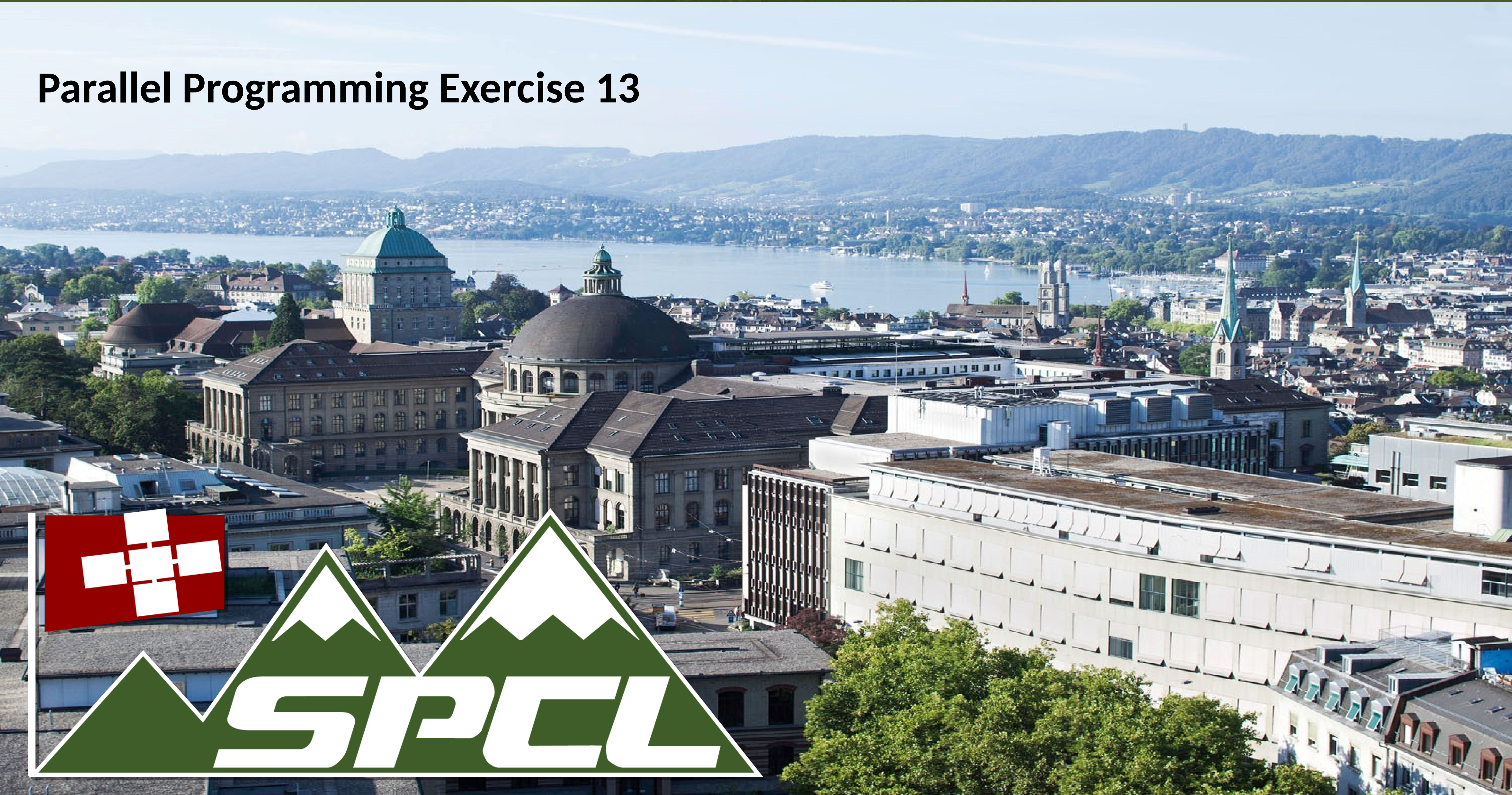


Parallel Programming Exercise 13

Exercise 12: Consensus Protocol

- There are N threads, each proposes a value
- All threads must agree on one value
- Some threads are allowed to “sleep”, i.e., get de-scheduled
- Protocol must terminate in finite number of steps (wait-free)

