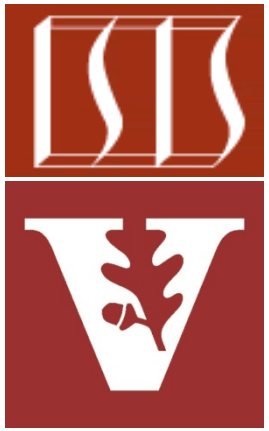
Benefits of Concurrency in Java & Android: Performance



Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Module

- Recognize how concurrency can improve performance in Java & Android

Using Concurrency to Increase Performance

Using Concurrency to Increase Performance

- "Performance" is a characterization of the amount of useful work that can be accomplished

See en.wikipedia.org/wiki/Computer_performance

Using Concurrency to Increase Performance

- "Performance" is a characterization of the amount of useful work that can be accomplished, e.g.
  - Decreasing response time for handling requests

See en.wikipedia.org/wiki/Computer_performance#Response_time

Using Concurrency to Increase Performance

- "Performance" is a characterization of the amount of useful work that can be accomplished, e.g.
  - Decreasing response time for handling requests

- ***Service time*** *– How long it takes to do requested work*
- ***Wait time*** *– How long the request has to wait for before it gets to run*
- ***Transmission time*** *– How long it takes to move request to computer doing the work & the response back to requestor*

Using Concurrency to Increase Performance

- "Performance" is a characterization of the amount of useful work that can be accomplished, e.g.
  - Decreasing response time for handling requests

- ***Service time*** *– How long it takes to do requested work*
- ***Wait time*** *– How long the request has to wait for before it gets to run*
- ***Transmission time*** *– How long it takes to move request to computer doing the work & the response back to requestor*

Using Concurrency to Increase Performance

- "Performance" is a characterization of the amount of useful work that can be accomplished, e.g.
  - Decreasing response time for handling requests

- ***Service time*** *– How long it takes to do requested work*
- ***Wait time*** *– How long the request has to wait for before it gets to run*
- ***Transmission time*** *– How long it takes to move request to computer doing the work & the response back to requestor*

Using Concurrency to Increase Performance

- "Performance" is a characterization of the amount of useful work that can be accomplished, e.g.
  - Decreasing response time for handling requests
  - Increasing the amount of work that can be performed within a given time

See en.wikipedia.org/wiki/Computer_performance#Throughput

Using Concurrency to Increase Performance

- "Performance" is a characterization of the amount of useful work that can be accomplished, e.g.
  - Decreasing response time for handling requests
  - Increasing the amount of work that can be performed within a given time

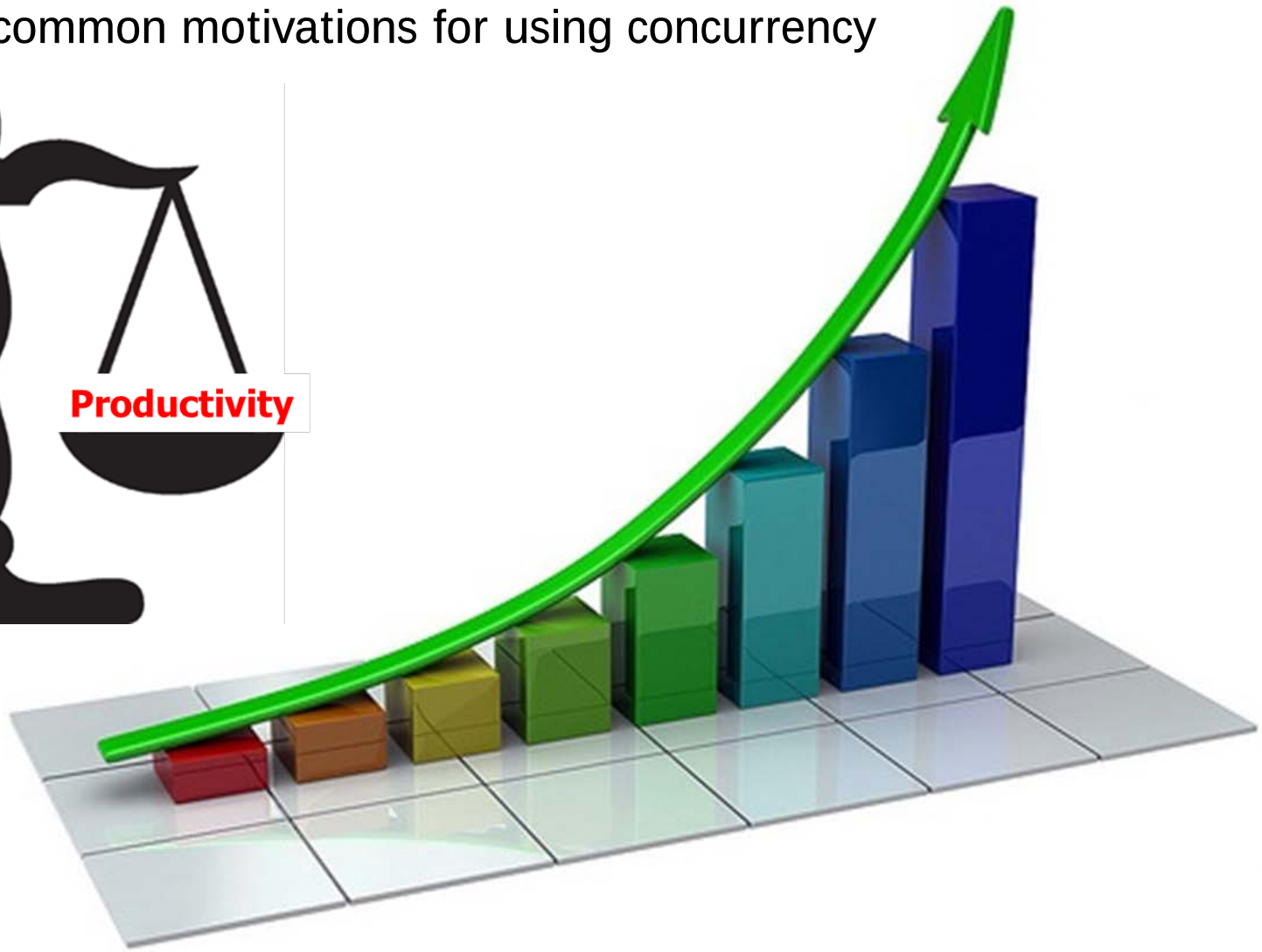
See en.wikipedia.org/wiki/Up_to_eleven for more on maximizing performance

Using Concurrency to Increase Performance

- Decreasing response time & increasing the amount of work performed within a given time are common motivations for using concurrency

Using Concurrency to Increase Performance

- Decreasing response time & increasing the amount of work performed within a given time are common motivations for using concurrency



Optimizing these properties requires an understanding of patterns & tradeoffs amongst constraints & quality attributes

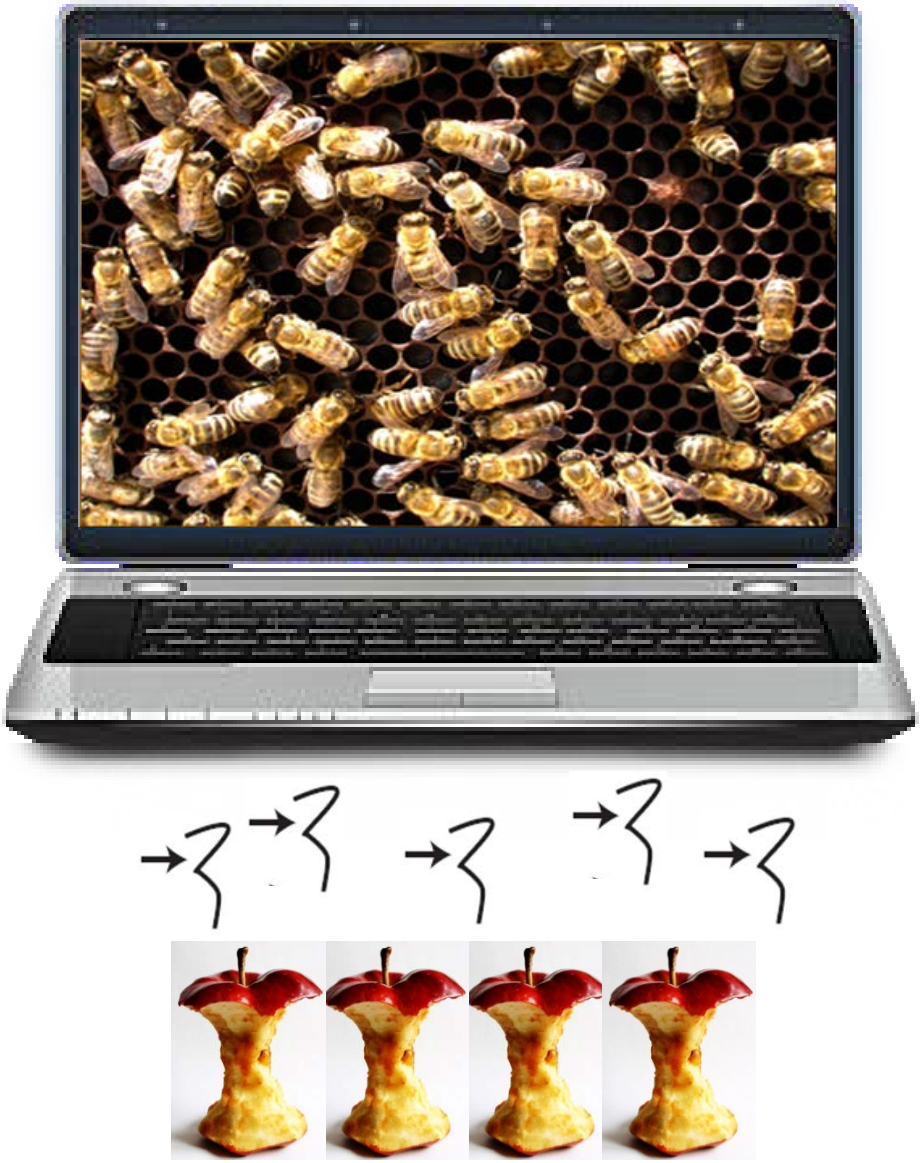
Using Concurrency to Increase Performance

