

Read mapping

Michael Schatz

Sept 21, 2022

Lecture 7: Applied Comparative Genomics



Assignment 2: Genome Assembly

Due Monday Sept 19 by 11:59pm

The screenshot shows a GitHub repository page for 'appliedgenomics2022' with the path 'main/assignments/assignment2'. The repository has 11 stars and 11 forks. A pull request titled 'master update assignment2 assignment' is shown, last updated 10 minutes ago. The main content area displays the assignment details:

Assignment 2: Genome Assembly

Assignment Date: Monday, September 19, 2022
Due Date: Monday, September 19, 2022 11:59pm

Assignment Overview

In this assignment, you are given a set of unassembled reads from a mysterious pathogen that contains a secret message encoded somewhere in the genome. The secret message will be recognizable as a novel insertion of sequence not found in the reference. Your task is to assess the quality of the reads, assemble the genome, identify and decode the secret message. If all goes well the secret message should describe a recognizable organism; otherwise troubleshoot your assembly and try again. As a reminder, any questions about the assignment should be posted to Piazza.

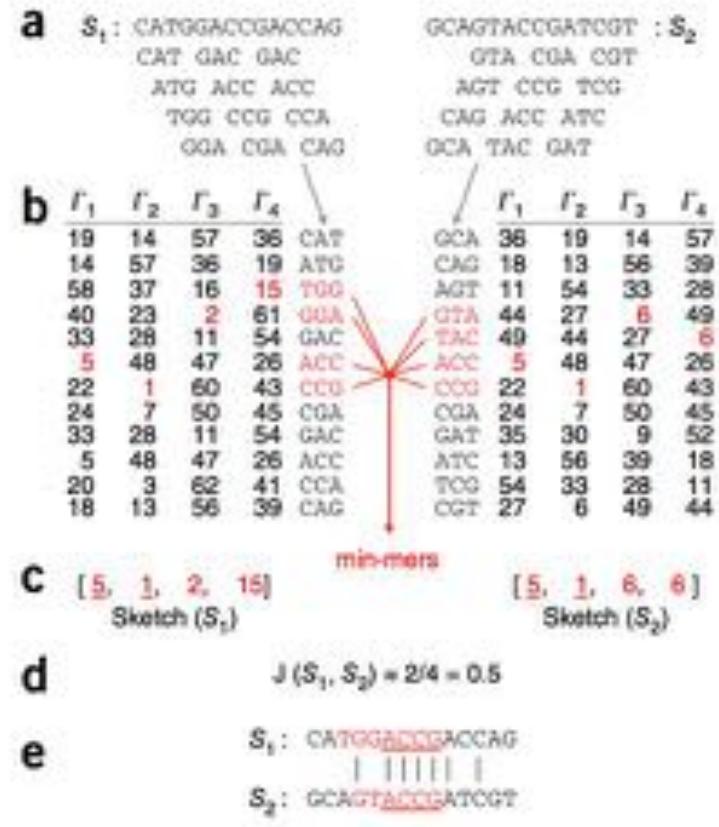
This assignment can be completed on your local machine or run the code using [bioconda](#). There are notes for both in the Resources section. Alternatively, you can try running the code using [Docker](#). Docker is a powerful containerization tool to make software easier to distribute. The ref-

<https://github.com/schatzlab/appliedgenomics2022/tree/main/assignments/assignment2>
Check Piazza for questions!

