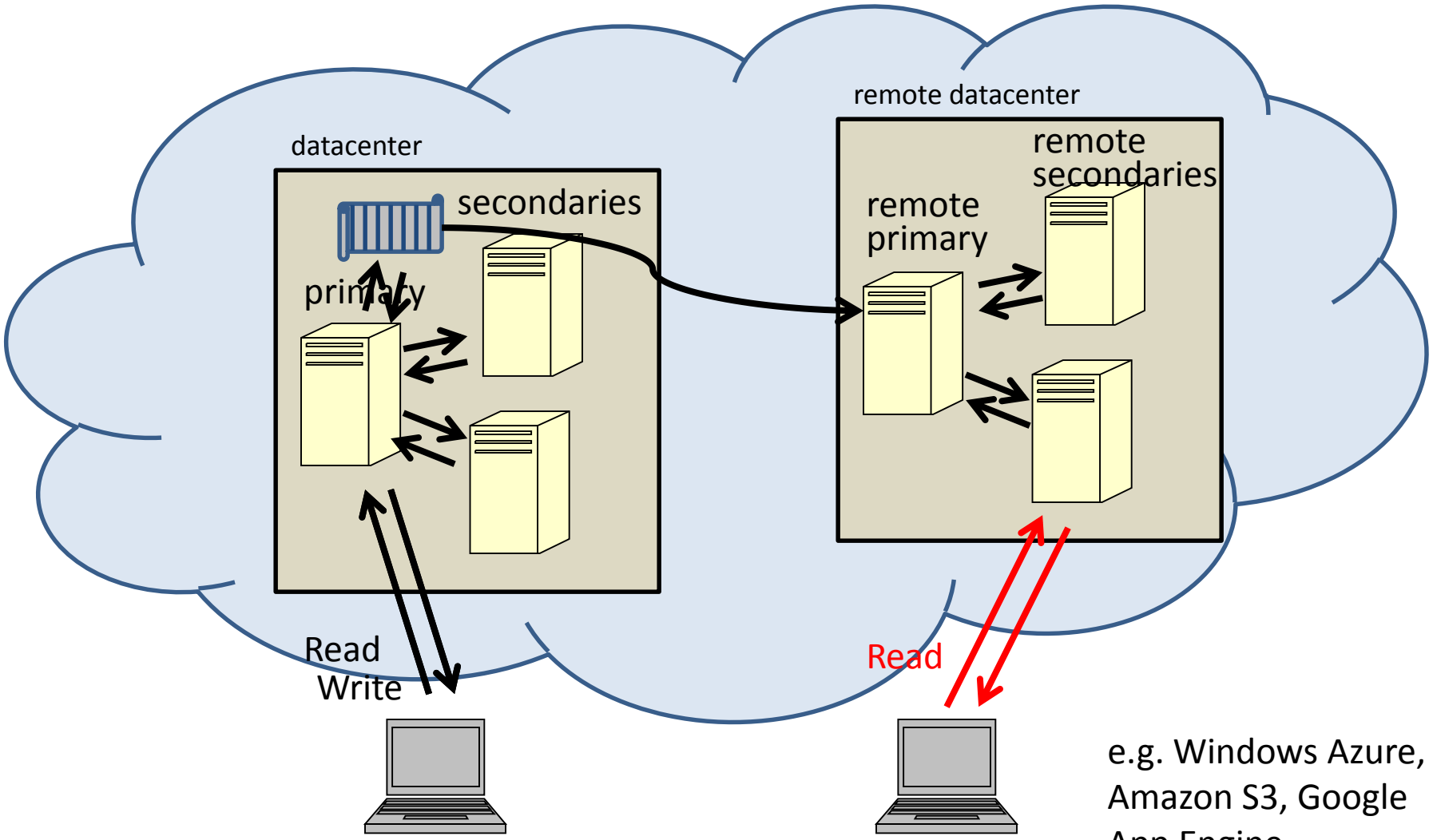
Replicated Data Consistency in the Cloud Explained through Baseball

CACM Vol 56 No 12 (Dec 2013)

By Doug Terry (MS Research)

Data Replication in the Cloud



e.g. Windows Azure, Amazon S3, Google App Engine...