- Performance can be accelerated via parallel processing

See en.wikipedia.org/wiki/Parallel_computing

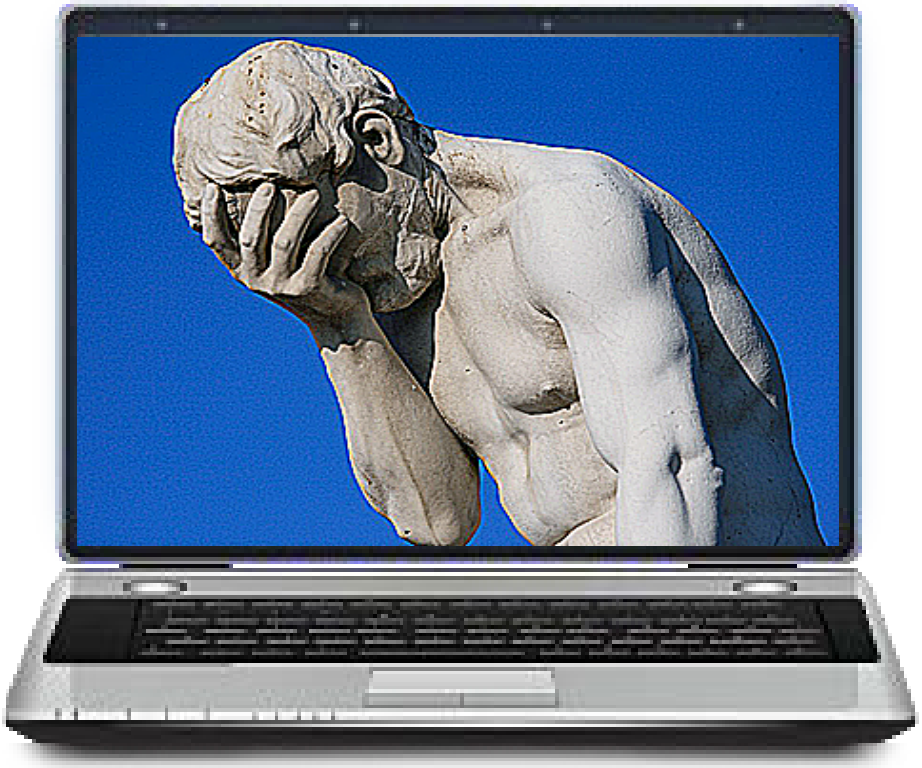
Using Concurrency to Increase Performance

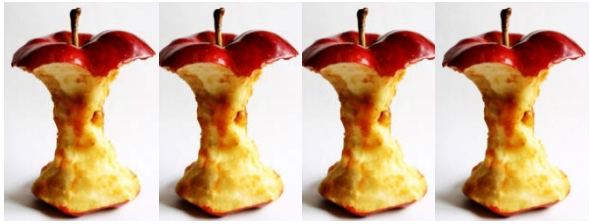
- Performance can be accelerated via parallel processing, e.g.
  - Performing computations simultaneously



Using Concurrency to Increase Performance

- Performance can be accelerated via parallel processing, e.g.
  - Performing computations simultaneously
    - Particularly for computations with no inter-dependencies

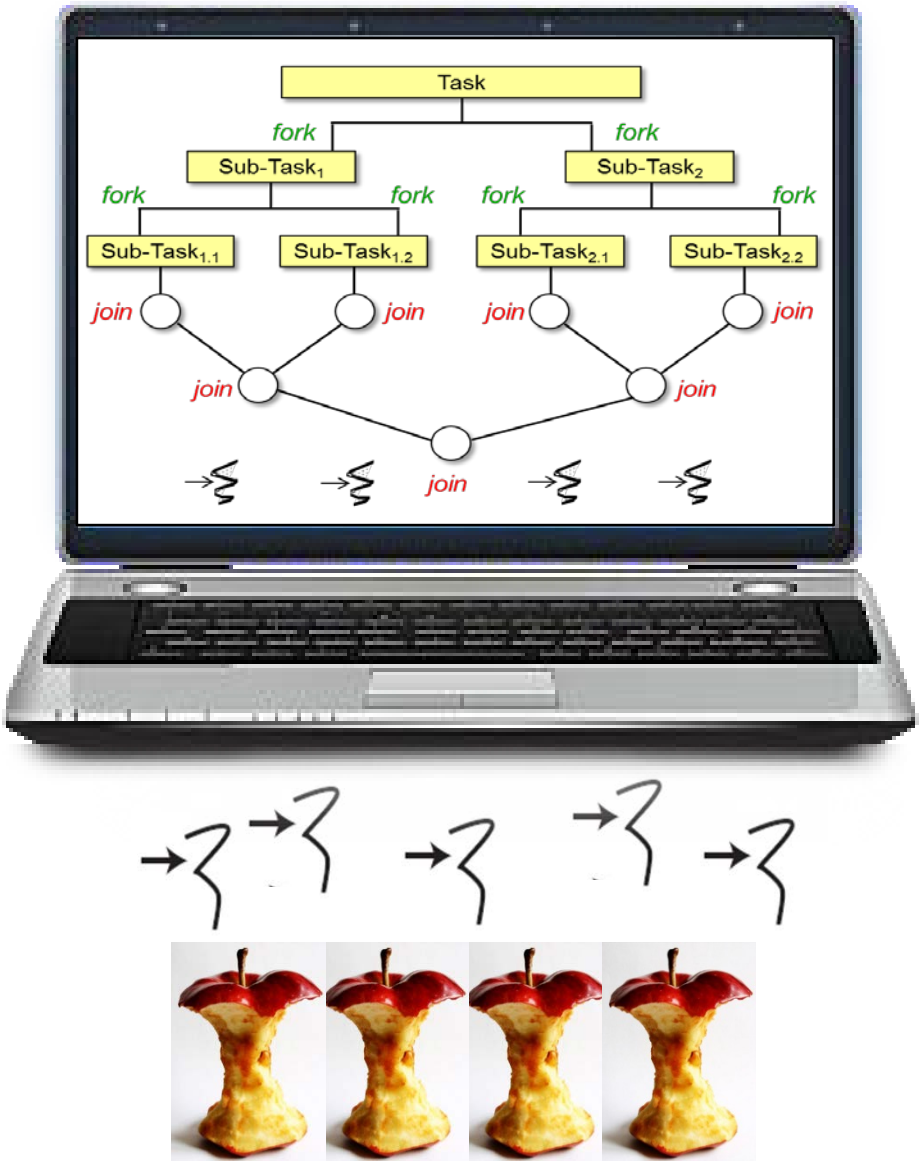




See en.wikipedia.org/wiki/Embarrassingly_parallel

Using Concurrency to Increase Performance

- Performance can be accelerated via parallel processing, e.g.
  - Performing computations simultaneously
  - Dividing a large problem into multiple smaller problems that can be processed in parallel



See en.wikipedia.org/wiki/Divide_and_conquer_algorithm

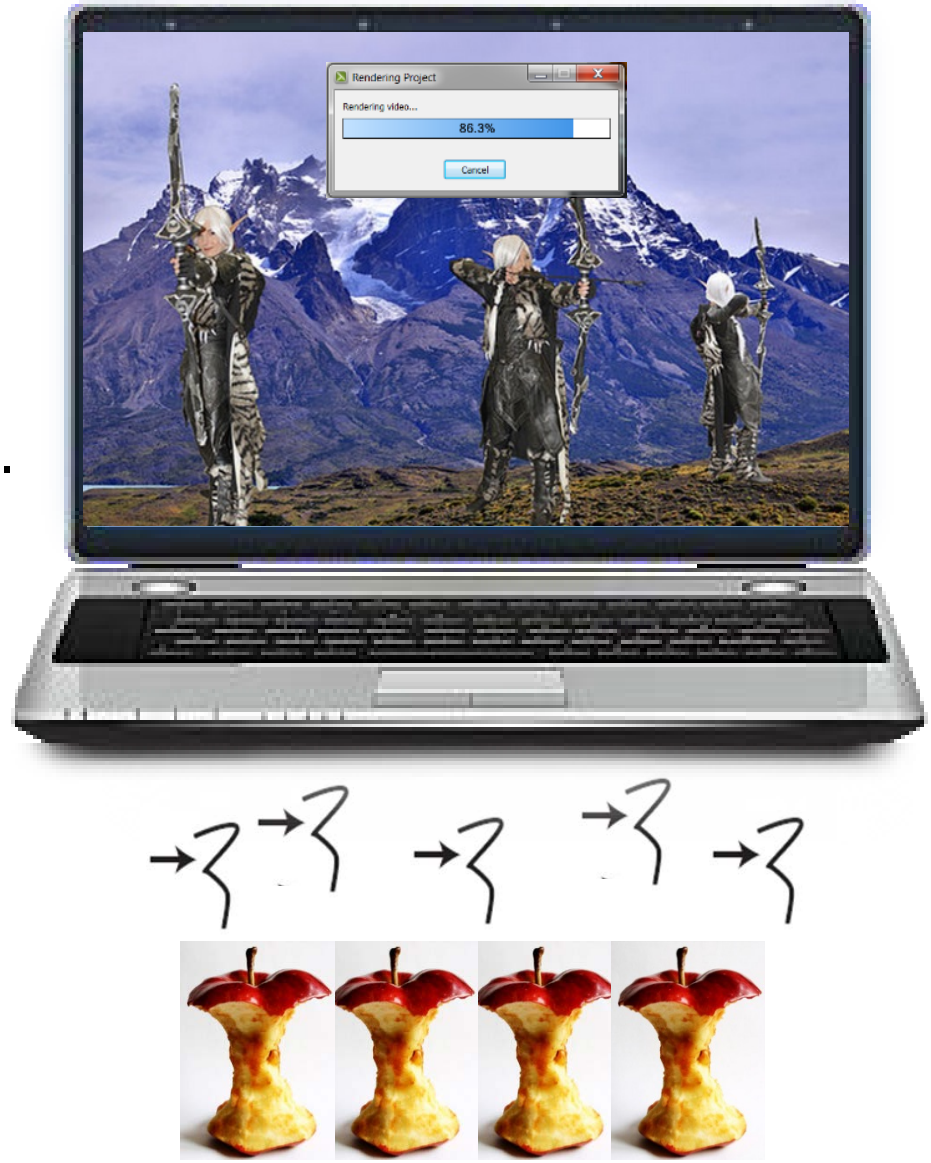
Using Concurrency to Increase Performance