Very fast approximate overlapping

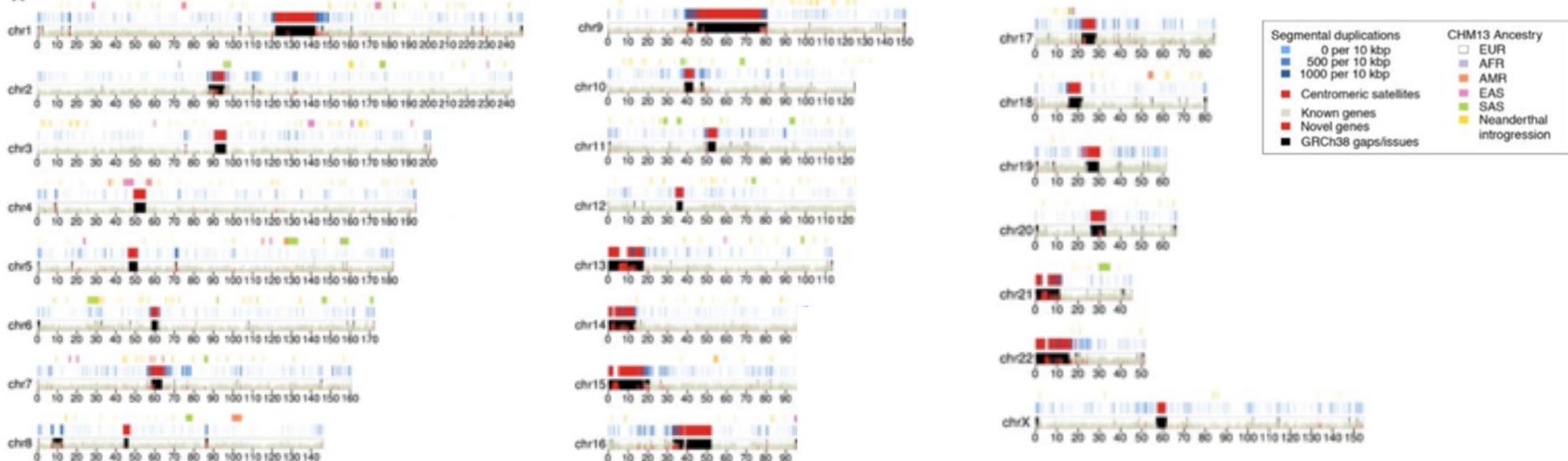
Maybe we don't need to compute the exact identity of the overlap region, just approximate it

- If two reads overlap, they should share many of the same kmers: Their Jaccard coefficient should be high: $|\text{intersection}| / |\text{union}|$
- But tracking all of the kmers for a read is a lot of overhead
- Instead, compare the “sketch” of the reads: a small fraction of kmers carefully chosen
- LSH: Find the sketch by applying N hash functions to the kmers, and keeping the minimum hash values reported from each (N=4 in example)
- This forms a nice “random” sample of the reads, and the Jaccard coefficient is a good approximation of the sequence similarity



The complete sequence of a human genome

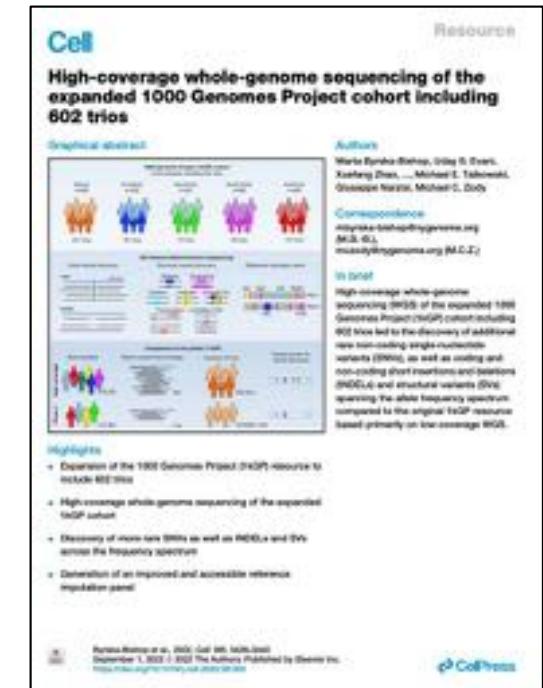
A



CHM13v1.1 genome size is **3.057 Gbp with zero Ns**
Every chromosome is telomere-to-telomere, quality estimated >Q70
~190 Mbp (~8%) of new sequence vs. GRCh38, fixes thousands of errors

(Nurk et al. Science, 2022)

