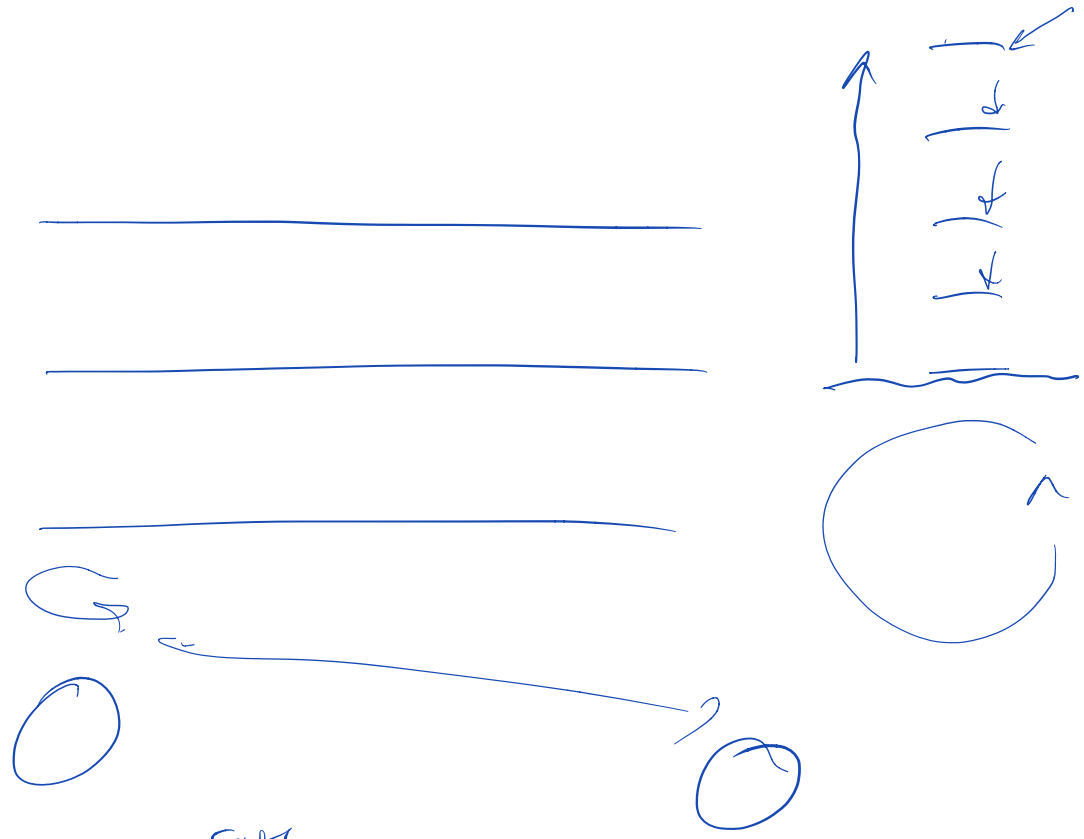


PLANE	FLIGHT#	FUEL
AIRBUS IBERIA	235	— l

RING

PLANE	FLIGHT#	RING

# of planes in RING r ≤ 3

LANDING STRIP

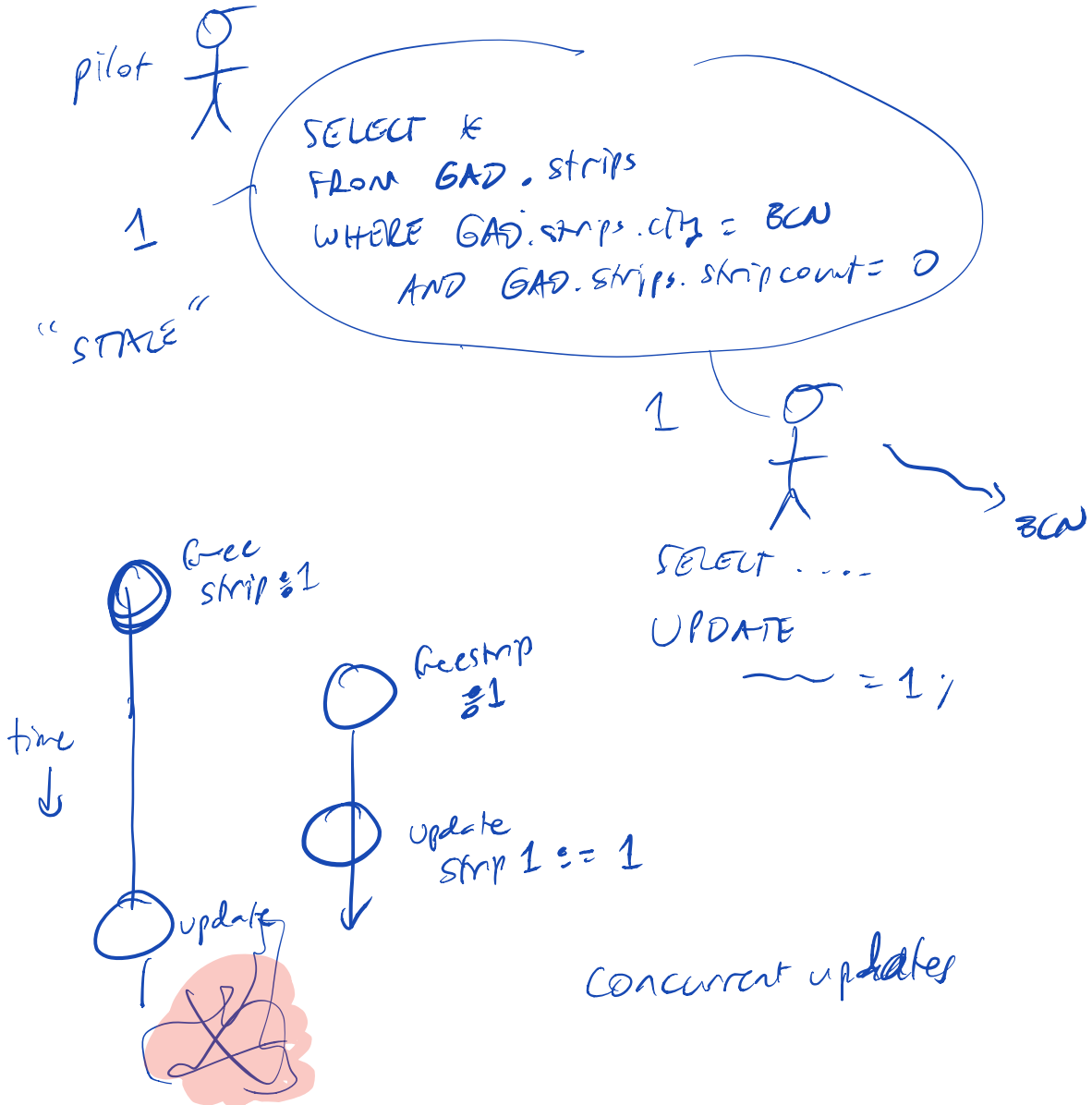
"	"	STRIP

# of planes STRIP n ≤ 1

Global airport database

- concurrent updates
- "consistency guarantees"

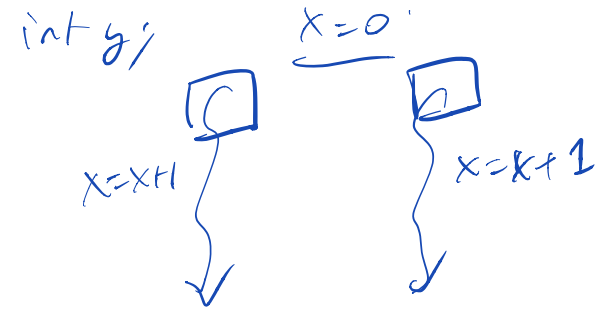
inconsistent ≡
broken invariants



concurrent updates

```
int x;
int y;
```

C



x ?

x=2

```
LOAD x, R0
  {R0 = x}
INC R0
  {R0 = 1}
STORE R0, x
  {x = 1}
```

```
LOAD x, R0
  {R0 = x}
INC R0
  {R0 = 1}
STORE R0, x
  {x = 1}
```

not Serializable

[x=x+1] [x=x+1]

Serializable

[x=x+1] [x=x+1]

SOLUTION (MIT world)

→ LOCKS

mutual exclusion

Synchronized

```
pthread_mutex_lock(&l);
x = x + 1;
pthread_mutex_unlock(&l);
```