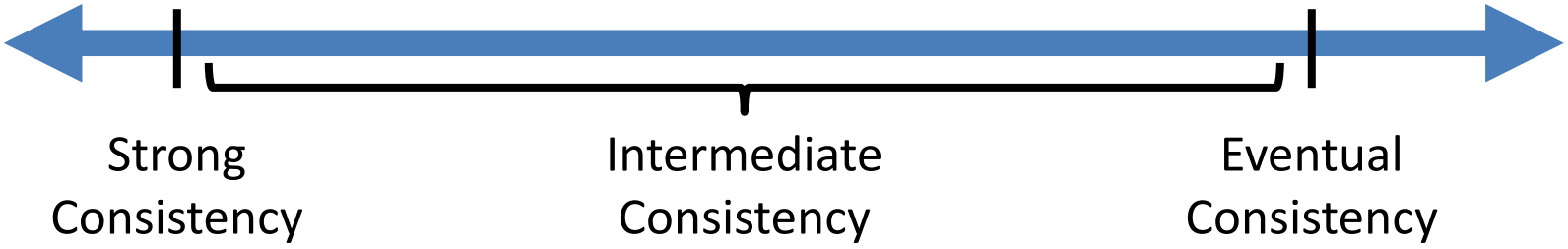
Questions for Replicated Cloud Storage

1. What consistency choices *do* storage systems offer?
2. What *might* they offer?
3. Why are these choices *useful*?

Some Popular Systems

- Amazon S3 – **eventual** consistency
- Amazon Simple DB – **eventual** or strong
- Google App Engine – **strong** or eventual
- Yahoo! PNUTS – **eventual** or strong
- Windows Azure Storage – **strong** or eventual
- Cassandra – **eventual** or strong (if R+W > N)
- …

A Spectrum of Consistency



Lots of consistencies proposed in research community: probabilistic quorums, session guarantees, epsilon serializability, fork consistency, causal consistency, demarcation, continuous consistency, ...

My Favorite Read Consistency Guarantees

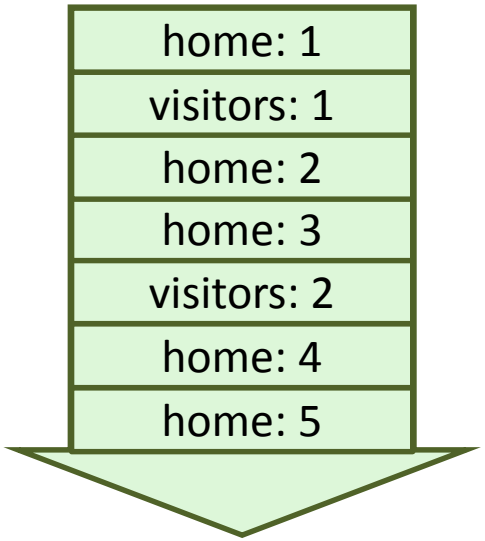
Strong Consistency	See all previous writes.
Eventual Consistency	See subset of previous writes.
Consistent Prefix	See initial sequence of writes.
Monotonic Freshness	See increasing subset of writes.
Read My Writes	See all writes performed by reader.
Bounded Staleness	See all “old” writes.

The Game of Baseball

```
for inning = 1 .. 9
      outs = 0;
      while outs < 3
            visiting player bats;
            for each run scored
                  score = Read ("visitors");
                  Write ("visitors", score + 1);
      outs = 0;
      while outs < 3
            home player bats;
            for each run scored
                  score = Read ("home");
                  Write ("home", score + 1);
  end game;
```

Strong Consistency

aka linearizability, one-copy serializability



Visitors Home

2 5

Reader #1

Visitors Home

2 5

Reader #2

Visitors Home

2 5

Under the covers:

- read-any/write-all or quorums
- 2-phase commit
- primary copy

Guarantee: See all previous writes.

