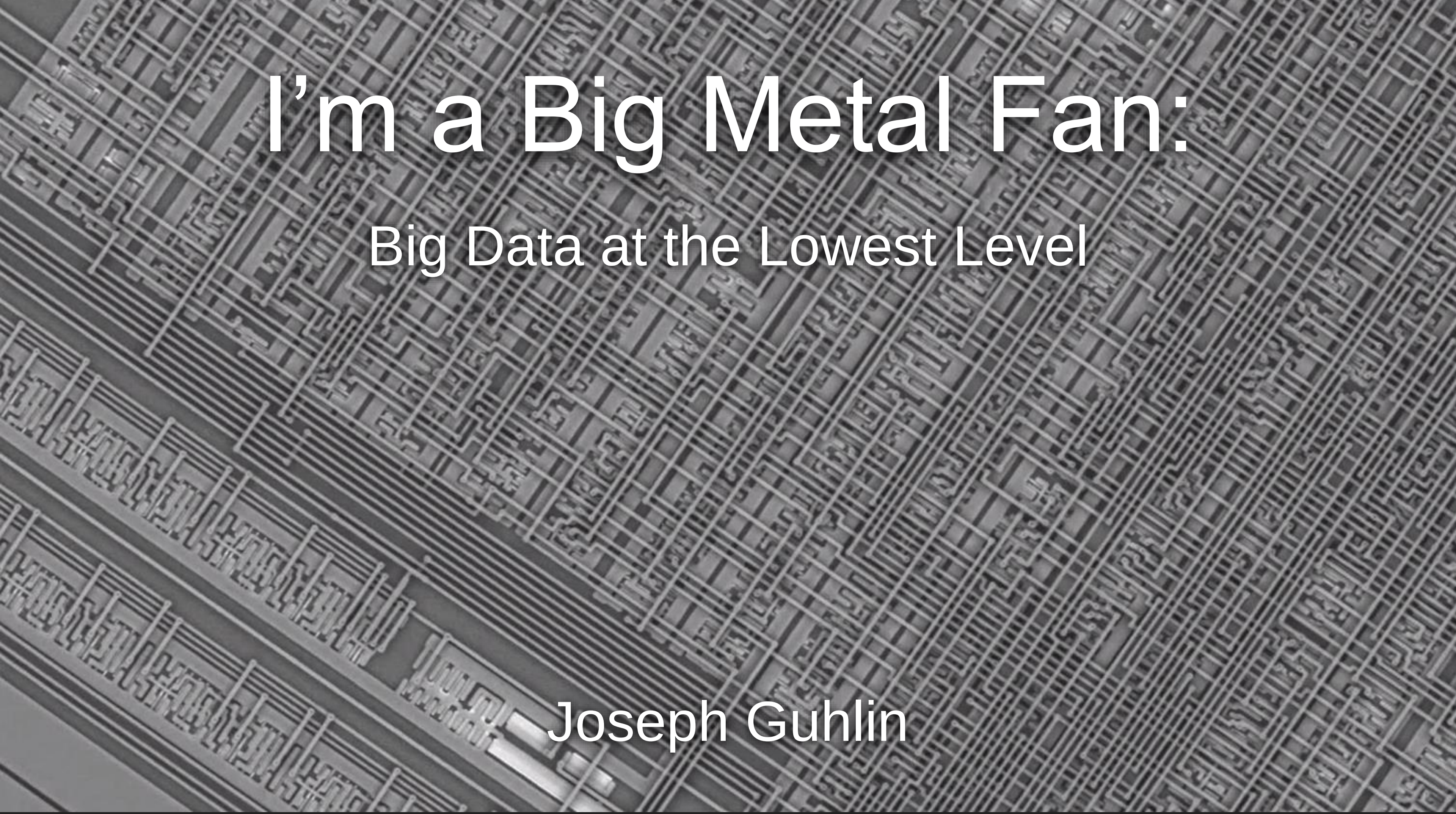
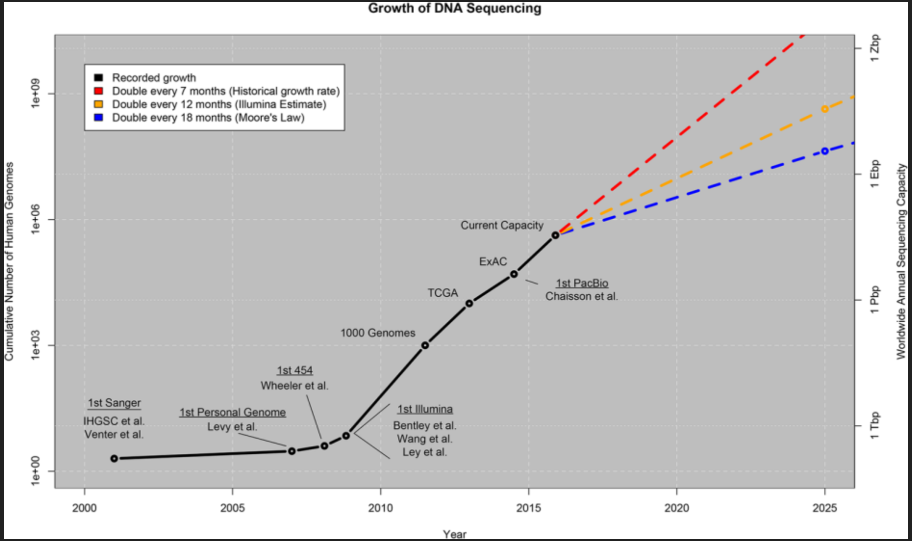