- Performance can be accelerated via parallel processing, e.g.
  - Performing computations simultaneously
  - Dividing a large problem into multiple smaller problems that can be processed in parallel, e.g.
    - Web searches

See en.wikipedia.org/wiki/MapReduce

Using Concurrency to Increase Performance

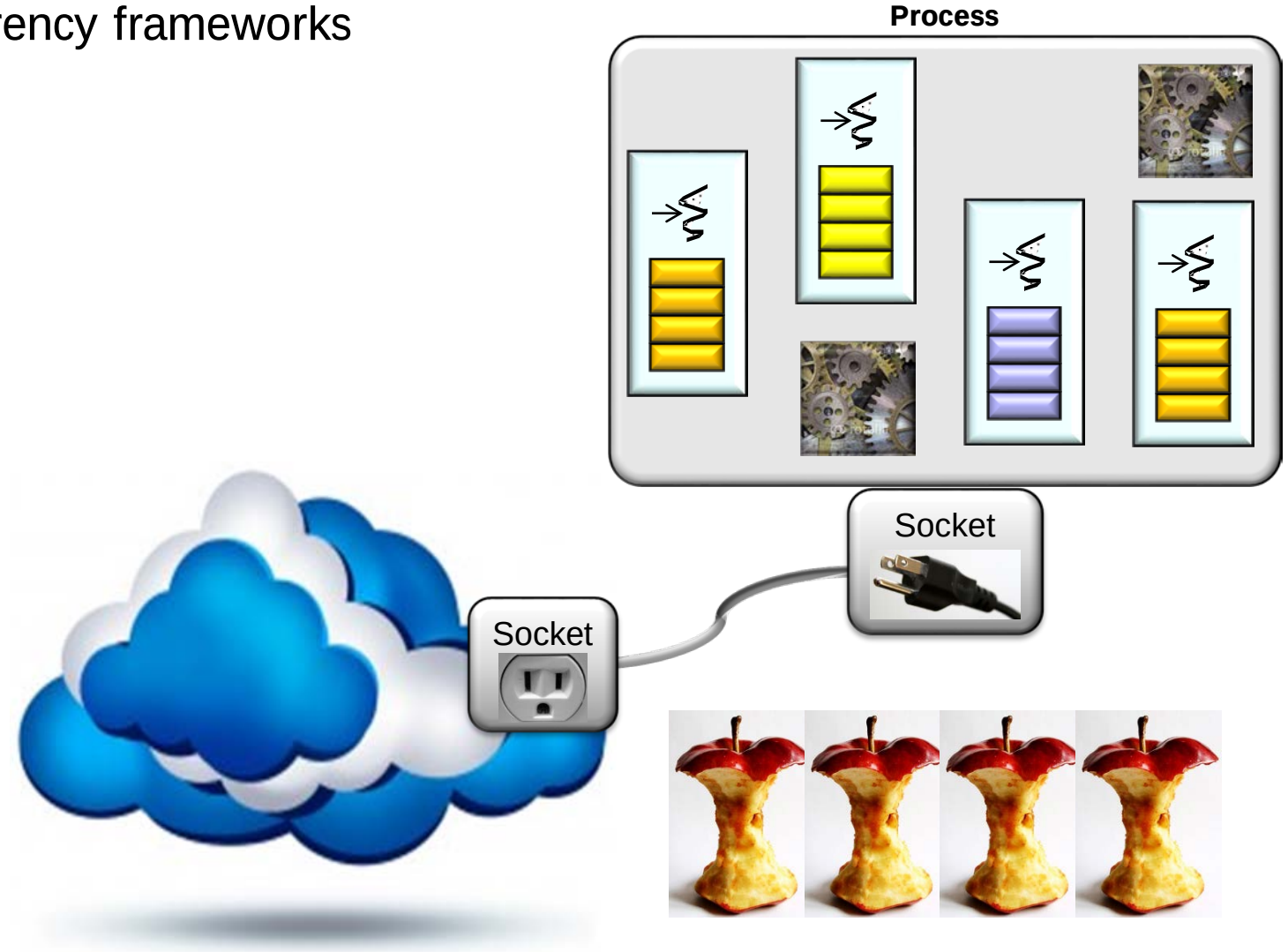
- Performance can be accelerated via parallel processing, e.g.
  - Performing computations simultaneously
  - Dividing a large problem into multiple smaller problems that can be processed in parallel, e.g.
    - Web searches
    - Image rendering



See en.wikipedia.org/wiki/Parallel_rendering

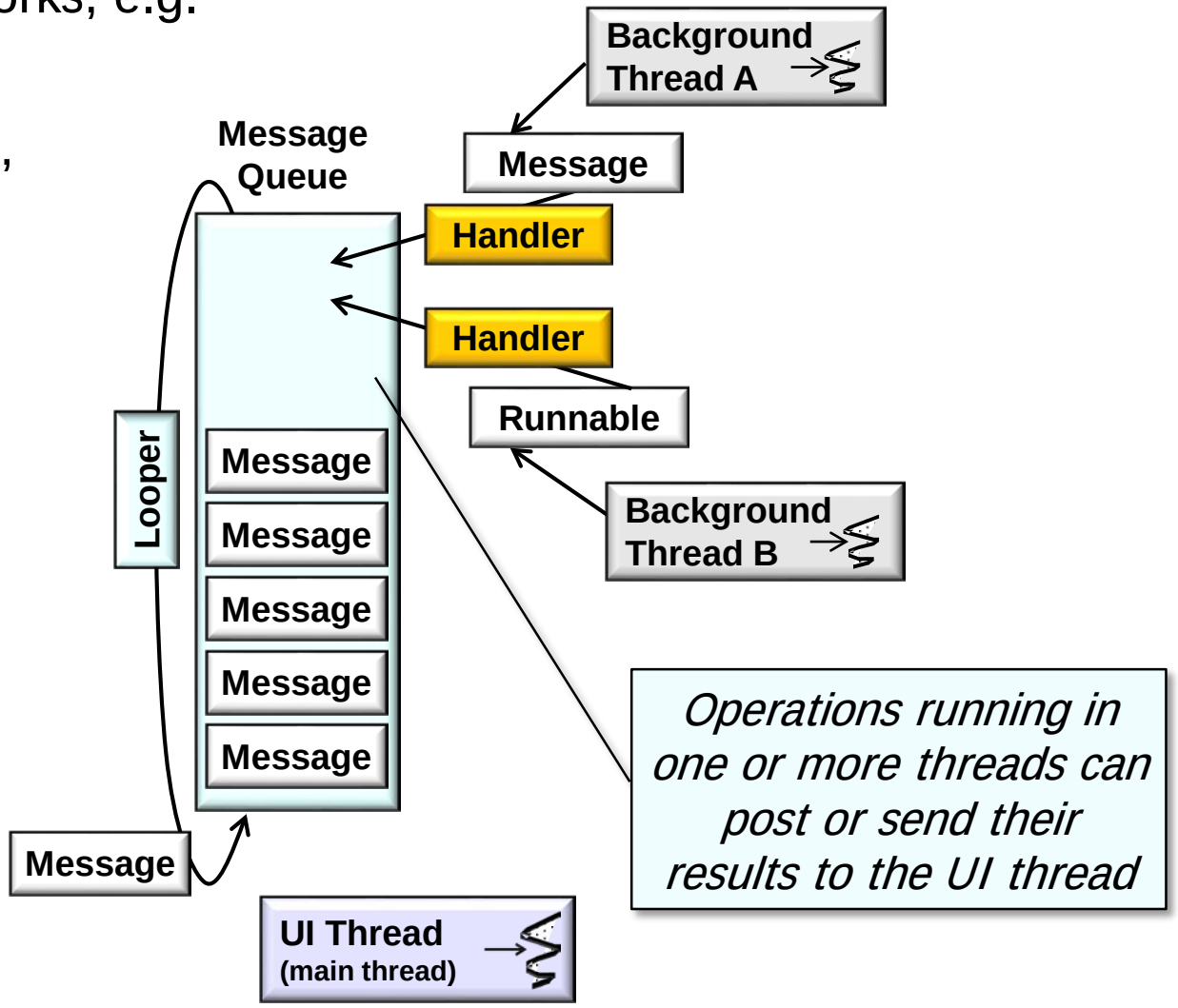
Using Concurrency to Increase Performance

- Android enables parallelism by overlapping computation & communication via two concurrency frameworks