ACID

"transactions"

Atomicity — "atom" atomos — all or nothing

Consistency — maintains invariants

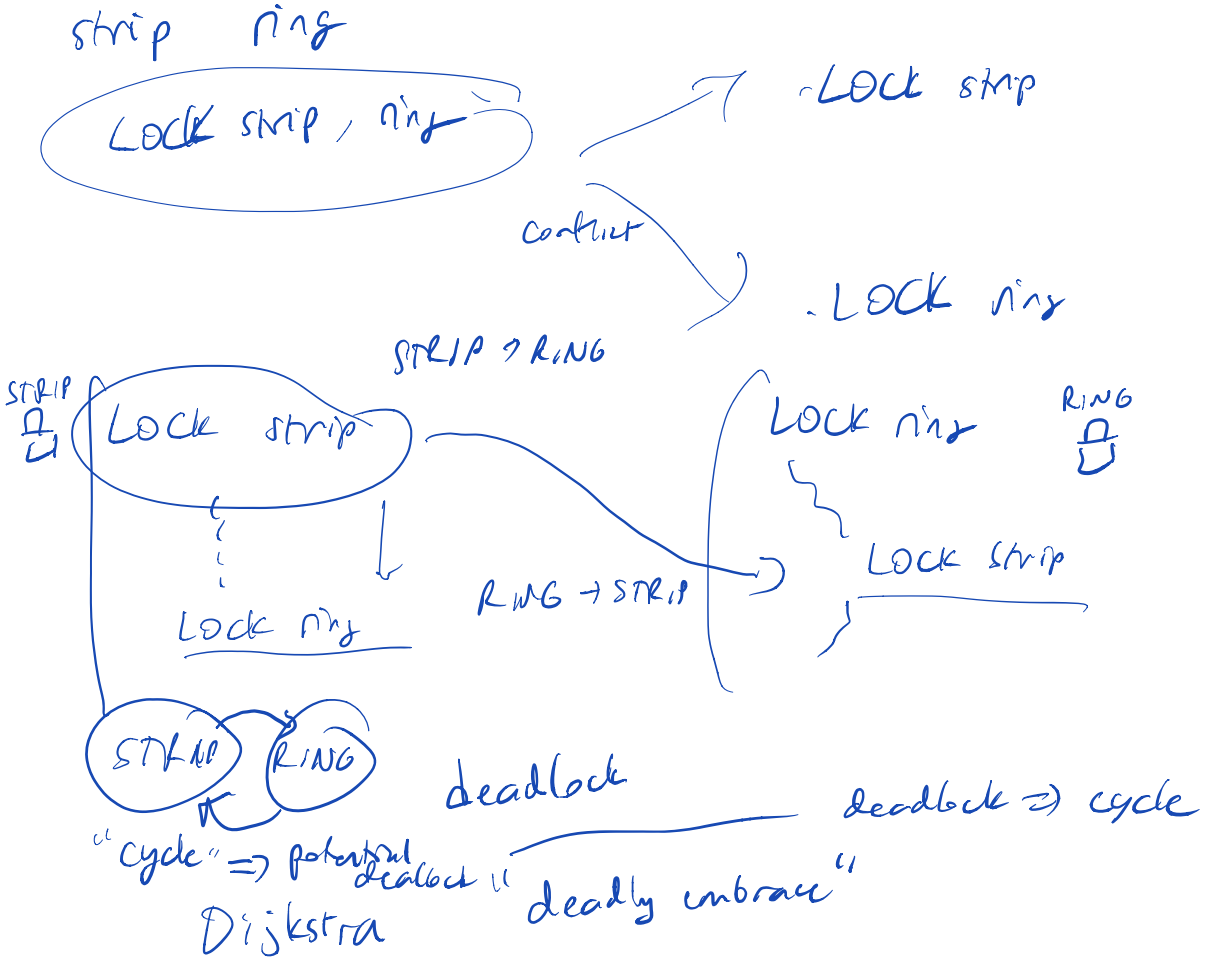
Isolation — no observable intermediate updates

Durability — persistence

LOCK STRIP ← "exclusive lock"

SELECT ... UPDATE ...