Report card:

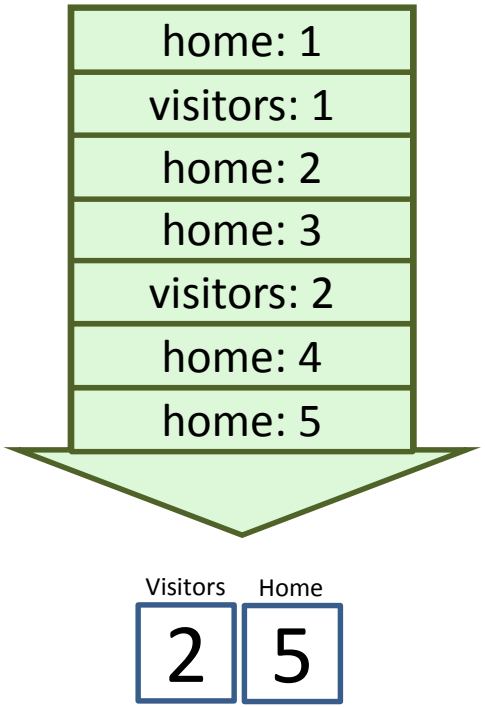
Consistency A

Performance D

Availability F

Eventual Consistency

aka weak/relaxed/optimistic/lazy consistency



Reader #1

Visitors	Home
1	3

Reader #2

Visitors	Home
2	2

Under the covers:

- update anywhere or primary copy
- arbitrary write propagation
- read tentative data from any replica

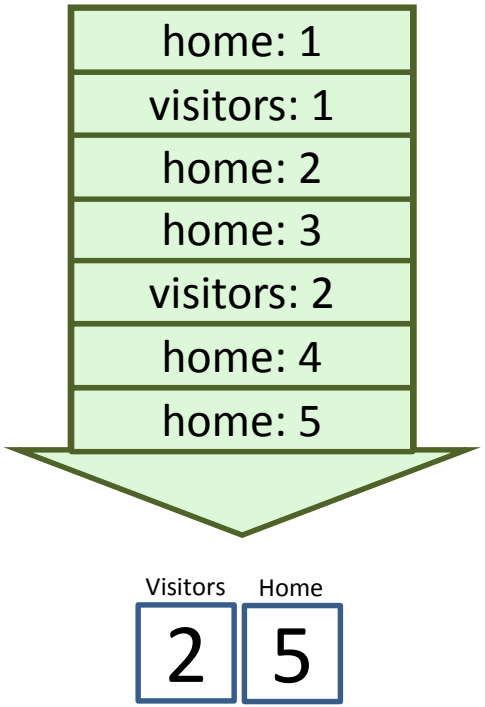
Guarantee: See subset of previous writes; eventually see all writes.

Report card:

Consistency	D
Performance	A
Availability	A

Consistent Prefix

aka ordered delivery, snapshot isolation



Reader #1

Visitors	Home
1	2

Reader #2

Visitors	Home
2	3

Under the covers:

- update primary or anywhere
- ordered writes
- read committed data from any replica

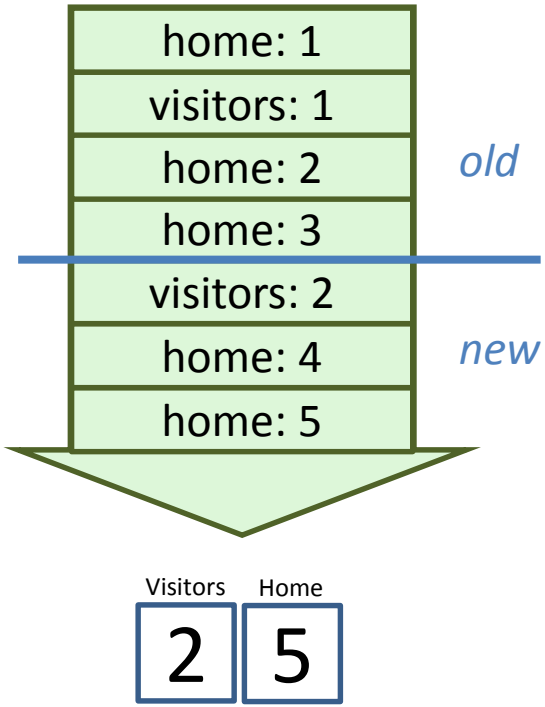
Guarantee: See initial sequence of writes.

Report card:

Consistency	C
Performance	B
Availability	A

Bounded Staleness

aka periodic snapshots, continuous consistency



Reader #1

Visitors	Home
1	3

Reader #2

Visitors	Home
1	4

Under the covers:

- primary replica
- bounded delivery to secondary replicas
- or check on read

Guarantee: See all "old" writes.

