FOUNTAINHEAD

“Baruch MFE Big Data in Finance”

~ Class 3 ~

Baruch College
Wednesday 17th February 2016

Class 3

In this class we will cover these topics:

1. A “Golden Age” of computational discovery.
2. Synthetic data generation.
3. Running at scale.
4. Exam preparation. Quiz 3.
5. Thinking in Parallel: a worked example of CVA using GPUs.
6. Assignment A. Q&A.
7. Meet Bjarne Stroustrup? 13 Jun 2016. Inventor of C++.

Synthetic data sets

Generating Big Data sets synthetically is a useful skill:

1. Scripts.
2. Deterministic and reproducible. (V?)
3. Quantity. (V?)
4. Data ingestion. (V?)
5. Quality. (V?)
6. Format.
7. Security and privacy.
8. Proof-of-concept. Test before investing in collecting Big Data. (V?)
9. Hypothetical data. Hypothetical hardware.

Running at scale

Let’s start by looking at things that make scaling difficult:

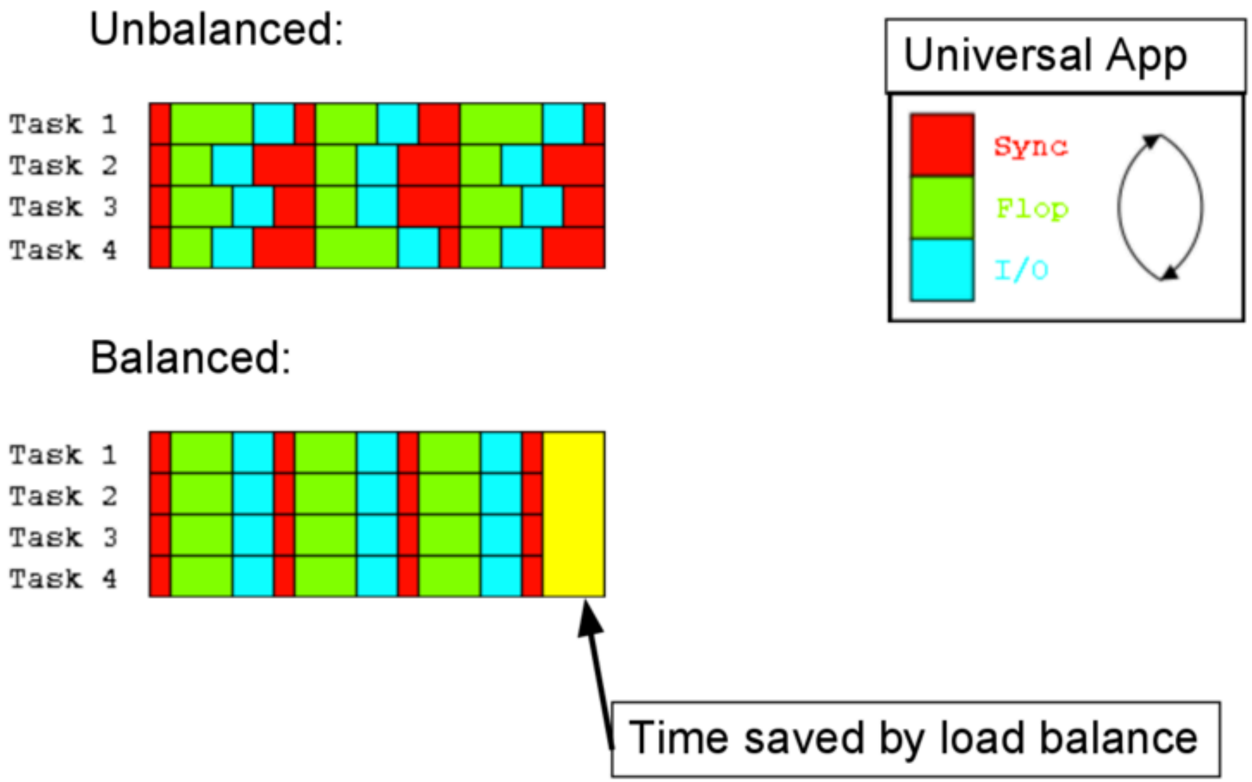
1. Limited resources.
2. Dependencies. (Amdahl’s law.)
3. Communication. Messages. Too much chatter.
4. Communication. Data movement. Too little bandwidth.
5. Poor programming. Program structure. Algorithms.
6. Poor programming. Program resources.
7. Data growth rate.
8. Increasing complexity of data.

Running at scale (cont.)

General principles of scaling:

1. Strong scaling: fixed data size (Amdahl’s law).
2. Weak scaling: data size grows with resources (Gustafson's law).
3. In general, changing problem size concurrency expose or remove compute resources.
4. Bottlenecks shift.
5. In general, first bottleneck wins.
6. Scaling brings additional resources too. More CPUs (of course). More cache(s). More memory BW in some cases.