UNLOCK STRIP



deadlock avoidance

request all locks at start

deadlock detection

try all locks
If one is locked
retry all
⇝ spinning

"canonical" order
R, S S, R ⇒ "R, S" acquire

correctness ++! 10/10

performance ??. ---

"S,R" release (reverse)

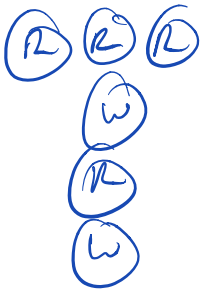
"database-level locks"

"READER" lock
shared lock
} all readers can operate concurrently

"WRITE" lock
exclusive lock
} only one writer (& no readers) at a time.

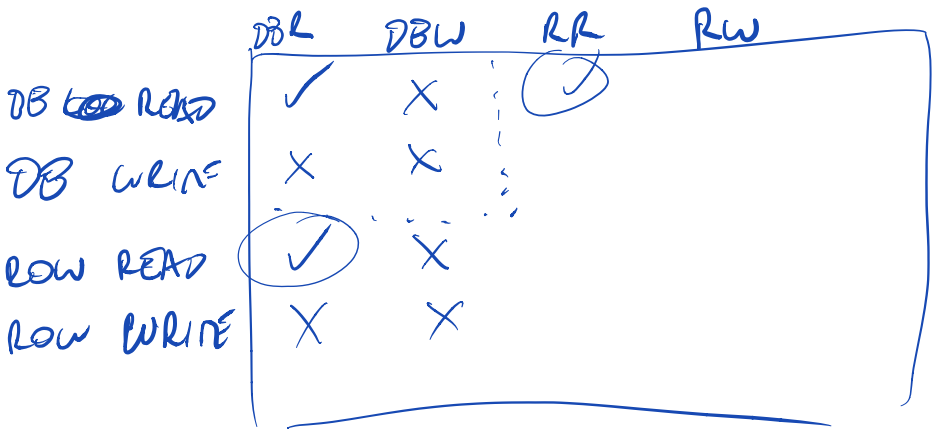
before

after

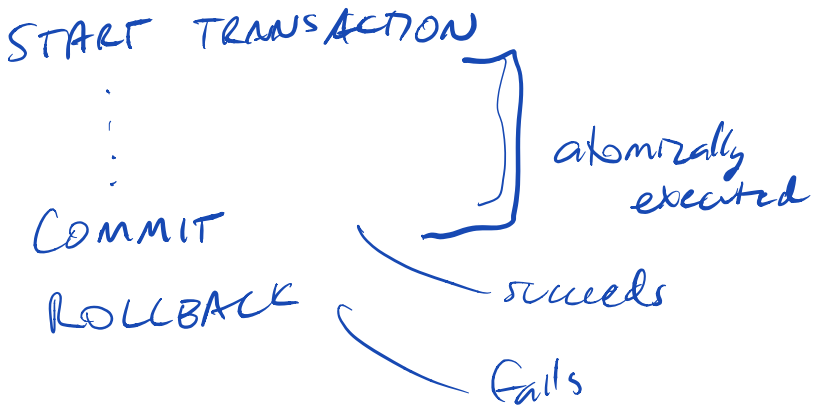


Lock conflict table

in mode now \ operation	READ	WRITE
READ SHARED	✓	X
WRITE EXCLUSIVE	X	X



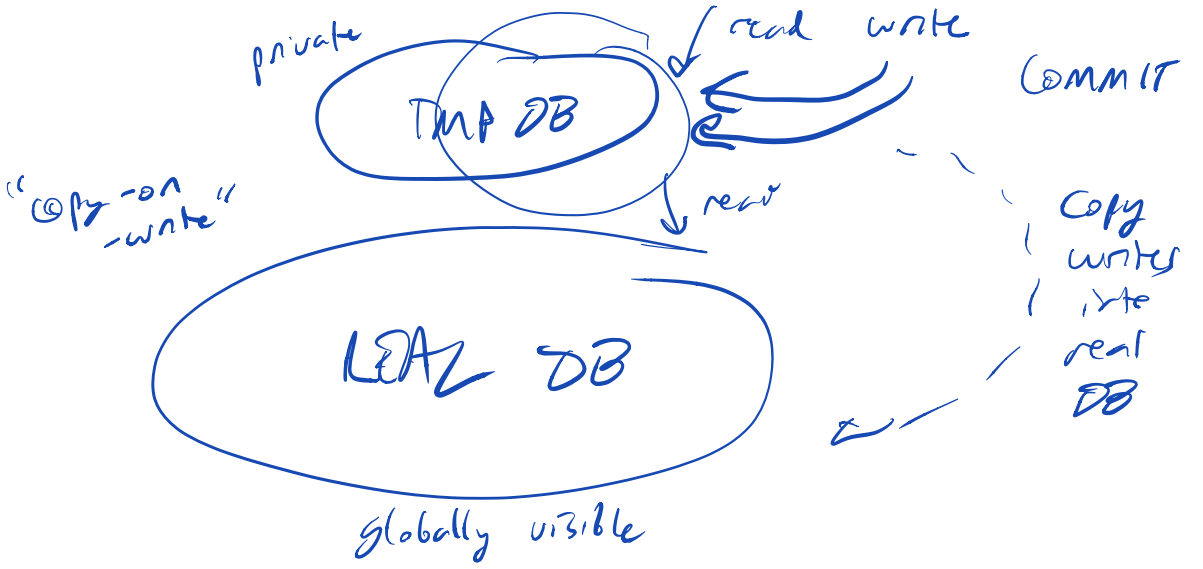
"ring + skip ≤ 4"



Kink += 1
Khlock += 1
Kylock += 1
Kink += 1
if (... Kardashians > 1)
ROLLBACK.

log
Kink 0