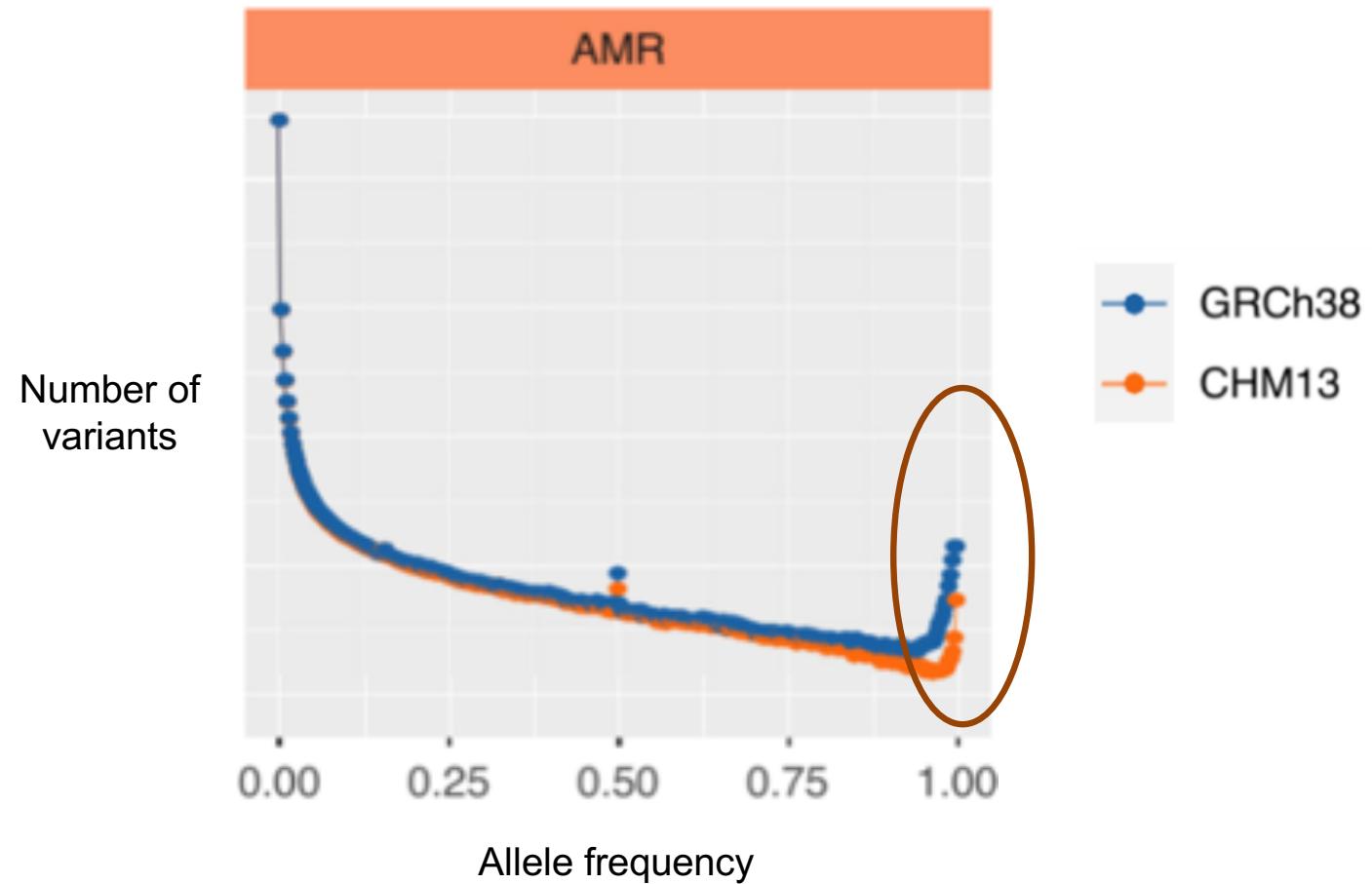
T2T Variants: 1000 Genomes Project



26 populations from 5 superpopulations (continental regions)
3202 samples (2504 core genomes + 698 offspring)

3202 samples x 30Gb = 96Tb input data | >5Pb of intermediate data | >>1M core hours

Explaining Decreased Per-Sample Count





Science

HOME · EDITIONS · COMMUNITY · JOURNALS · SEARCH · LOGIN / REGISTRATION · ADVERTISE & SPONSOR

COMPLETING THE HUMAN GENOME

A fully sequenced human genome was announced more than 10 years ago. Since then, owing to technological limitations, some genomic regions remained unsequenced. Here, *Science* and other journals present research by the International Human Genome Sequencing Consortium, reporting on the evidence to complete a comprehensive human reference genome.