Running at scale: load balancing



Unbalanced:

Task 1
Task 2
Task 3
Task 4

Balanced:

Task 1
Task 2
Task 3
Task 4

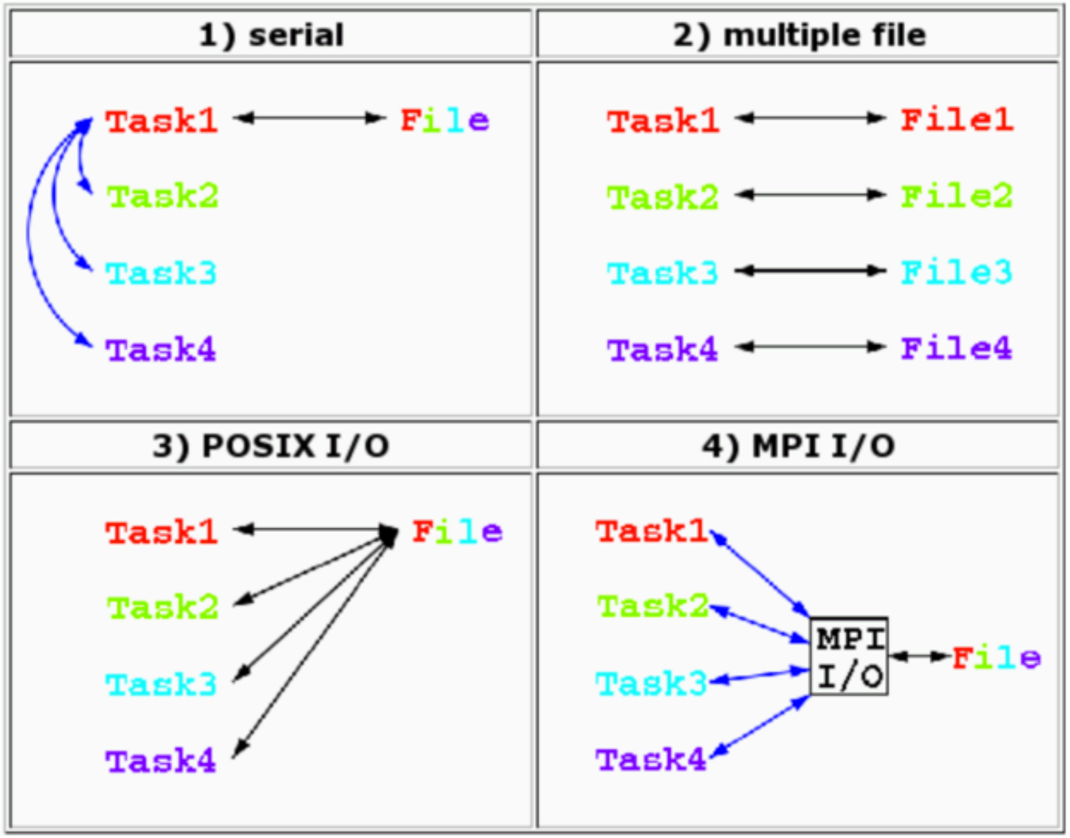
Universal App

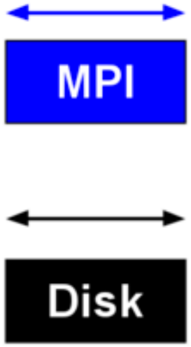
🟥 Sync
🟩 Flop
🟦 I/O

Time saved by load balance

Fountainhead

Running at scale: I/O





Some strategies fall down at scale

Running at scale: sync

Synchronization occurs are two conceptual levels:

1. Programmer: As a programmer you can control …
    1. Which MPI calls you use (it’s not required to use them all!).
    2. Message sizes, problem size (sometimes).
    3. The temporal granularity of synchronization.
2. Machine/OS: Language writers and system architects control …
    1. Influence over last two items above.
    2. The intrinsic amount of noise in the machine. What else is going on (e.g. JVM garbage collection).

Running at scale: key points

Parallel programming, and running at scale, is challenging:

1. Parallel programs are easier to mess up than serial ones! (One reason to always start with a serial version first.)
2. There are fundamental limits to parallelism (Amdahl and Gustafson's laws).
3. Things that work on a small scale have a tendency to break down at large scale. (True of all things in life?)
4. Never assume an order of execution, unless you program it in.
5. Exascale computing. MPI already runs on systems with 1.6m cores!

FOUNTAINHEAD

Exam preparation

Quiz 3 – model answers

Assignment A

Questions?

FOUNTAINHEAD

Thinking in Parallel

Enterprise-wide CVA at scale using GPUs

FOUNTAINHEAD

Meet Bjarne Stroustrup? (13 Jun)