=false


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```

done1=false



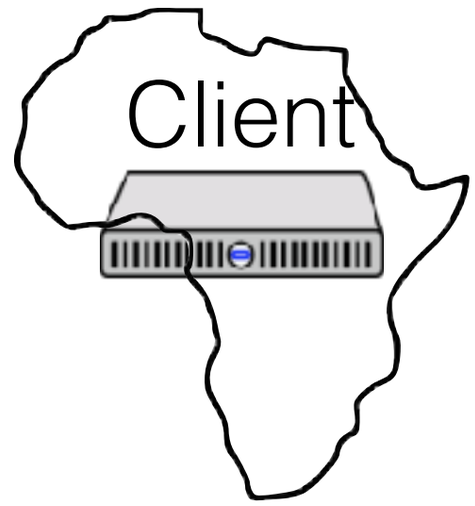

```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

done2=false


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```

~~done1=false~~



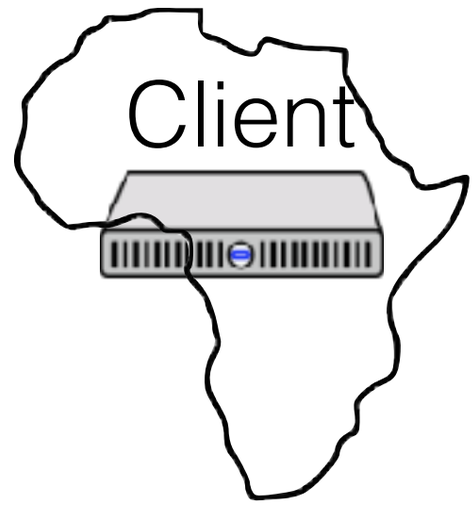

```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

done2=false


```
put (k1, f(data))
put (done1, true)
```


```
while(get(done1) == false)
    ;
put (k2, g(get(k1));
put (done2, true)
```

done1=true




```
while(get(done2) == false)
    ;
rslt = h(get(k1), get(k2))
```

done2=false

Questions

While a write is waiting on invalidations, can clients read old values from caches?

Questions

While a write is waiting on invalidations, can the writing client perform a different write?

Questions

While a write is waiting on invalidations, can the server process a read to a different location?

Questions

While a write is waiting on invalidations, can the server process a read to the same location?

Questions

While a write is waiting on invalidations, can the server process a write to a different location?

Questions

While a write is waiting on invalidations, can the server process a write to the same location?

More Questions

Why does the server wait until write is applied before returning to the client?

Why queue incoming requests during a write?

How much directory state is needed at server?