I’m a Big Metal Fan:

Big Data at the Lowest Level

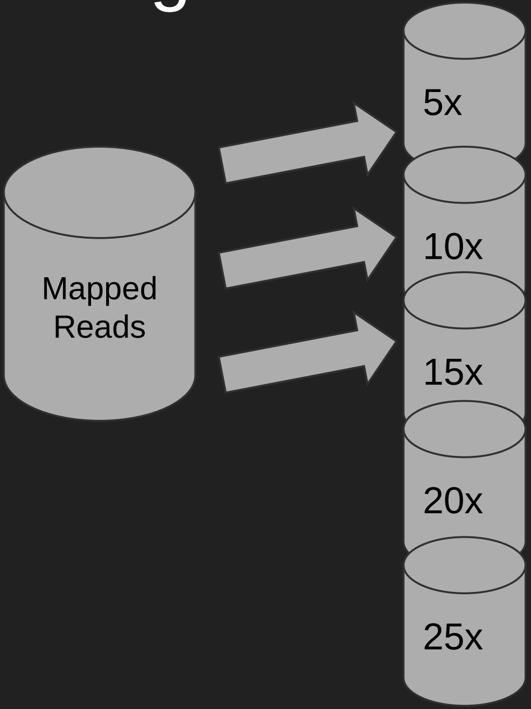
Joseph Guhlin

More Data Requires More Planning



Big Data: Astronomical or Genomical?
Stephens et al, 2015

Subsampling for ML



Mapped Reads

5x

10x

15x

20x

25x

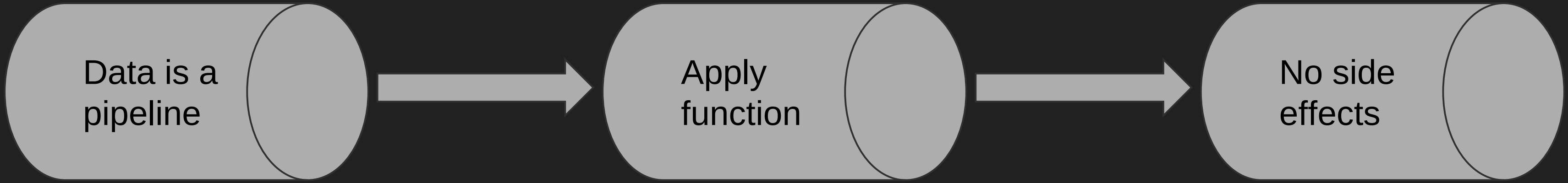
Most Problems are Smaller Problems

Mapping Reads

- QC
- Trim
- Align
- Process

Also applies to “smaller” problems!

- Extract every 1000bp of Sequence
- Split at 3 or more contiguous N’s
- Append Sequence Identifier



Data is a pipeline

Apply function

No side effects

Ways of Doing This

Workflow Managers → Snakemake, Nextflow

Simple Methods → Multiple Threads

Programming Paradigms → Map/Reduce

→

Scatter/Gather

1Mb Genomes vs 10Gb Genomes

Algorithms/software often written for the dataset you **have**

Throwing Hardware at the problem **mostly** works

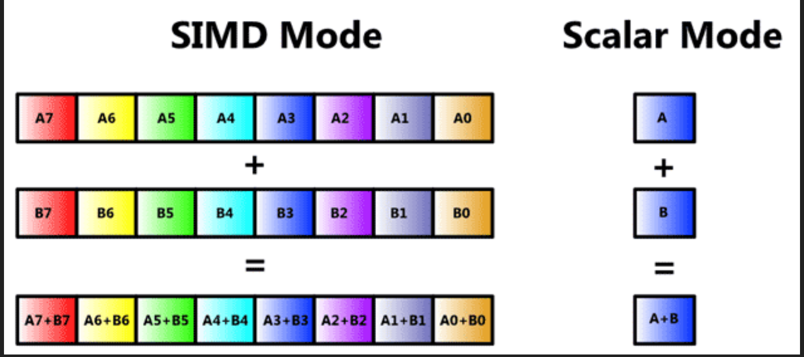
Think about scaling to larger genomes, deeper reads

Common Mistakes:

- Loading all data into memory (memory map the file → easiest solution)
- Nothing in parallel
- Waiting on one function to complete before starting the next step