4 RESULTS FOUND

View all results

View all references

Segmental duplications and their variation in a complete human genome

International Human Genome Sequencing Consortium, Nils G. Autio, Christopher L. Brinkman, Michael C. Burrows, Michael C. Chaisson, Michael C. Chinwalla, Michael C. Dabholkar, Michael C. Dugan, Michael C. Evans, Michael C. Fitch, Michael C. Garroway, Michael C. Green, Michael C. Haines, Michael C. Hargrave, Michael C. Heger, Michael C. Hedges, Michael C. Hwang, Michael C. Jansson, Michael C. Johnson, Michael C. Ketchum, Michael C. Lai, Michael C. Lee, Michael C. Levy, Michael C. Muzny, Michael C. Nelson, Michael C. O'Connor, Michael C. O'Neil, Michael C. Pevsner, Michael C. Redman, Michael C. Rokhsar, Michael C. Salzberg, Michael C. Schuster, Michael C. Sherry, Michael C. Stoye, Michael C. Tector, Michael C. Thomas, Michael C. Venter, Michael C. Wilson, Michael C. Young, Michael C. Zoghbi

DOI: 10.1126/science.1081920 — 10 APRIL 2002

View all references

Complete genomic and epigenetic maps of human centromeres

International Human Genome Sequencing Consortium, Nils G. Autio, Christopher L. Brinkman, Michael C. Burrows, Michael C. Chaisson, Michael C. Chinwalla, Michael C. Dabholkar, Michael C. Dugan, Michael C. Evans, Michael C. Fitch, Michael C. Garroway, Michael C. Green, Michael C. Haines, Michael C. Hargrave, Michael C. Heger, Michael C. Hwang, Michael C. Jansson, Michael C. Johnson, Michael C. Ketchum, Michael C. Lai, Michael C. Lee, Michael C. Levy, Michael C. Muzny, Michael C. Nelson, Michael C. O'Connor, Michael C. O'Neil, Michael C. Pevsner, Michael C. Redman, Michael C. Rokhsar, Michael C. Salzberg, Michael C. Schuster, Michael C. Sherry, Michael C. Stoye, Michael C. Tector, Michael C. Thomas, Michael C. Venter, Michael C. Wilson, Michael C. Young, Michael C. Zoghbi

DOI: 10.1126/science.1081921 — 10 APRIL 2002

View all references

From telomere to telomere: The transcriptional and epigenetic state of human repeat elements

International Human Genome Sequencing Consortium, Nils G. Autio, Christopher L. Brinkman, Michael C. Burrows, Michael C. Chaisson, Michael C. Chinwalla, Michael C. Dabholkar, Michael C. Dugan, Michael C. Evans, Michael C. Fitch, Michael C. Garroway, Michael C. Green, Michael C. Haines, Michael C. Hargrave, Michael C. Heger, Michael C. Hwang, Michael C. Jansson, Michael C. Johnson, Michael C. Ketchum, Michael C. Lai, Michael C. Lee, Michael C. Levy, Michael C. Muzny, Michael C. Nelson, Michael C. O'Connor, Michael C. O'Neil, Michael C. Pevsner, Michael C. Redman, Michael C. Rokhsar, Michael C. Salzberg, Michael C. Schuster, Michael C. Sherry, Michael C. Stoye, Michael C. Tector, Michael C. Thomas, Michael C. Venter, Michael C. Wilson, Michael C. Young, Michael C. Zoghbi

DOI: 10.1126/science.1081922 — 10 APRIL 2002

View all references

A complete reference genome improves analysis of human genetic variation

International Human Genome Sequencing Consortium, Nils G. Autio, Christopher L. Brinkman, Michael C. Burrows, Michael C. Chaisson, Michael C. Chinwalla, Michael C. Dabholkar, Michael C. Dugan, Michael C. Evans, Michael C. Fitch, Michael C. Garroway, Michael C. Green, Michael C. Haines, Michael C. Hargrave, Michael C. Heger, Michael C. Hwang, Michael C. Jansson, Michael C. Johnson, Michael C. Ketchum, Michael C. Lai, Michael C. Lee, Michael C. Levy, Michael C. Muzny, Michael C. Nelson, Michael C. O'Connor, Michael C. O'Neil, Michael C. Pevsner, Michael C. Redman, Michael C. Rokhsar, Michael C. Salzberg, Michael C. Schuster, Michael C. Sherry, Michael C. Stoye, Michael C. Tector, Michael C. Thomas, Michael C. Venter, Michael C. Wilson, Michael C. Young, Michael C. Zoghbi