Using Concurrency to Increase Performance

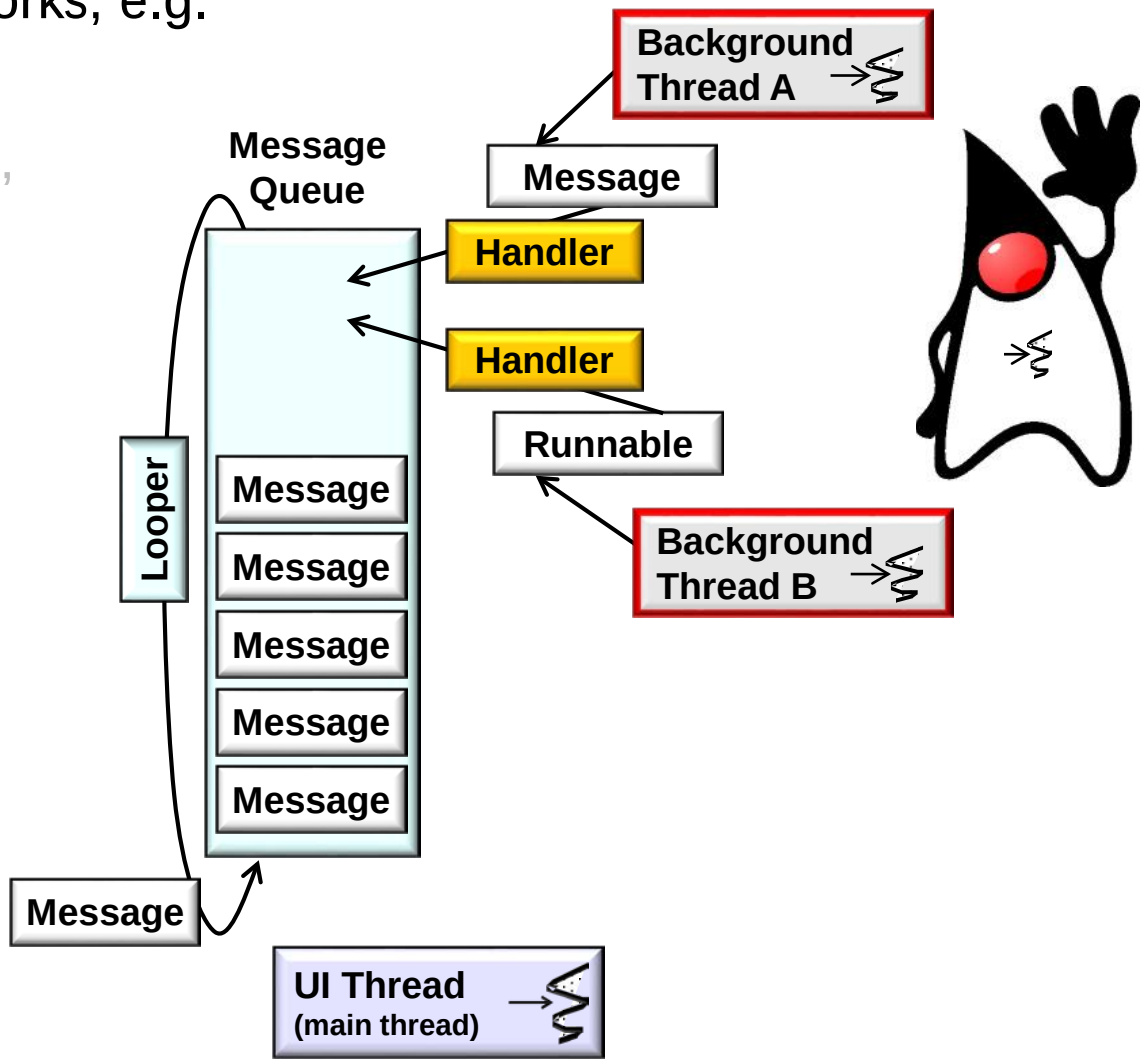
- Android enables parallelism by overlapping computation & communication via two concurrency frameworks, e.g.
  - HaMeR framework
    - "Handlers, Messages, & Runnables"



See code.tutsplus.com/tutorials/concurrency-on-android-using-hamer-framework--cms-27129

Using Concurrency to Increase Performance

- Android enables parallelism by overlapping computation & communication via two concurrency frameworks, e.g.
  - HaMeR framework
    - "Handlers, Messages, & Runnables"
  - Works with Java threads

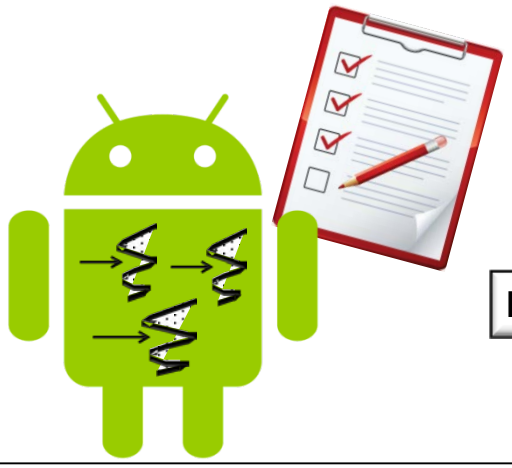


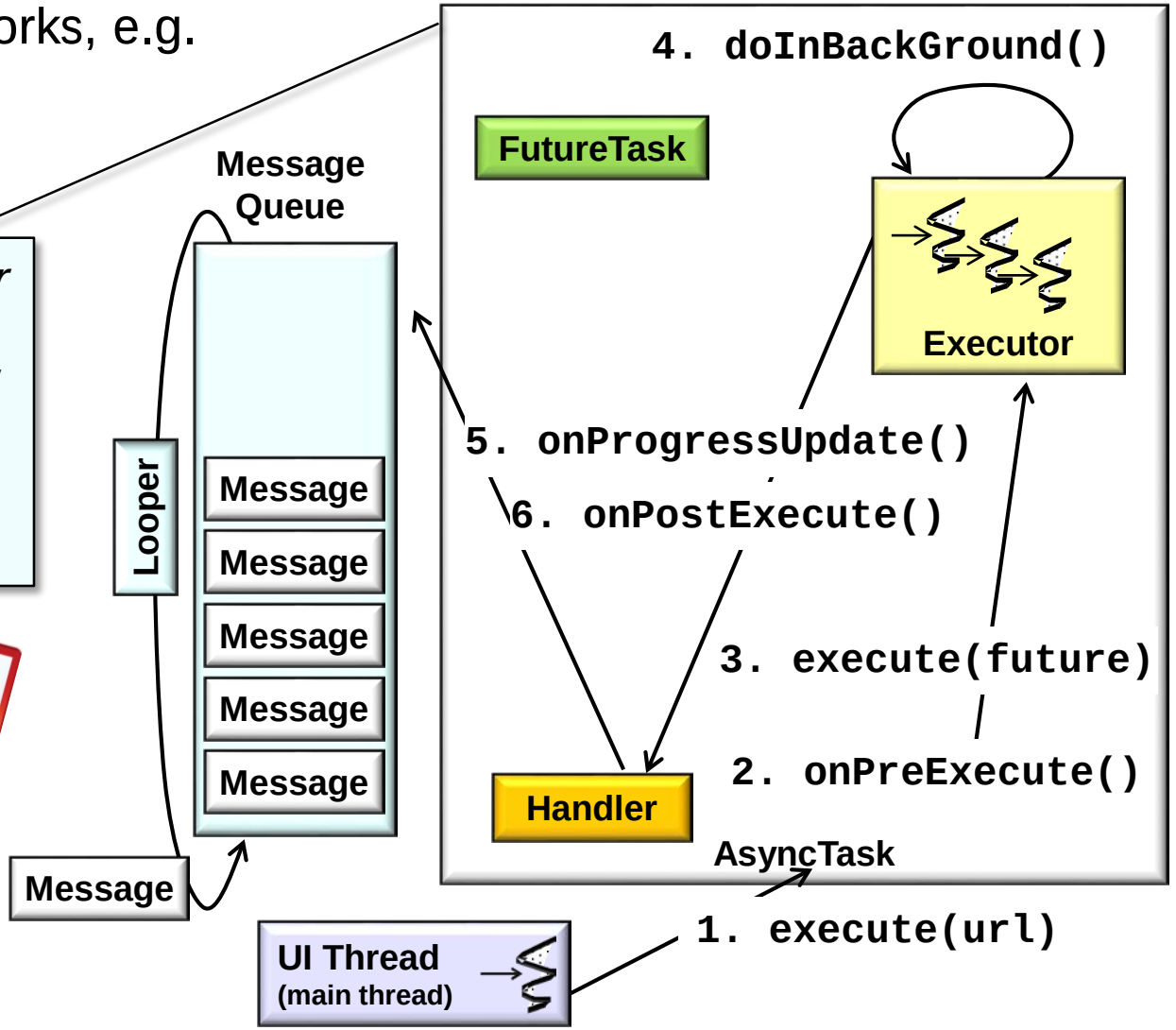
See docs.oracle.com/javase/tutorial/essential/concurrency/threads.html

Using Concurrency to Increase Performance

- Android enables parallelism by overlapping computation & communication via two concurrency frameworks, e.g.
  - HaMeR framework
  - AsyncTask framework

Operations run in one or more threads & publish results to the UI thread without using threads, handlers, messages, and/or runnables