Exercise 12: Wait-free implies Lock-free

- Explain why a valid wait-free consensus protocol cannot use locks.
- By definition of wait-free, the consensus protocol must finish execution in a finite number of steps.
- Assume that N threads are executing a consensus protocol that uses a lock for mutual exclusion.
- Assume that thread i acquires the lock and all other threads wait.
- If thread i crashes, without releasing the lock, the consensus protocol will never complete.
- However, also by definition of the consensus protocol, we allow a number of threads to fail.
- Therefore, a consensus protocol cannot use locks, because there is no guarantee that it will be wait-free.

Exercise 12: Valence States

- Think of the program states, as we used them in state diagrams to prove mutual exclusion.
- A state is usually described by a set of values, e.g. PC of each thread, current variable values etc.
- In a program implementing a consensus protocol, each state is also described by the possible consensus values.
- Assume that initially there are two possible values for threads to agree on.
- You can place states into 2 categories: bivalence states or univalence states
- Explain why there is a finite number of bivalent states in any wait-free consensus protocol.
- By definition, the consensus protocol will complete in a finite number of steps.
- Therefore, there are a finite number of states.
- Thus, all kinds of states (bivalence and univalence) will be finite in number.

Exercise 12: Consensus among prisoners

- Imagine there are 100 people in a prison.
- Each day the warden picks a prisoner (with replacement - each prisoner has probability to be picked 1/100).
- The prisoner is led to a room with a light that he can turn on or off. Initially the light is turned off.
- After the prisoner was in the room he can state ”by now every prisoner was in the room at least once”. If this statement is made and it is true, all prisoners are released. If the statement is made and it is false, all prisoners are shot.
- Devise a strategy that the prisoners can follow to make sure they get released some day in the future with absolute certainty (no other communication is allowed).
- Extra assumption 1: Prisoners can communicate before the whole process starts.
- Extra assumption 2: Prisoners have a sense of time (know how many days have passed).

Exercise 12: Consensus among prisoners – Solution 1

- **Before the first day, prisoners assign a number from 1 to 100 to each one.**
- **The following is repeated for every 100 day interval until success.**
- **If on the first day of the interval, the prisoner who has been assigned number 1 enters the room, he switches on the light.**
- **Prisoners assigned numbers 2-99 do the following:**
  - If he enters the room and the light is switched off, he does nothing.
  - If he enters the room, the light is switched on and the interval day does not match his number, he switches the light off.
  - If he enters the room, the light is switched on and the interval day matches his number, he does nothing (leave the light on).
- **If on the last day of the interval, the prisoner who has been assigned number 100 enters the room and the light is on, he makes the statement.**
- **The probability in a 100 day interval that the above will succeed is .**

Exercise 12: Consensus among prisoners – Solution 2

- **Before the first day, prisoners select one of them to be the leader/counter.**
- **Each prisoner, except the leader, does the following:**
  - If he enters the room for the first time and the light is off, he switches the light on.
  - If he enters the room and the light is on, he does nothing. Also, if it is his first time in the room, he does not count it.
- **The leader does the following:**
  - If he enters the room and the light is on, he counts +1 and switches the light off.
  - When he counts up to 99, he makes the statement.
- **A bit harder to estimate the average time needed for success, but probably better than the first solution.**
- **If we assume that on average each prisoner enters the room every 100 days, then, on average, the leader will count +1 every 200 days. Therefore, a naïve estimation would be days.**

Exercise 12: Implementing two thread consensus

- **Assume you have a machine with atomic registers and an atomic test-and-set operation with the following semantics (X is initialized to 1):**

```
int TAS() {
    res = X;
    if (res == 1) {
        X = 0;
    }
    return res;
}
```

- **Implement a two-process consensus protocol using TAS() and atomic registers.**