DOI: 10.1126/science.1081923 — 10 APRIL 2002

View all references

Epigenetic patterns in a complete human genome

International Human Genome Sequencing Consortium, Nils G. Autio, Christopher L. Brinkman, Michael C. Burrows, Michael C. Chaisson, Michael C. Chinwalla, Michael C. Dabholkar, Michael C. Dugan, Michael C. Evans, Michael C. Fitch, Michael C. Garroway, Michael C. Green, Michael C. Haines, Michael C. Hargrave, Michael C. Heger, Michael C. Hwang, Michael C. Jansson, Michael C. Johnson, Michael C. Ketchum, Michael C. Lai, Michael C. Lee, Michael C. Levy, Michael C. Muzny, Michael C. Nelson, Michael C. O'Connor, Michael C. O'Neil, Michael C. Pevsner, Michael C. Redman, Michael C. Rokhsar, Michael C. Salzberg, Michael C. Schuster, Michael C. Sherry, Michael C. Stoye, Michael C. Tector, Michael C. Thomas, Michael C. Venter, Michael C. Wilson, Michael C. Young, Michael C. Zoghbi

DOI: 10.1126/science.1081924 — 10 APRIL 2002

View all references

The complete sequence of a human genome

International Human Genome Sequencing Consortium, Nils G. Autio, Christopher L. Brinkman, Michael C. Burrows, Michael C. Chaisson, Michael C. Chinwalla, Michael C. Dabholkar, Michael C. Dugan, Michael C. Evans, Michael C. Fitch, Michael C. Garroway, Michael C. Green, Michael C. Haines, Michael C. Hargrave, Michael C. Heger, Michael C. Hwang, Michael C. Jansson, Michael C. Johnson, Michael C. Ketchum, Michael C. Lai, Michael C. Lee, Michael C. Levy, Michael C. Muzny, Michael C. Nelson, Michael C. O'Connor, Michael C. O'Neil, Michael C. Pevsner, Michael C. Redman, Michael C. Rokhsar, Michael C. Salzberg, Michael C. Schuster, Michael C. Sherry, Michael C. Stoye, Michael C. Tector, Michael C. Thomas, Michael C. Venter, Michael C. Wilson, Michael C. Young, Michael C. Zoghbi