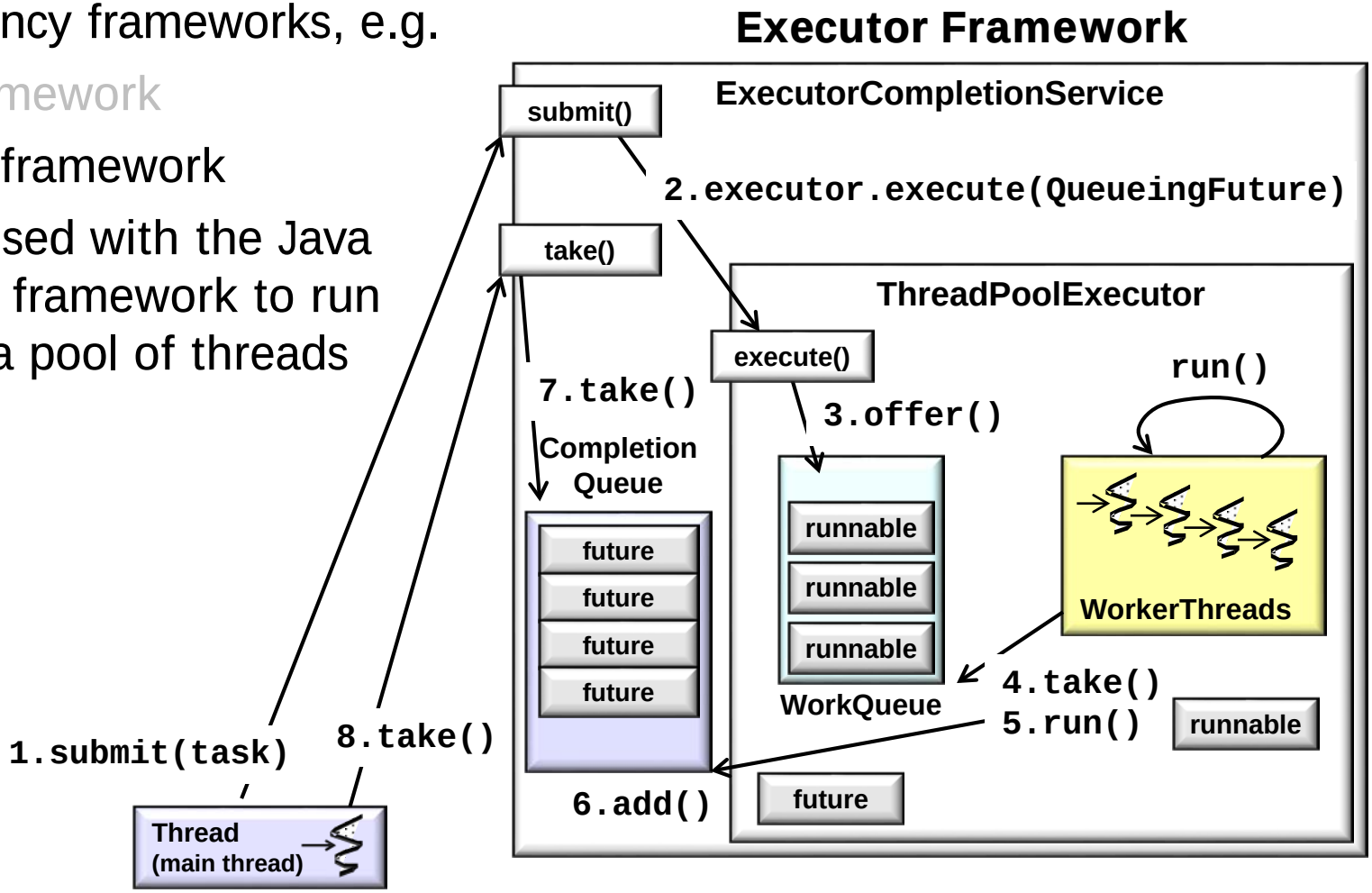




See developer.android.com/reference/android/os/AsyncTask.html

Using Concurrency to Increase Performance

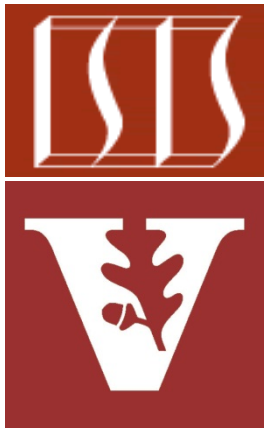
- Android enables parallelism by overlapping computation & communication via two concurrency frameworks, e.g.
  - HaMeR framework
  - AsyncTask framework
    - Can be used with the Java Executor framework to run tasks in a pool of threads



See docs.oracle.com/javase/tutorial/essential/concurrency/executors.html

End Benefits of Concurrency in Java & Android (Part 1)

Benefits of Concurrency in Java & Android: Responsiveness



Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Module

- Recognize how concurrency can improve performance in Java & Android
- Recognize how concurrency can improve responsiveness in Java & Android

Impediments to Leveraging Hardware Parallelism

Impediments to Leveraging Hardware Parallelism

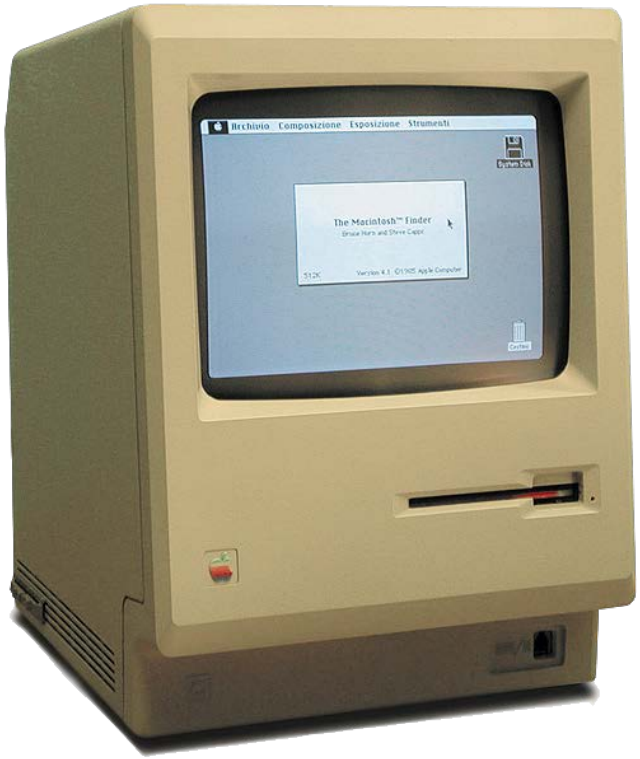
- Despite all the benefits of concurrency to improve performance, there are limits in practice to leveraging hardware parallelism

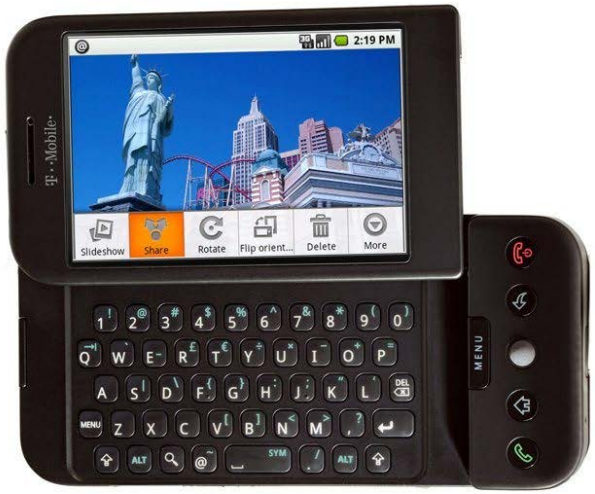
Impediments to Leveraging Hardware Parallelism

- Not every computing platform supports the latest hardware advances

Impediments to Leveraging Hardware Parallelism

- Not every computing platform supports the latest hardware advances
  - e.g., older computing devices just have a single core, which limits available parallelism





Impediments to Leveraging Hardware Parallelism

- It's also hard to fully leverage parallelism due to various impediments, e.g.

Impediments to Leveraging Hardware Parallelism

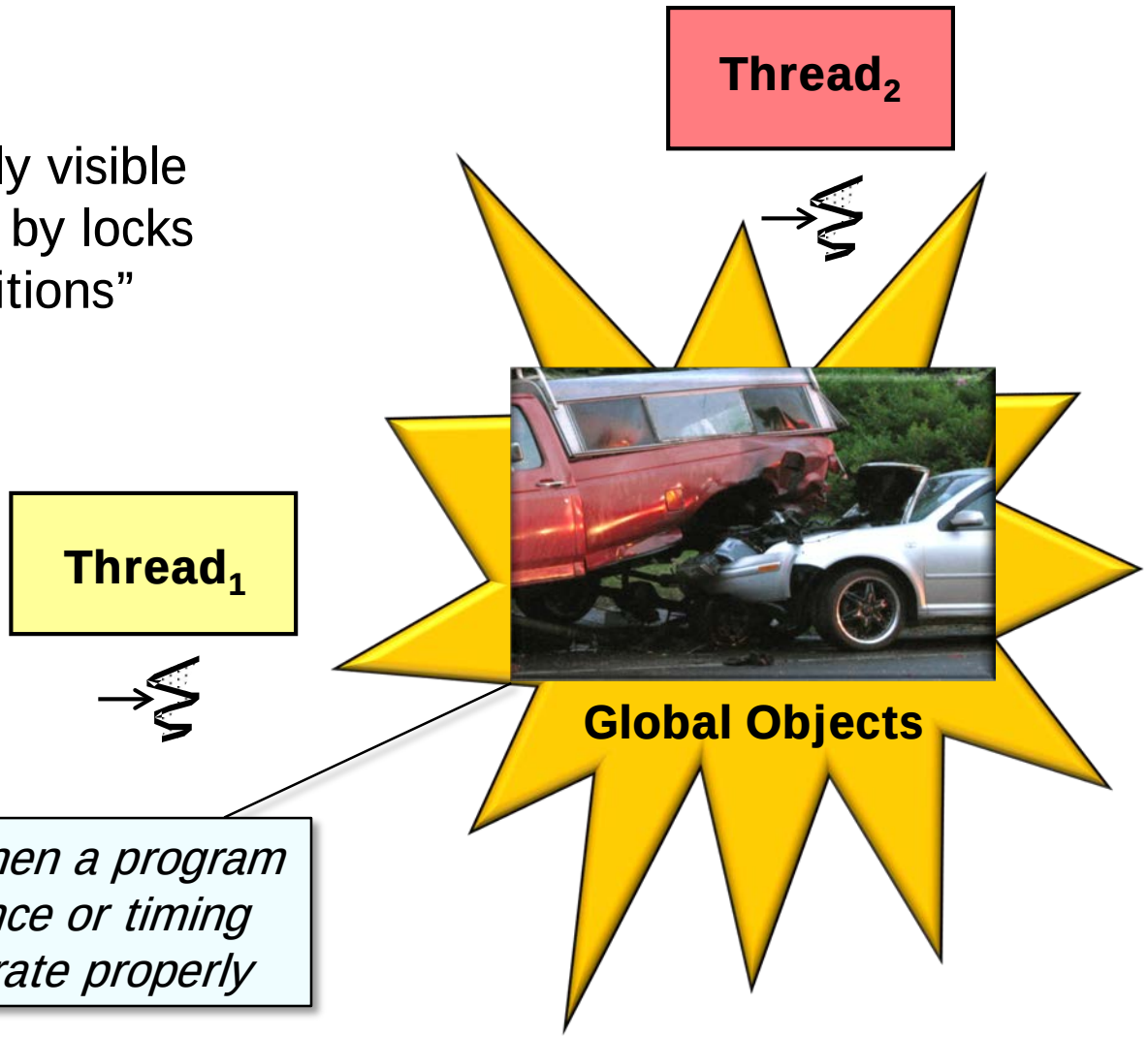
- It's also hard to fully leverage parallelism due to various impediments, e.g.
  - Legacy code that's not thread safe