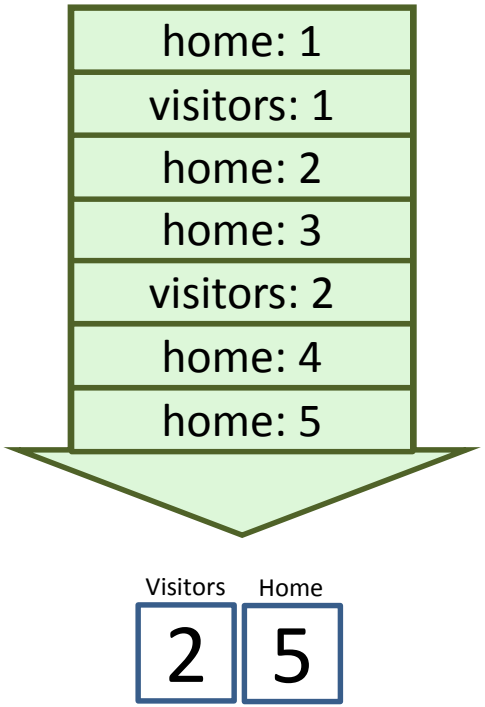
variants: bounds on data values, log, etc.

Report card:

Consistency	B
Performance	C
Availability	D

Monotonic Freshness

aka session guarantees



Reader #1
at time t1

Visitors	Home
1	2

Reader #1
at time t2

Visitors	Home
2	3

Under the covers:

- update anywhere or primary copy
- client records read-set
- restrict reads to sufficiently up-to-date replicas

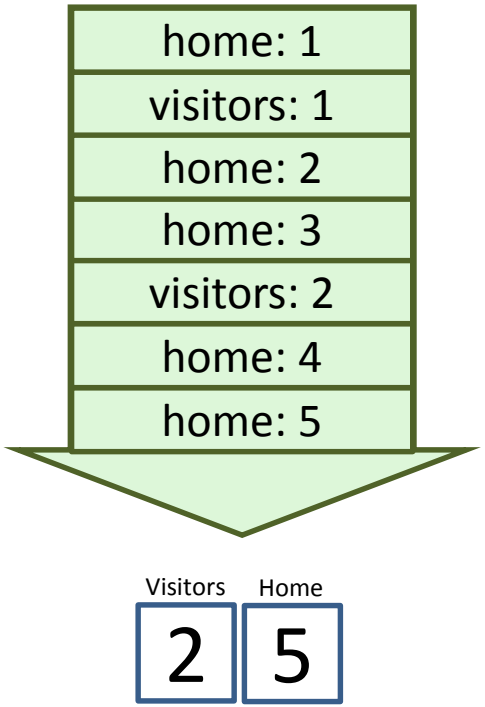
Guarantee: See increasing subset of previous writes.

Report card:

Consistency	C
Performance	B
Availability	B

Read My Writes

aka session guarantees



Scorekeeper

Visitors	Home
2	5

Other Reader

Visitors	Home
1	3

Under the covers:

- update anywhere or primary copy
- client records write-set
- restrict reads to sufficiently up-to-date replicas

Guarantee: See all writes performed by reader.

Report card:

Consistency	C
Performance	C
Availability	C

Consistency Trade-offs

		consistency	*performance*	*availability*
Strong Consistency	See all previous writes.	A	D	F
Eventual Consistency	See subset of previous writes.	D	A	A
Consistent Prefix	See initial sequence of writes.	C	B	A
Bounded Staleness	See all “old” writes.	B	C	D
Monotonic Reads	See increasing subset of writes.	C	B	B
Read My Writes	See all writes performed by reader.	C	C	C

The Game of Baseball

```
for inning = 1 .. 9
      outs = 0;
      while outs < 3
            visiting player bats;
            for each run scored
                  score = Read ("visitors");
                  Write ("visitors", score + 1);
      outs = 0;
      while outs < 3
            home player bats;
            for each run scored
                  score = Read ("home");
                  Write ("home", score + 1);
  end game;
```

Official Scorekeeper

```
score = Read ("visitors");
Write ("visitors", score + 1);
```

- Desired consistency?

