

Abstract Data Types for Concurrency

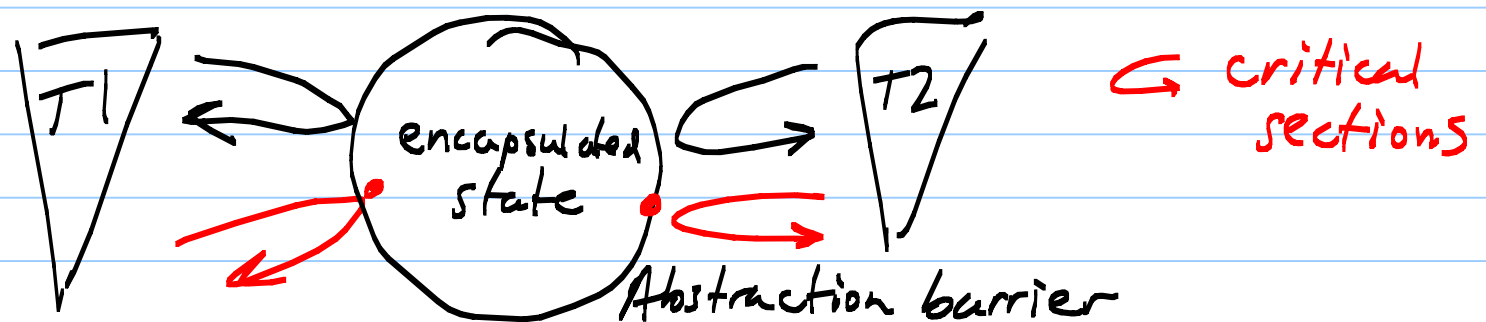
Note Title

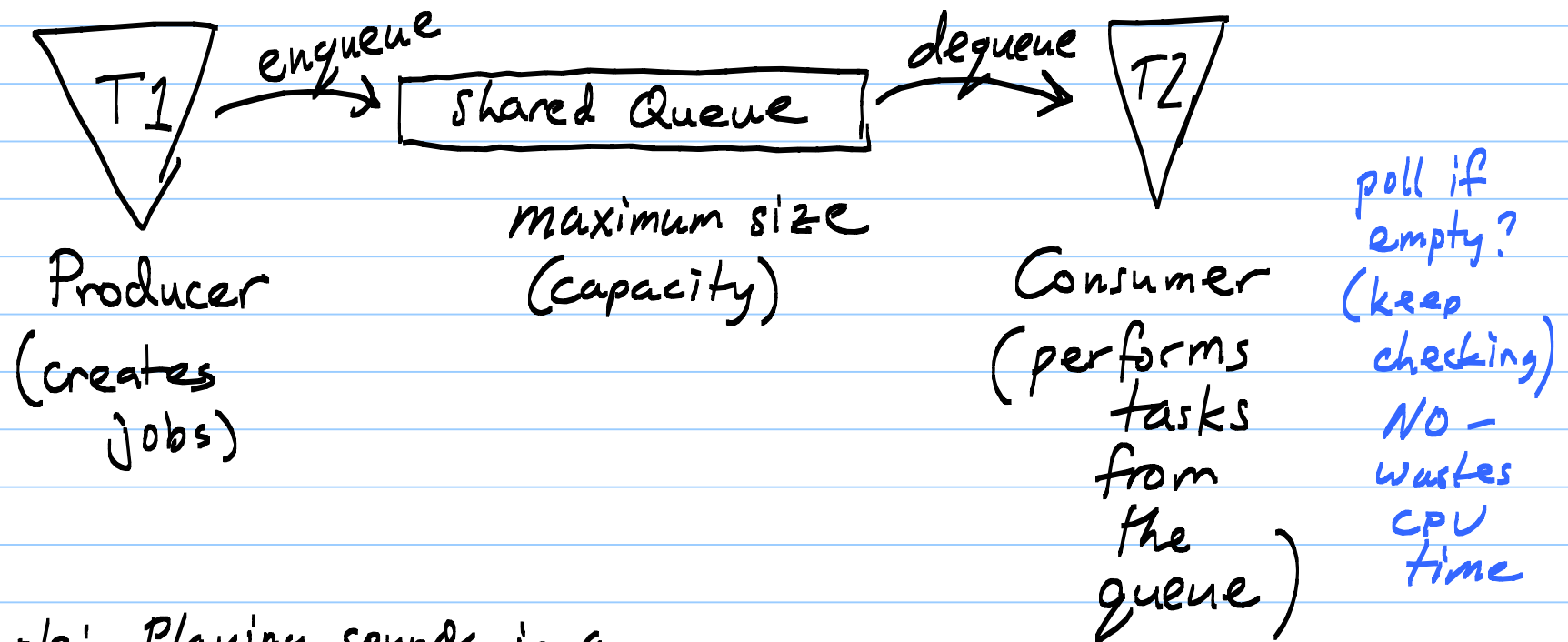
- Problems with explicit locks
- Wanted: Thread-safe abstract data types
- Example: Blocking Queue
- Synchronized methods
- wait and notify
- implementation & application of Blocking Queue