Khlock 0
Kylock 0
Kink 1

"undo"



LOCKS : XACTIONS

- Serializability ACID
- reasonable concurrency

(of) of serialized xactions (esp. if long)

availability

"nines" of avail

99%

3.65 days unavailable

99.9% ~ 8 hours/year

99.999% ~ 5 min.

not just locking failures

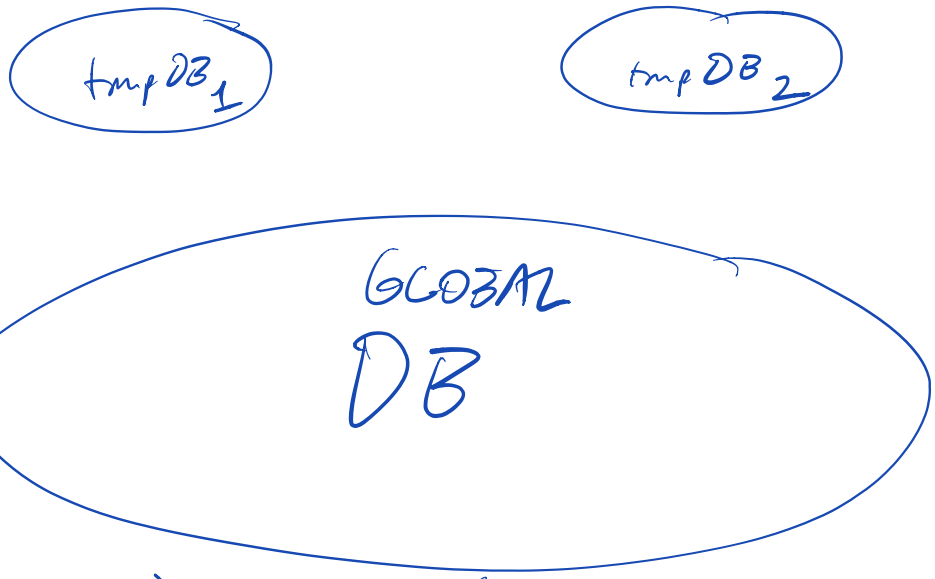
OLTP

mostly write

online transaction processing

BANK

"inconsistency" (Amazon)



(WRITES1 ∪ READS1) == (values of WRITES1, values of READS1) ✓

Check read set & write set
make sure not modified
if not → commit
if modified → ROLLBACK

Optimistic Concurrency control

Kung & Robinson

locking – pessimistic
– progress

"livelock"

"unfair"

guarantees
serializability

aborts

Multi-version Concurrency Control

"snapshot isolation"

Kink 3 (Many)

Kink 1 (11:05 am June 1 2017)