  Strong

  = Read My Writes!

Choices:

- Strong
- Eventual
- Prefix
- Bounded
- Monotonic
- Read My Writes

Umpire

```
if middle of 9th inning then
    vScore = Read ("visitors");
    hScore = Read ("home");
    if vScore < hScore
        end game;
```

- Desired consistency?

  Strong consistency

Choices:

- Strong
- Eventual
- Prefix
- Bounded
- Monotonic
- Read My Writes

Radio Reporter

```
do {
    vScore = Read ("visitors");
    hScore = Read ("home");
    report vScore and hScore;
    sleep (30 minutes);
}
```

- Desired consistency?
  Consistent Prefix
  Monotonic Freshness
  or Bounded Staleness

Choices:
- Strong
- Eventual
- Prefix
- Bounded
- Monotonic
- Read My Writes

Radio Reporter

```
do {
    BeginTx();
        vScore = Read ("visitors");
        hScore = Read ("home");
    EndTx();
    report vScore and hScore;
    sleep (30 minutes);
}
```

Use transaction to read from a consistent snapshot.

- Desired consistency?

  Consistent Prefix

  Monotonic Freshness or Bounded Staleness

Choices:

- Strong
- Eventual
- Prefix
- Bounded
- Monotonic
- Read My Writes

Sportswriter

```
While not end of game {
    drink beer;
    smoke cigar;
}
go out to dinner;
vScore = Read ("visitors");
hScore = Read ("home");
write article;
```

- Desired consistency?

  Eventual?

  Strong = Bounded Staleness

Choices:

- Strong
- Eventual
- Prefix
- Bounded
- Monotonic
- Read My Writes

Statistician

```
Wait for end of game;
score = Read ("home");
stat = Read ("season-runs");
Write ("season-runs", stat + score);
```

- Desired consistency?

  Strong Consistency (1st read)

  Read My Writes (2nd read)

Choices:

- Strong
- Eventual
- Prefix
- Bounded
- Monotonic
- Read My Writes

Stat Watcher

```
do {
    stat = Read ("season-runs");
    discuss stats with friends;
    sleep (1 day);
}
```

- Desired consistency?

  Eventual Consistency

Choices:

- Strong
- Eventual
- Prefix
- Bounded
- Monotonic
- Read My Writes

Summary of Baseball Participants

Official scorekeeper:

score = **Read** (“visitors”);

Write (“visitors”, …

Read My Writes

Umpire:

if middle of 9th inning then

vScore = **Read** (“visitors”);

hScore = **Read** (“home”);

if vScore < hScore

end game;

Strong Consistency

Radio reporter:

do {

vScore = **Read** (“visitors”);

hScore = **Read** (“home”);

report vScore and hScore;

sleep (30 minutes);

}

Consistent Prefix + Monotonic Freshness

Sportswriter:

While not end of game {

drink beer;

smoke cigar;

}

go out to dinner;

vScore = **Read** (“visitors”);

hScore = **Read** (“home”);

write article;

Bounded Staleness

Statistician:

Wait for end of game;

score = **Read** (“home”);

stat = **Read** (“season-runs”);

Write (“season-runs”, stat);

Strong Consistency

Read My Writes

Stat watcher:

stat = **Read** (“season-runs”);

discuss stats with fans;

Eventual Consistency

Observations

- Different clients want different guarantees, even when accessing the same data
- All six consistency guarantees are useful
- Clients (e.g. scorekeeper) may obtain strong consistency with a weaker guarantee
- Clients (e.g. radio reporter) may want multiple guarantees for same read
- Clients (e.g. statistician) may want different guarantees for different reads
- Strong consistency would be okay but result in worse performance (and availability)

Read Performance

Consistency	Average Read Time (ms)
strong	179
causal	48
bounded (30)	85
read-my-writes	28
monotonic	25
eventual	25

System: Pileus-on-Azure
Client: laptop in Seattle running YCSB
Replicas: West Europe (primary), North Europe, West US, East US

Conclusions

- Replication schemes involve tradeoffs between consistency, performance, and availability
- Applications may benefit from choices between strong and eventual consistency
- Choosing the best consistency requires understanding application semantics, usage scenarios, system properties, etc.