See en.wikipedia.org/wiki/Legacy_system

See www.wesleysteiner.com/professional/MakingLegacyCodeSafe.html

Impediments to Leveraging Hardware Parallelism

- It's also hard to fully leverage parallelism due to various impediments, e.g.
  - Legacy code that's not thread safe
    - e.g., accessing globally visible objects not protected by locks can cause "race conditions"



Race conditions occur when a program depends on the sequence or timing of threads for it to operate properly

See en.wikipedia.org/wiki/Race_condition#Software

Impediments to Leveraging Hardware Parallelism

- It's also hard to fully leverage parallelism due to various impediments, e.g.
  - Legacy code that's not thread safe
  - GUI toolkits aren't thread -safe by design

JavaFX™

See community.oracle.com/blogs/kgh/2004/10/19/multithreaded-toolkits-failed-dream

Impediments to Leveraging Hardware Parallelism

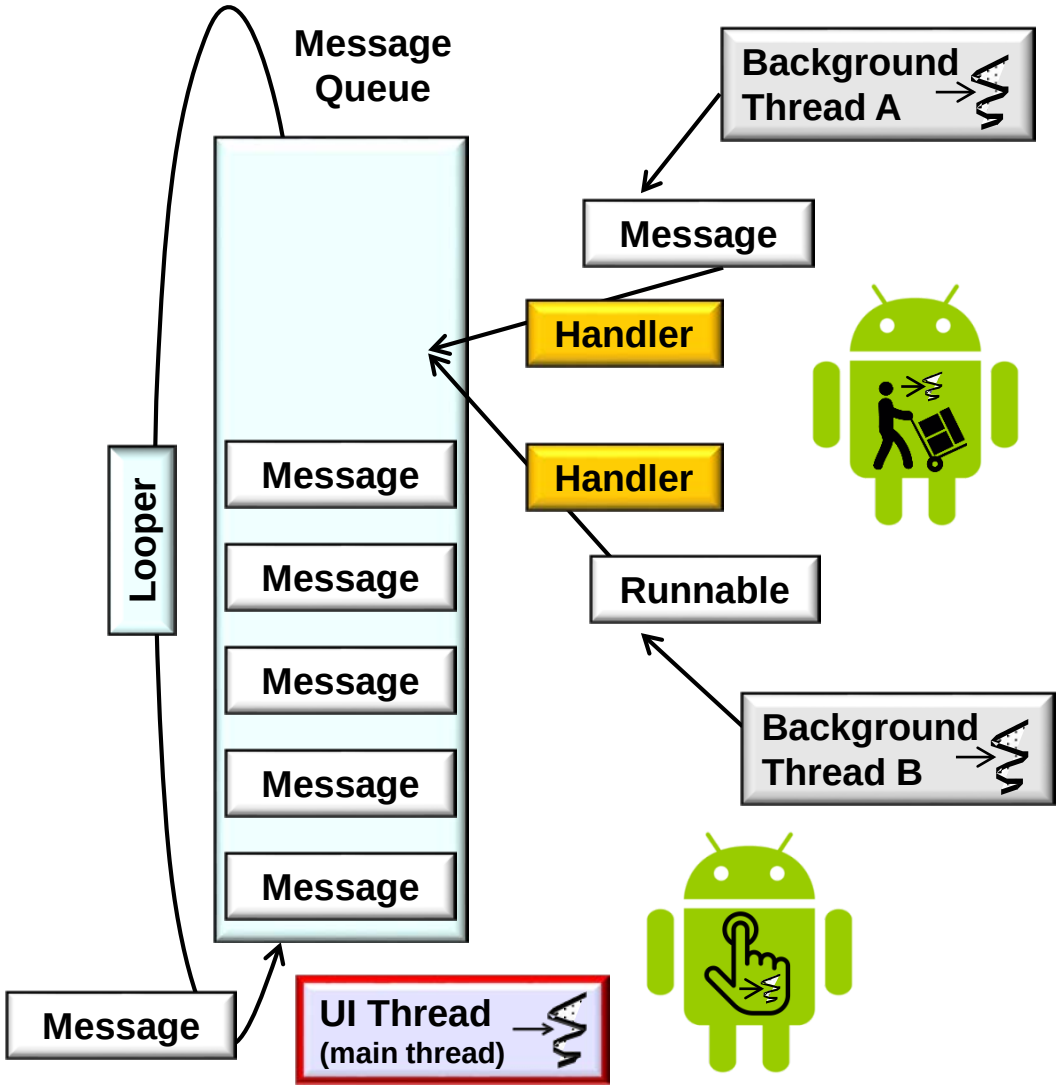
- It's also hard to fully leverage parallelism due to various impediments, e.g.
  - Legacy code that's not thread safe
  - GUI toolkits aren't thread -safe by design, e.g.
    - Eliminate the need for internal locking

Impediments to Leveraging Hardware Parallelism

- It's also hard to fully leverage parallelism due to various impediments, e.g.
  - Legacy code that's not thread safe
  - GUI toolkits aren't thread -safe by design, e.g.
    - Eliminate the need for internal locking
    - Minimize the need for app developers to understand concurrency

Impediments to Leveraging Hardware Parallelism

- It's also hard to fully leverage parallelism due to various impediments, e.g.
  - Legacy code that's not thread safe
  - GUI toolkits aren't thread -safe by design, e.g.
    - Eliminate the need for internal locking
    - Minimize the need for app developers to understand concurrency
    - Android only allows the UI thread to access GUI components

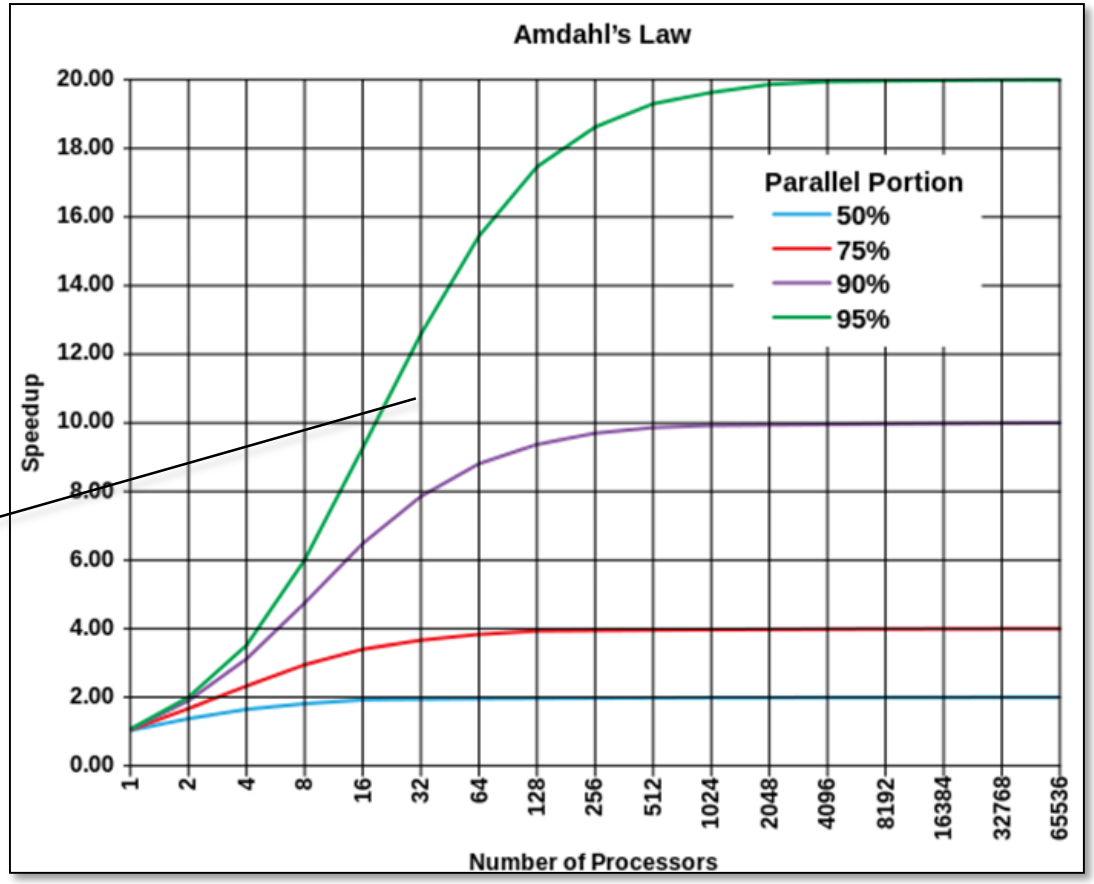


See developer.android.com/training/multiple-threads/communicate-ui.html

Impediments to Leveraging Hardware Parallelism

- It's also hard to fully leverage parallelism due to various impediments, e.g.
  - Legacy code that's not thread safe
  - GUI toolkits aren't thread -safe by design
  - The impact of Amdahl's Law

