Consensus Number

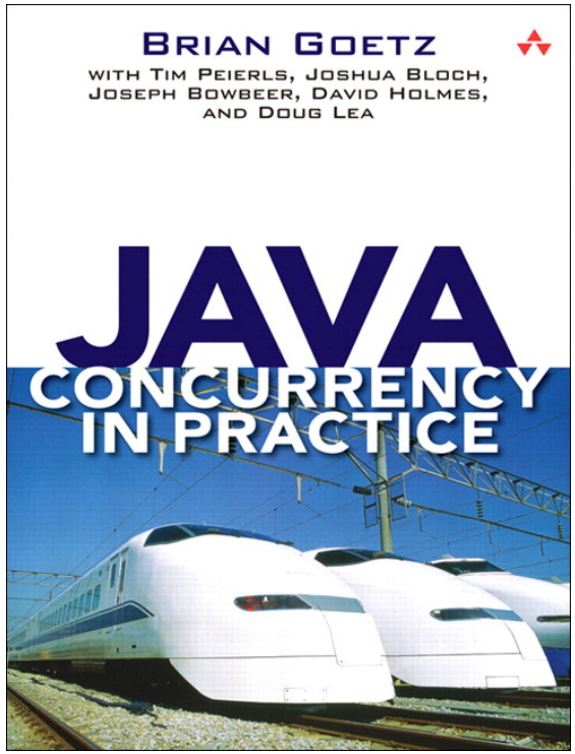
魏恒峰

hfwei@nju.edu.cn

2017 年 12 月 14 日

Are you Familar with Concurrent Programming?

The Key: Synchronization!



Using the Synchronization Primitives Provided by Your Favorite Languages.

`synchronized`
`Semaphore`
`BlockingQueue`
`ConcurrentMap`
`Phaser`
`Barrier`

synchronized	BlockingQueue	Phaser
Semaphore	ConcurrentMap	Barrier

```
class AtomicInteger:
  get()
  set(int newValue)

  getAndIncrement()
  getAndDecrement()
  getAndSet(int newValue)

  compareAndSet(int expectedValue, int newValue)
```

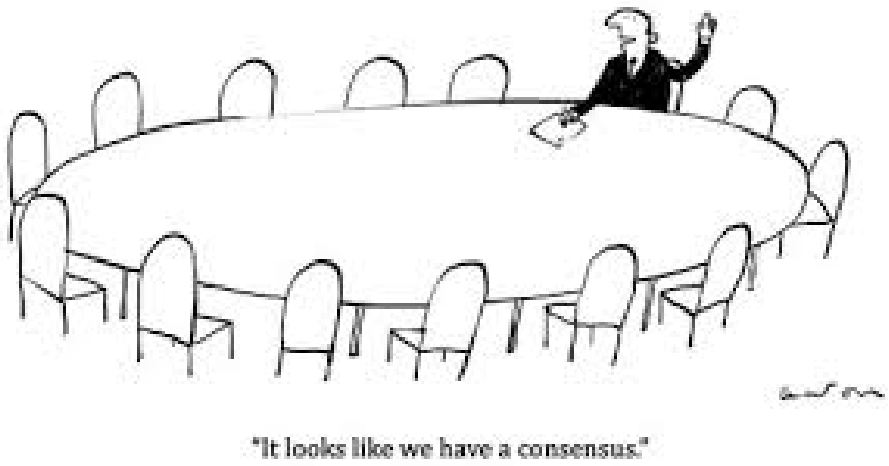
```
compareAndSet(int expectedValue, int newValue)
compareAndSwap(int expectedValue, int newValue)
```

CAS — CMPXCHG

impl.

usage

Consensus



"It looks like we have a consensus."

Propose

Decide

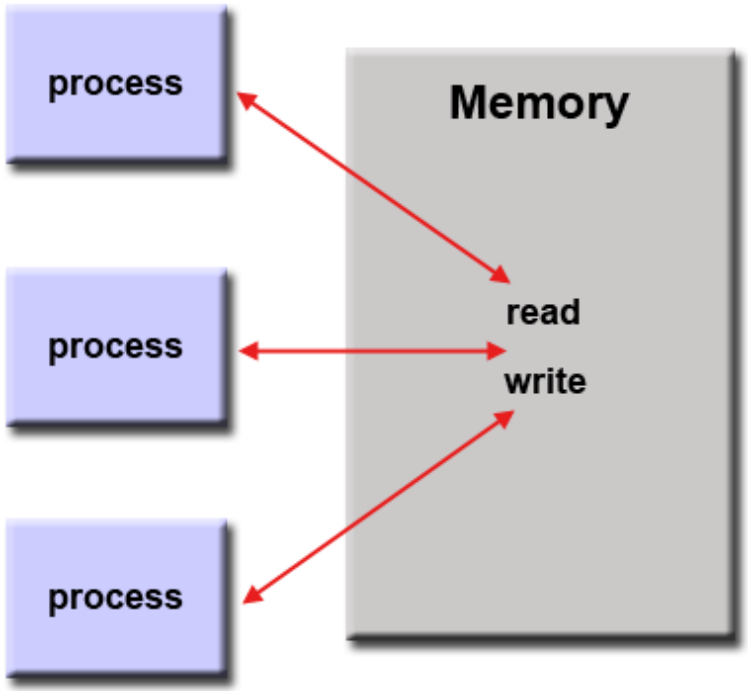
Propose

Decide

Definition (The Consensus Problem)

- Agreement: All (non-faulty) processes must agreen on the same value.
- Validity: The common decision value must be the value proposed by some process.
- Termination: Each (non-faulty) process must eventually decide on a value.

More clarification on “termination”



(redraw)

consensus object (fig here)

```
1    public interface Consensus<T> {
2      T decide(T value);
3    }
```

consensus protocol

```
1  public abstract class ConsensusProtocol<T>
2                        implements Consensus<T> {
3    protected T[] proposed = (T[]) new Object[N];
4
5    void propose(T value) {
6      proposed[ThreadID.get()] = value;
7    }
8
9    public abstract T decide(T value);
10 }
```

implement X using Y


Thank
You!