DOI: 10.1126/science.1081925 — 10 APRIL 2002

View all references









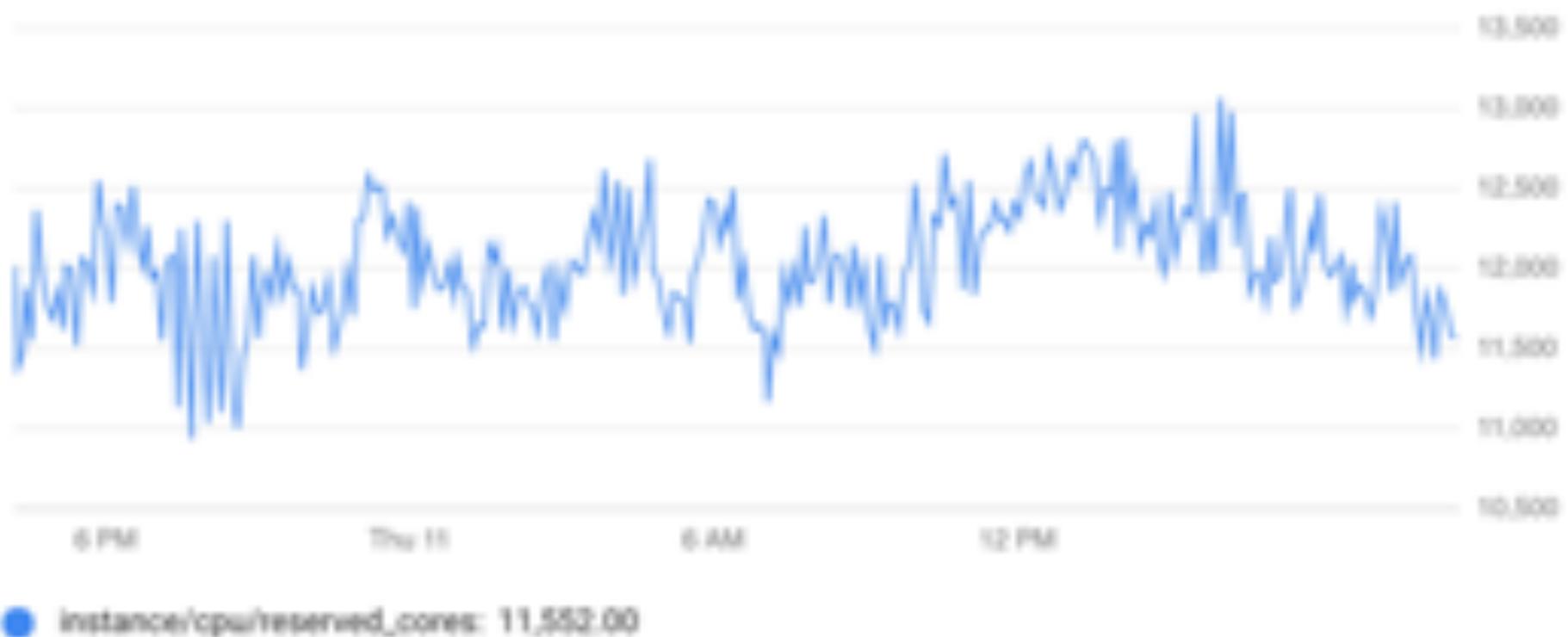

Core usage over 24 hours

Preview

1 hour

4 hours

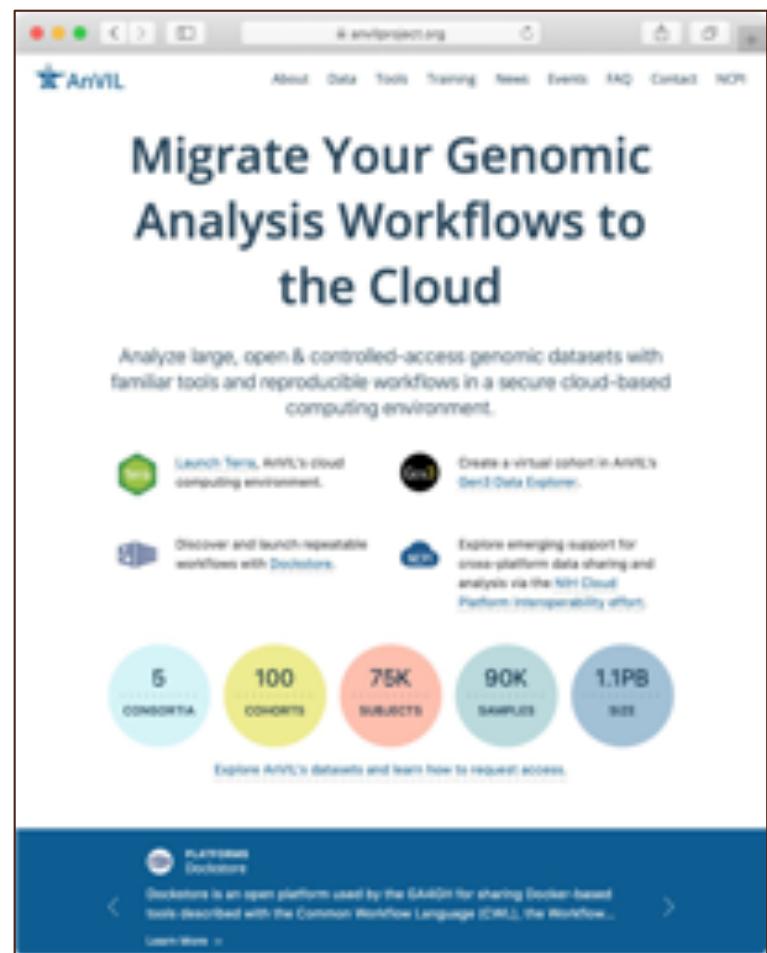
1 day



What is the AnVIL?

Scalable and interoperable computing resource for the genomics scientific community

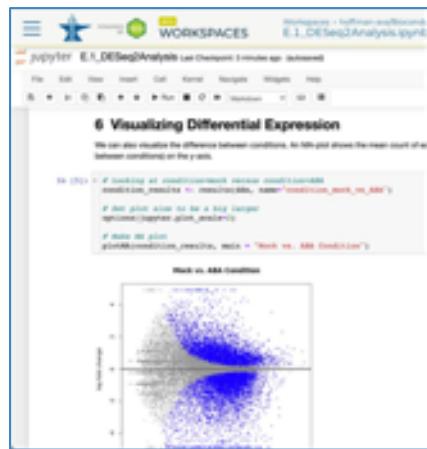
- **Cloud-based infrastructure**
 - Highly elastic; shared analysis and computing environment
- **Data access and security**
 - Genomic datasets, phenotypes and metadata
 - Large datasets generated by NHGRI programs, as well as other initiatives / agencies
 - dbGaP Authenticated sharing of primary and derived datasets
- **Collaborative computing environment for datasets and analysis workflows**
 - Storage, scalable analytics, data visualization
 - Security, training & outreach, with new models of data access
 - ...for both users with limited computational expertise and sophisticated data scientist users