~~Kink 0 (12:30 pm July 4 2017)~~

12:27 pm July 4

timestamp

private DB

Kink 2 (12:27 pm July 4)

write skew anomaly

$V_1 = 100$

$V_2 = 100$

t_1 t_2

read V_1 V_2

$V_1 -= 200$

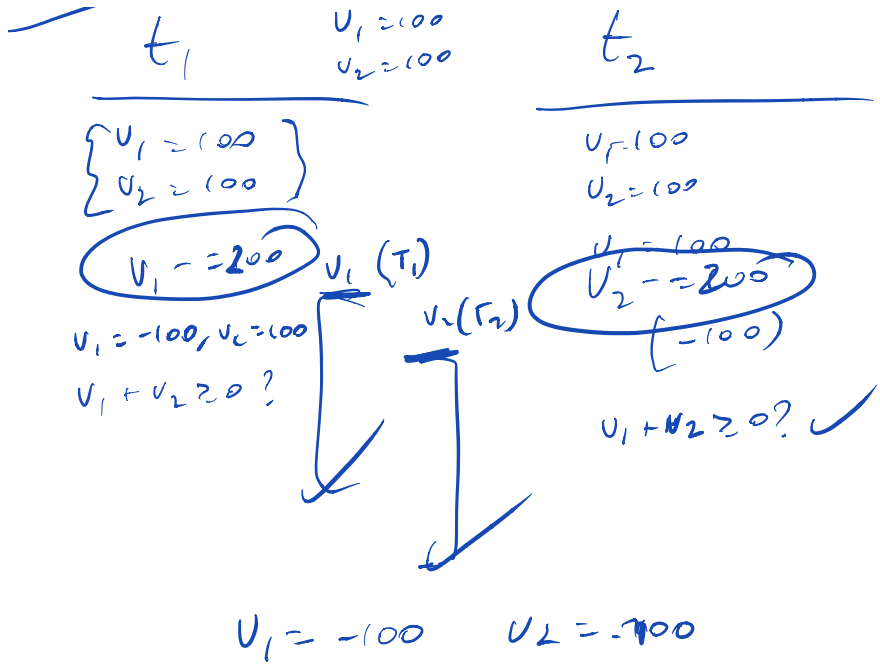
$V_1 = -100$ $V_2 = 100$

read V_1, V_2

$V_2 -= 200$

$V_1 + V_2 \geq 0$

SNAPSHOT



MVCC/snapshot isolation ⇒ more commits
some weirdness
(write skew)
"anomaly"

Perf. exclusive lock table → reader shared locks → row lock → OCC ⇢ MVCC

"serializability"

Fault tolerance
⇒ replication

SPoF
Single point of failure

MTTF = Mean time to failure

availability

$P[\text{fail}] = p$

$p = 1/1000$

$E(\#\text{ of coin flips}) = 1/p = 1000$

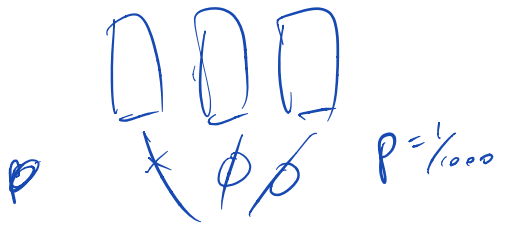
$\frac{1}{p = 1/1000}$ $\frac{2}{p^2 \; 1/1{,}000{,}000}$ $\frac{3}{1^3 \; \frac{1}{10^9}}$ ...

trimodular redundancy

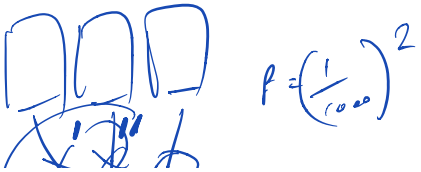
fail stop

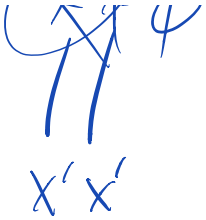
Lamport

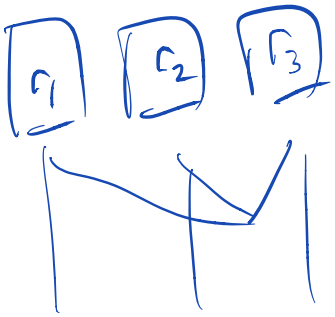
Byz Generals Problem



BFT







Tandem

Jim Gray