Lock-free Programming
Practical Continued Solutions

Lock-free Programming Practical Continued

- Why does the constructor create an element which is not used for anything?
  - iHead represents the element before the oldest
  - iTail represents the element after the newest
  - There is a constraint that iHead and iTail must never represent adjacent elements
  - This entails that the list always contains at least one element
  - Hence, we add a "dummy" element to the newly constructed list

Lock-free Programming Practical Continued

- Explain why the data race occurs
  - iHead and iTail can be accessed from different threads
  - iHead and iTail are not atomic variables
  - There is no ordering of accesses to the variables
- Can the race condition be avoided by using atomic variables?
  - No
  - iHead and iTail cannot be atomic, since list<T>::iterator is not trivially copyable