> *"The speedup of a program using multiple processors is limited by the sequential portion of the program that can't run in parallel"*



See en.wikipedia.org/wiki/Amdahl's_law

Using Concurrency to Improve Responsiveness

Using Concurrency to Improve Responsiveness

- Concurrency can often be used to improve *perceived* response time

See en.wikipedia.org/wiki/Responsiveness

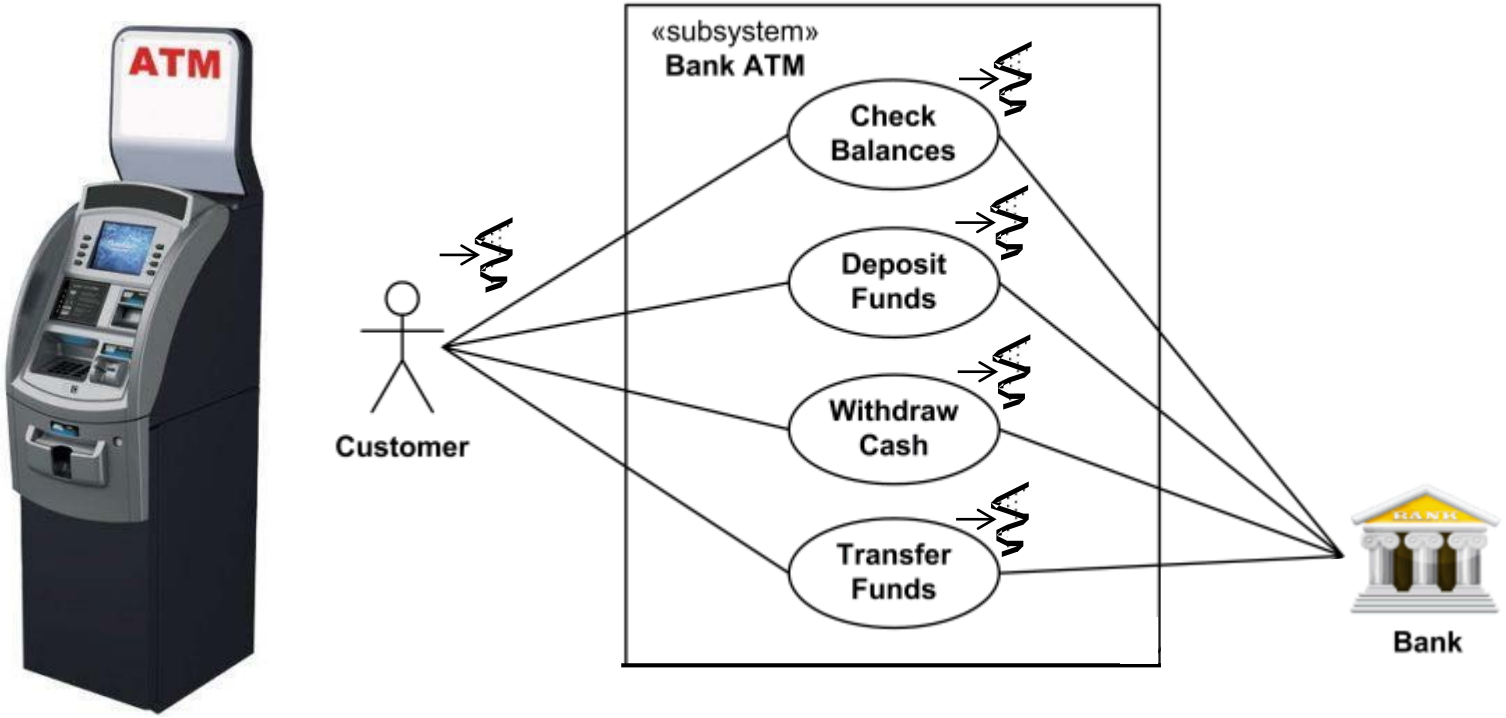
Using Concurrency to Improve Responsiveness

- Concurrency can often be used to improve *perceived* response time, e.g.
  - Don't ignore user input while long-duration computations or communications are occurring

See en.wikipedia.org/wiki/Spinning_pinwheel & en.wikipedia.org/wiki/Windows_wait_cursor

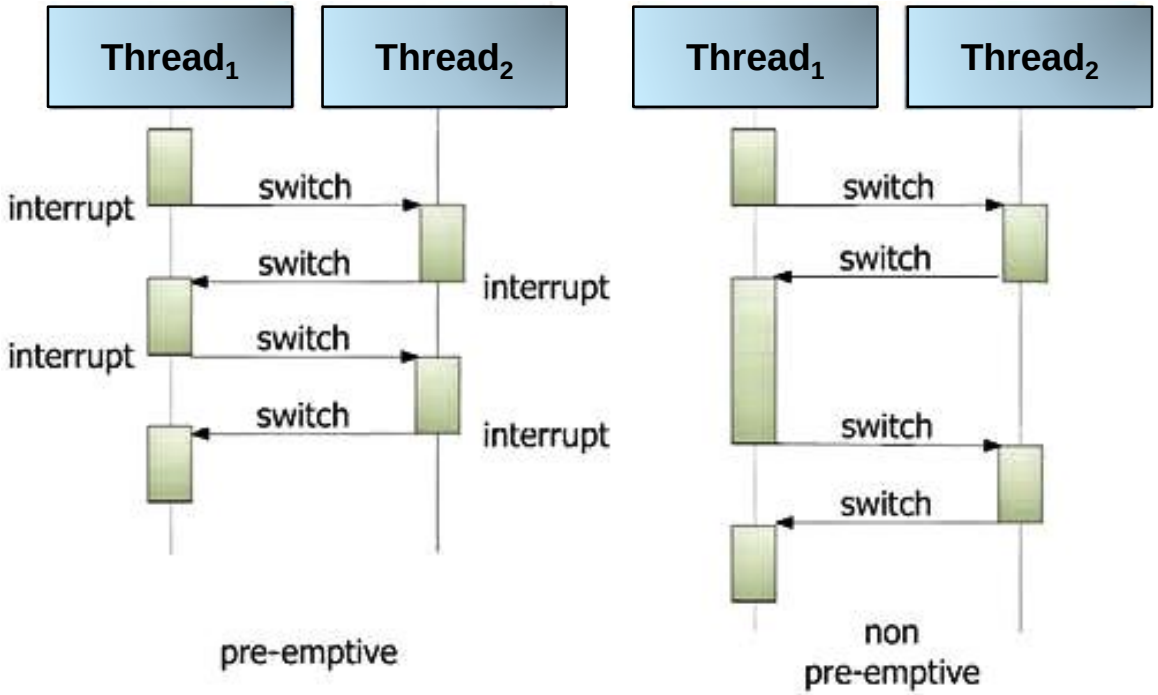
Using Concurrency to Improve Responsiveness

- Concurrency can often be used to improve *perceived* response time, e.g.
  - Don't ignore user input while long-duration computations or communications are occurring
    - e.g., allow worker threads to perform other processing in the background, while another thread handles user input



Using Concurrency to Improve Responsiveness

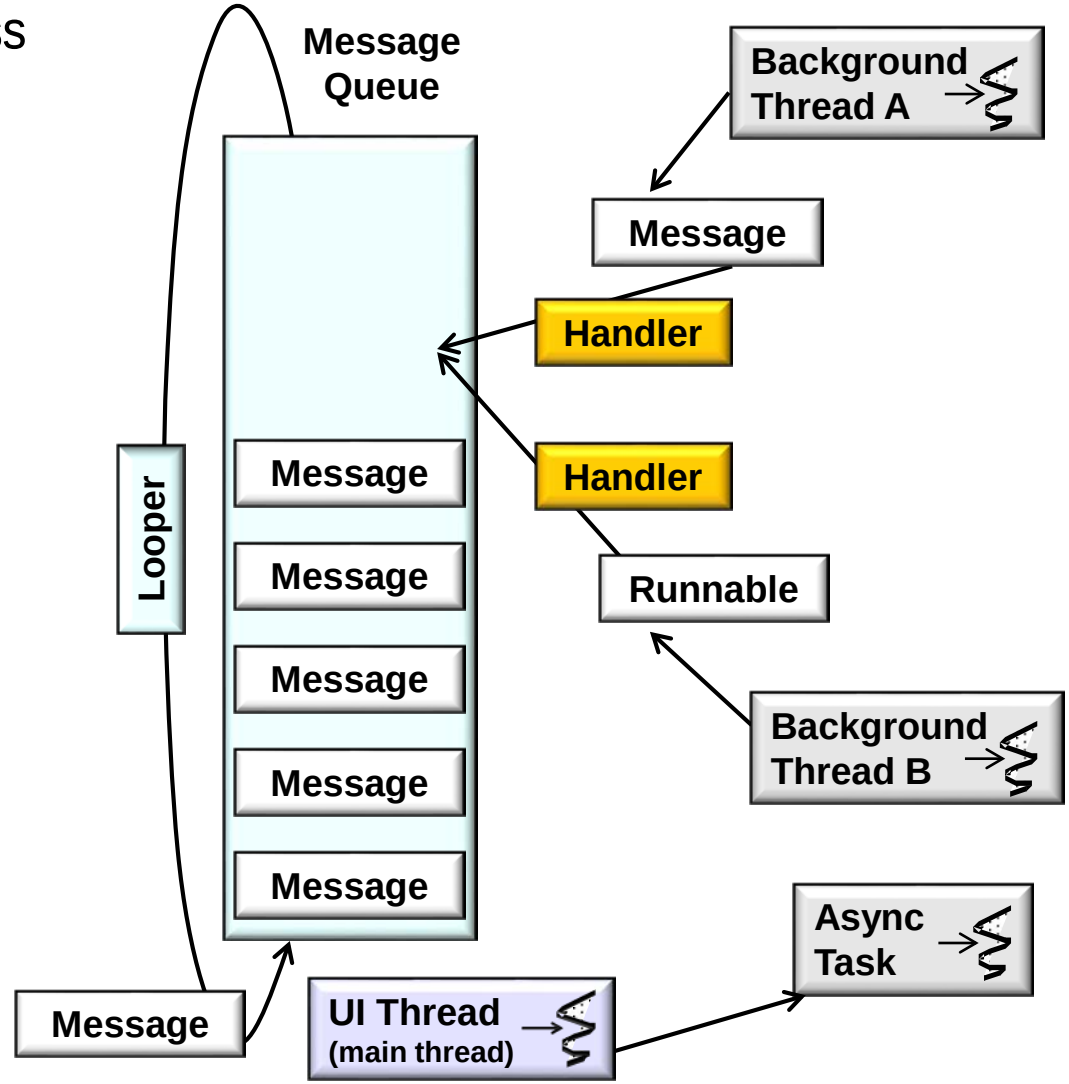
- Concurrency can often be used to improve *perceived* response time, e.g.
  - Don't ignore user input while long-duration computations or communications are occurring
  - As long as the software infrastructure supports preemptive multi-threading even single core hardware can be more responsive