Getting Fancy

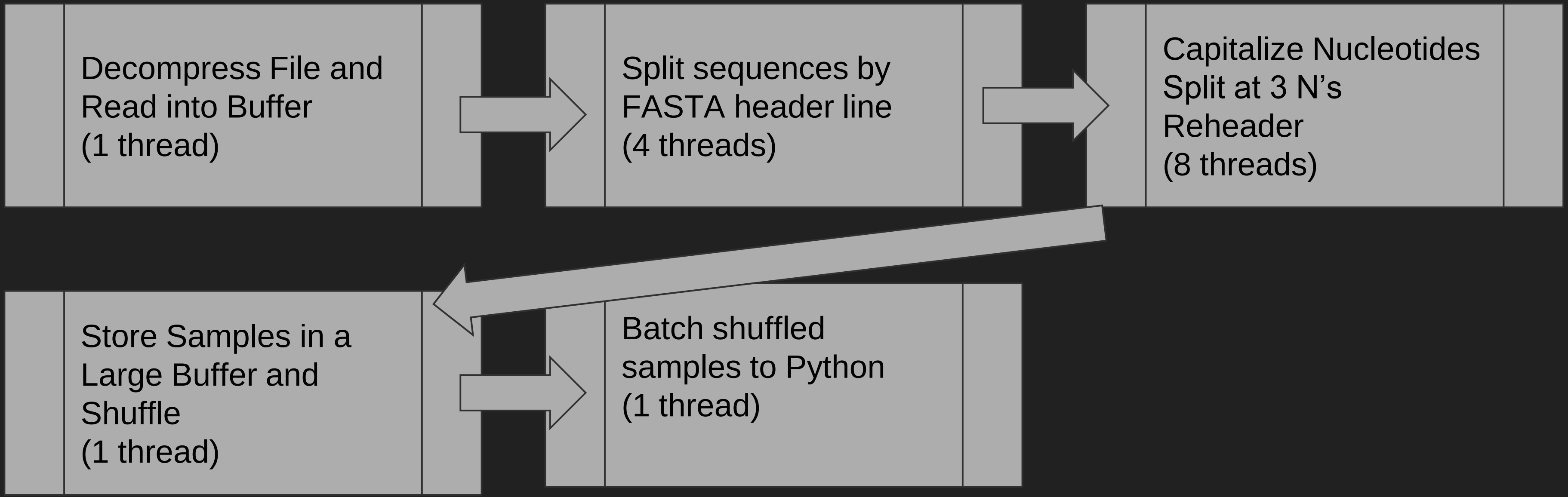
- Python / R
  - Language Interop
  - Develop intensive tasks in other languages
- Processor Intrinsics
  - SIMD - Single Instruction Multiple Data
  - Only certain types of data fit!



Rust → Python

Rust is a Systems Language → Can write Python Modules

Multiple-hours in Python to Minutes using Rust (but still working in Python)



Decompress File and Read into Buffer (1 thread)

Split sequences by FASTA header line (4 threads)

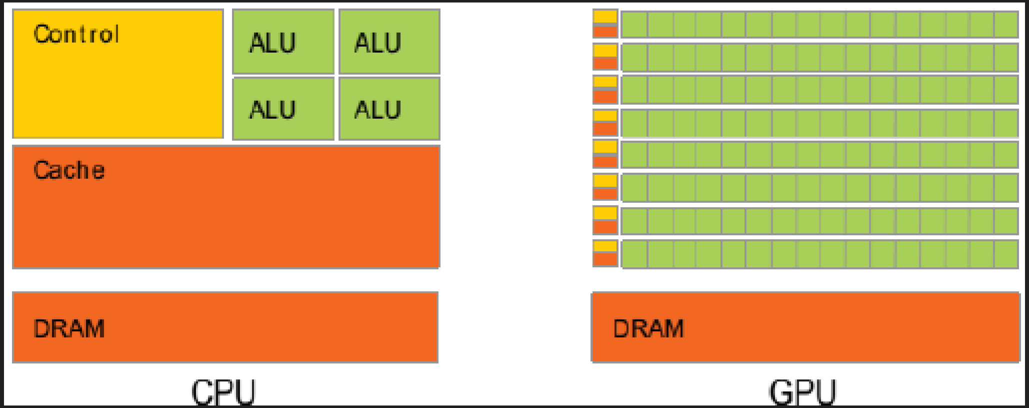
Capitalize Nucleotides
Split at 3 N's
Reheader
(8 threads)

Store Samples in a Large Buffer and Shuffle (1 thread)

Batch shuffled samples to Python (1 thread)

GPUs

- SIMD on Steroids
- Many more mathematical processes simultaneously
- Limited for Genomics → But if you can represent your problem in mathematical terms...



You Can Too!

- Python
  - NUMBA → Vectorizes, SIMD, compiles Python code
  - CUDA → Steeper learning curve, GPU
  - MxNET Gluon -> Linear Algebra on CPU or GPU
- R
  - BLAS/LAPACK
  - Microsoft R Open → SIMD, Multithreading
    - Bonus: Reproducibility with checkpointed CRAN

Summary

- Subsets of Problems
    - Break problems down into small solvable units
- Data as a Pipe
    - Push forward, never backwards
    - Copy/Clone data to be solved
- How can I scale this up?
    - Memory limited or CPU limited?
    - Bacteria now → 10Gb genome tomorrow?