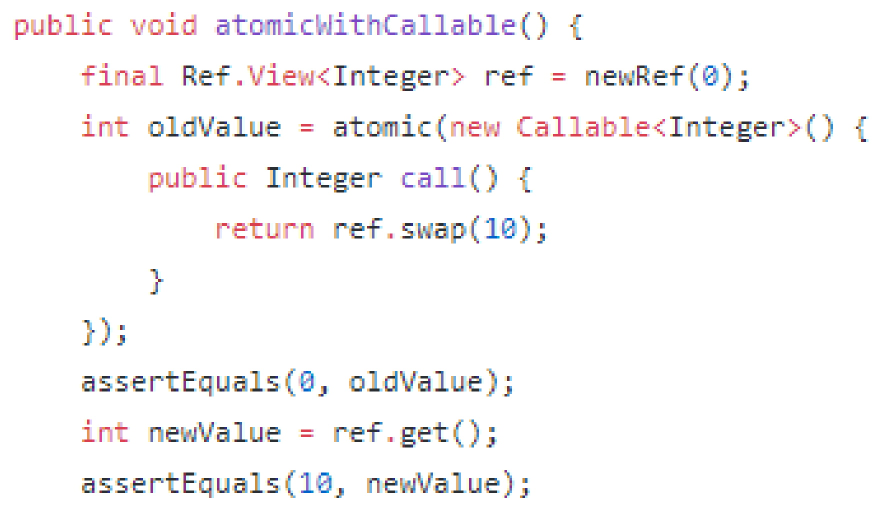
Exercise 12: Implementing two thread consensus - Solution

- **Code for both threads**

```
read own_value;
read other_value;
if (TAS() == 0) {
   return own_value;
} else
   return other_value;
```

Exercise 13: Transactional Memory

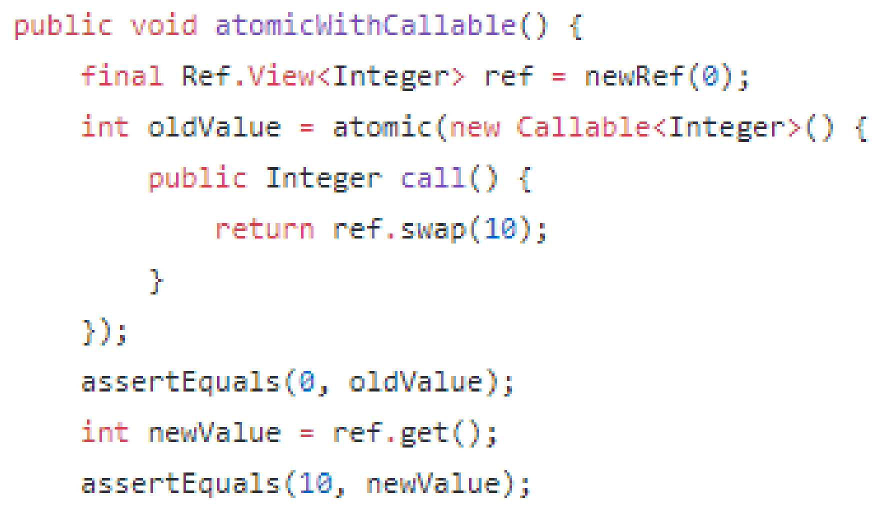
- **No locking required**
- **Need to wrap our code in an atomic block (callable/runnable in ScalaSTM)**



```
public void atomicWithCallable() {
    final Ref.View<Integer> ref = newRef(0);
    int oldValue = atomic(new Callable<Integer>() {
        public Integer call() {
            return ref.swap(10);
        }
    });
    assertEquals(0, oldValue);
    int newValue = ref.get();
    assertEquals(10, newValue);
```

Exercise 13: Transactional Memory

- **STM library needs to know the read and write set -> only access variables the library knows about! (use Ref.View<T>)**



```java
public void atomicWithCallable() {
    final Ref.View<Integer> ref = newRef(0);
    int oldValue = atomic(new Callable<Integer>() {
        public Integer call() {
            return ref.swap(10);
        }
    });
    assertEquals(0, oldValue);
    int newValue = ref.get();
    assertEquals(10, newValue);
```