Java Concurrency
Towards a better life …

By
- Srinivasan.raghavan@oracle.com
- Vaibhav.x.choudhary@oracle.com
Java Releases

JDK 1.5:
- Generics
- Concurrency
- Auto-boxing

J2SE 6:
- Collection Framework enhancement
- Drag and Drop
- Improve IO support

J2SE 7:
- Concurrency Utilities
- Swing, - Fork/Join Pool, LWC&HWC
  - G1

J2SE 8:
- Lambda expressions
- Stream Package
- More concurrency Updates
Agenda

- Basics of concurrency
- Java Concurrent Package
- Reentrant Lock / Performance
- Concurrent Collections
- Insight of Fork/Join
- Concurrency in Java 8
- Concurrent Collections java 8 updates
Basics of concurrency

• Amdhal's law
  - Speedup $\leq \frac{1}{F + \frac{(1-F)}{N}}$

• Age of multi-core processors and requirement of parallel programming in mainstream.

• Writing concurrent programming is hard and writing correct concurrent program is even harder.

• Research in the field has been lead by Java designers and the concurrency interest group
General Design Principles

• Its the mutable state stupid
• Design for immutability
• Make field final unless they need to be mutable
• Encapsulate to manage complexity
• Guard all variables in an invariant with same lock
• Document the synchronization policy
Dead Locks avoidance with try locks

- Dead Lock can be avoided using the try locks
- Try locks can used for Probabilistic lock acquire of the lock and exponential back-off and spin lock
- Classic Transfer money Deadlock Problem
• Intrinsic locks can also work as condition queues with wait and notify

• Problem occurs when Intrinsic condition queue may be used with more than one condition predicate

• Reentrant locks provides explicit condition queues which offers more features

• Fairness in access of the lock which reduce contention

• Help us create queues per condition predicate
Where a Lock replaces the use of synchronized methods and statements, a Condition replaces the use of the Object monitor methods.

Consider a case of bounded buffer

- final Condition notFull = lock.newCondition();
- final Condition notEmpty = lock.newCondition();

ArrayBlockingQueue provides the functionality

A typical consumer/producer problem with BlockingQueue.
Performance

Can Reentrant lock replace synchronized?

- No, there are big issues when used carelessly
- Use only when Fair queuing, probabilistic locking and explicit condition queues are required

![Graph showing throughput advantage of ReentrantLock over intrinsic locks for Java 5.0 and Java 6.](image)
Java Synchronizers

- **Executors**
  - Execution tasks in thread pools.
  - Cyclic barrier, Future task, Semaphores, Countdown latch

- **Non-blocking algorithms**
  - AtomicInteger
  - Compare and Swap (CAS)
  - Writing a correct non-blocking algorithm are tough.

- **Find out the code.**
Fork/Join Framework

- New parallel mechanism for compute intensive tasks, the fork-join framework.
- Runs on work-stealing algorithm.
- Approach
  - Partition into subproblems.
  - Create subtasks.
  - Fork subtasks.
  - Join subtasks.
  - Compose solution.
Fork/Join Framework
Java 8 way
Parallelism is the feature built in Java 8.

You can iterate, filter, sort, map, reduce an array in parallel.

The streams API makes heavy use of the Fork-Join framework.

The entire set of features offered by the streams API can be made to work in parallel by just asking for the parallel stream.

The inherit complexity of Fork-Join framework is handled under the hood.
Parallelism Built In

- Parallelism is the feature built in Java 8
- You can iterate, filter, sort, map reduce an array in parallel.
- The streams API makes heavy use of the Fork-Join framework.
- The entire set of features offered by the streams API can be made to work in parallel by just asking for the parallel stream.
- The inherit complexity of Fork-Join framework is handled under the hood.
Java 8 Concurrency Changes

• Addition of fork-join common pool

• ConcurrentHashMap-v8 which supports bulk operations in parallel

• Completable future

• Changes to the Future Task implementation
Concurrent Hashmap

- Started as a replacement to Hashtable.
- Implements Lock stripping
- Doesn’t throw concurrent modification exception
- Built in support for parallelism
- More in the code.
Q & A