See en.wikipedia.org/wiki/Preemption_(computing)

Using Concurrency to Improve Responsiveness

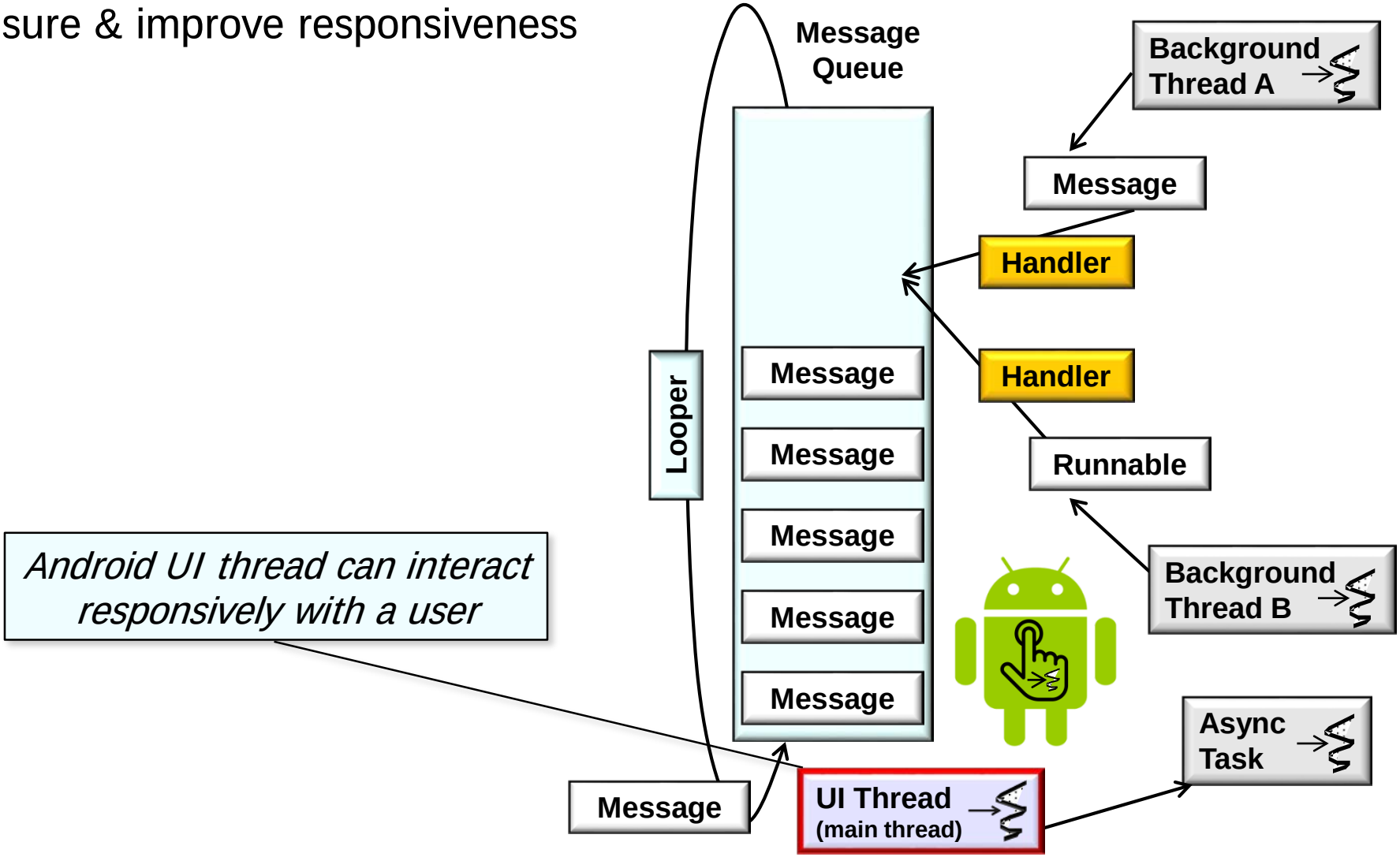
- Android concurrency frameworks define features & idioms that can help ensure & improve responsiveness



See developer.android.com/training/articles/perf-anr.html

Using Concurrency to Improve Responsiveness

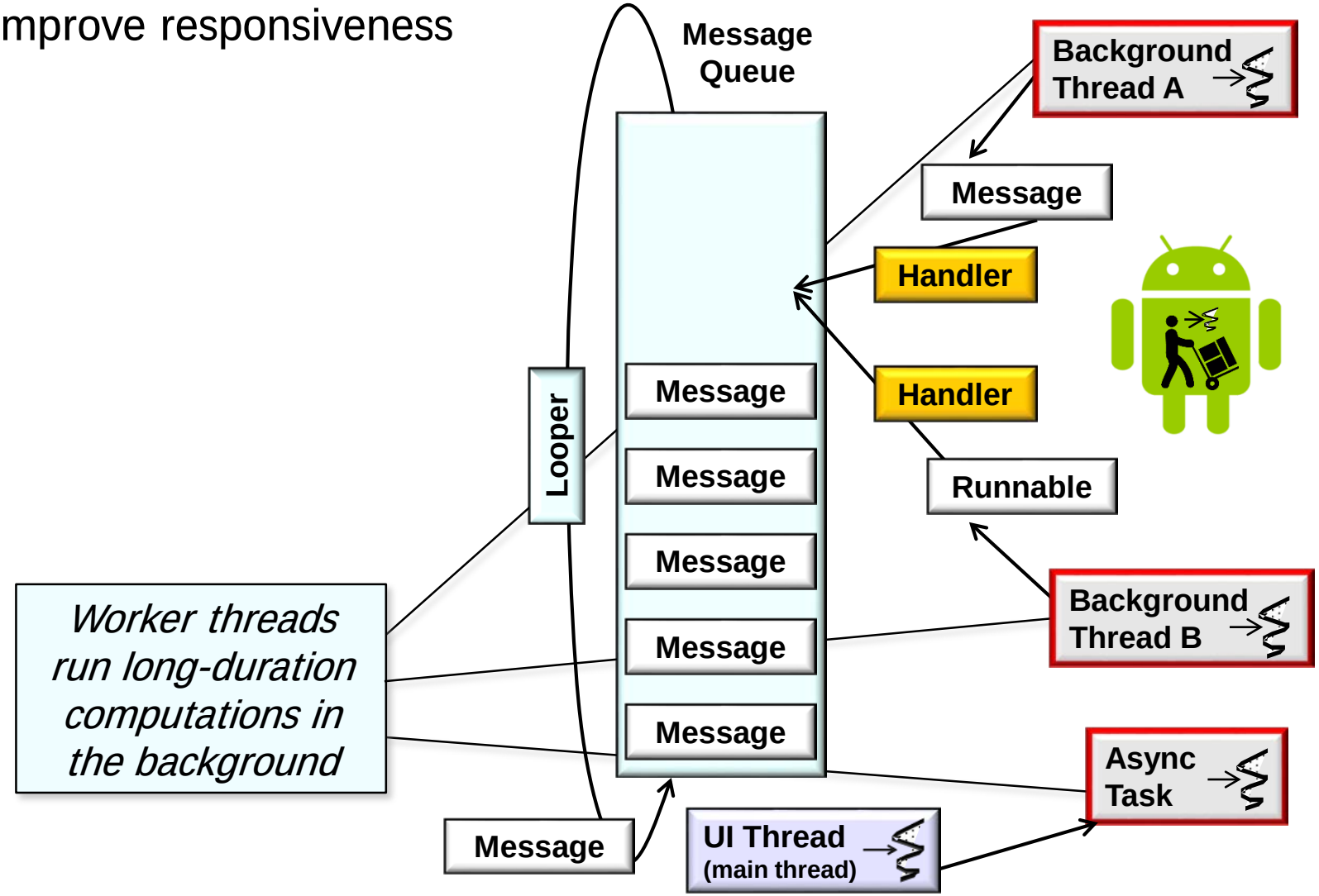
- Android concurrency frameworks define features & idioms that can help ensure & improve responsiveness



The UI thread never blocks for more than a few seconds

Using Concurrency to Improve Responsiveness

- Android concurrency frameworks define features & idioms that can help ensure & improve responsiveness



Worker threads can block waiting for I/O or computations to complete

End of Benefits of Concurrency in Java & Android (Part 2)