Problems with explicit locks

- forget to lock at some point in the code
 - forget to unlock
 - an exception is thrown — exit method w/o unlocking
- ```
try {
 ;
} finally { unlock... }
```

Wanted: Thread-safe abstract data types





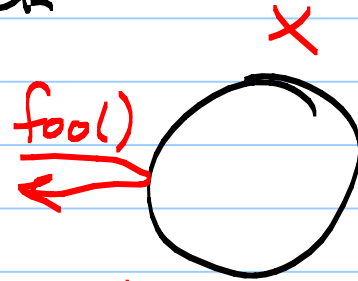
Example: Playing sounds in a separate thread

T1 - enqueues files

T2 - plays out the files to speaker

Every object has an associated lock

synchronized methods:



① acquire the object's lock on entry

`x.foo( — , — );`

② release lock on exit (automatically)

return  
or throw

if a thread calls a method that's not synchronized,  
it doesn't acquire the lock

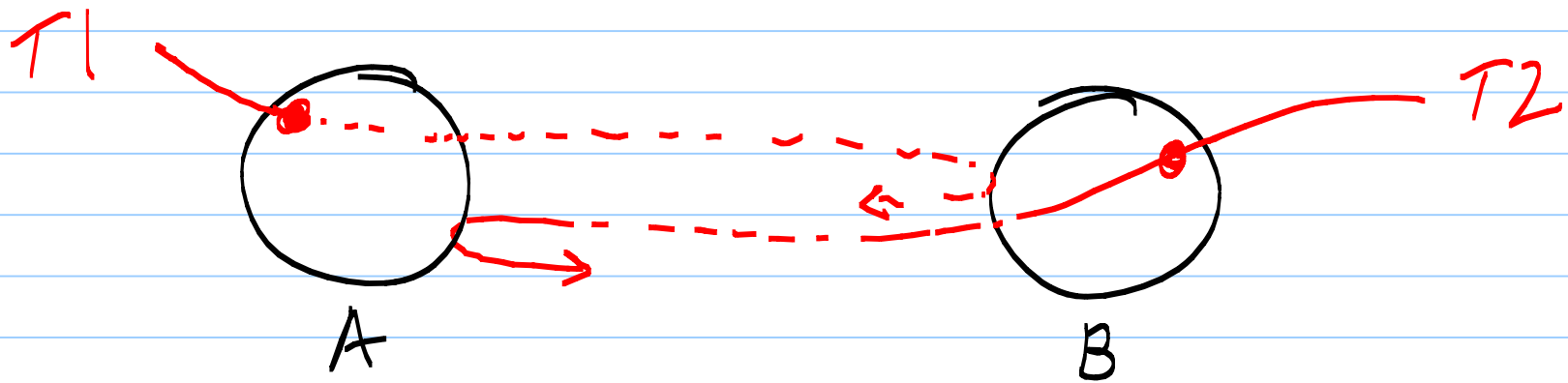
wait() —

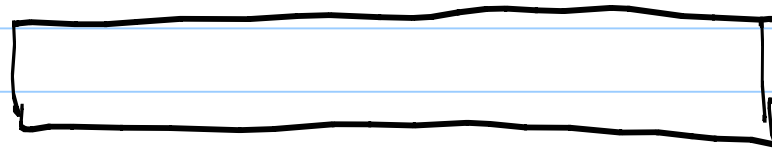
- blocks the thread waiting for some other thread to notify it (InterruptedException)
- release the lock temporarily until notified + get back in

notify() —

- lets waiting thread know that condition may have changed — wakes it up

Can deadlock w/ synchronized methods





buffering masks latency (delays)  
helps with jitter due to variations in latency