<https://anvilproject.org>

Schatz, Philippakis et al. (2022) *Cell Genomics*
doi: <https://doi.org/10.1016/j.xgen.2021.100085>

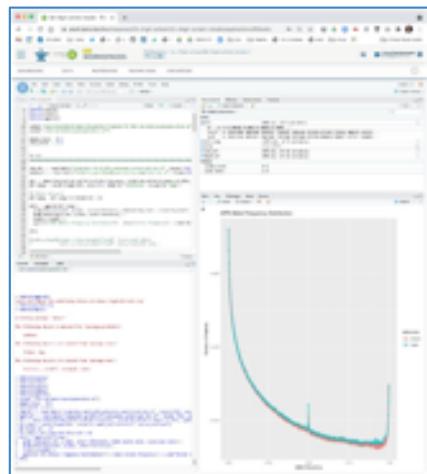
AnVIL Analysis Platforms



- + Code, text and plots in one document
- + Supports coding in Python or R
- Least scalable, not a complete IDE



- + Graphical interface for thousands of tools and workflows
- + Highly accessible and reproducible
- Tools must be preconfigured to use



- + Feature rich IDE for programming in R
- + Rich statistics, ML, and visualizations
- Limited support for other programming languages



- + Extremely scalable and flexible
- Most technically demanding
- Unpredictable and potentially large costs

My perspective



- + Code, text and plots in one document
- + Supports coding in Python or R
- Least scalable, not a complete IDE



- + Graphical interface for thousands of tools and workflows
- + Highly accessible and reproducible
- Tools must be preconfigured to use



- + Feature rich IDE for programming in R
- + Rich statistics, ML, and visualizations
- Limited support for other programming languages



- + Extremely scalable and flexible
- Most technically demanding
- Unpredictable and potentially large costs

Parallel Algorithm Spectrum

Embarrassingly Parallel



Each item is Independent

Loosely Coupled



Independent-Sync-Independent

Tightly Coupled



Constant Sync

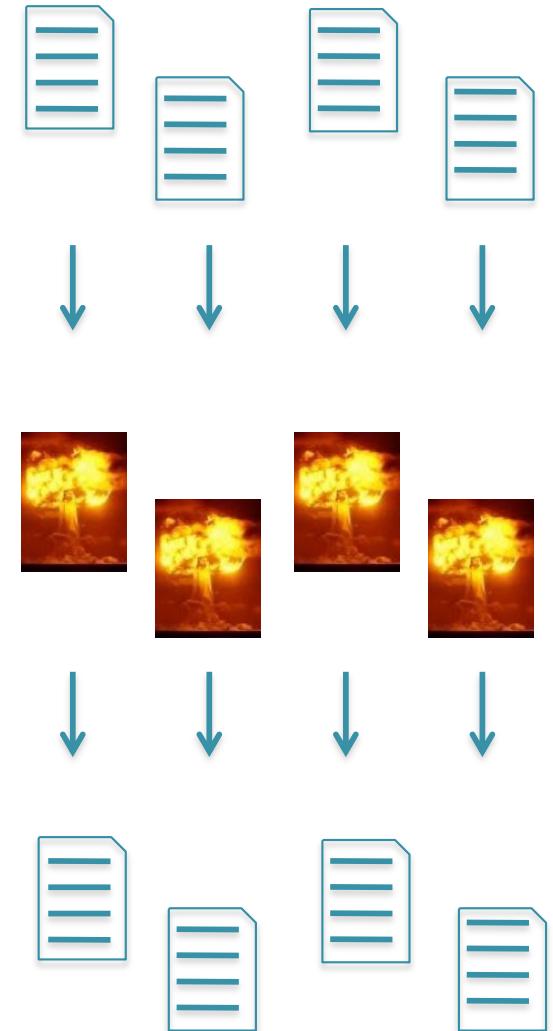
Embarrassingly Parallel

aka Elementary School Dance



Embarrassingly Parallel

- Batch computing
 - Each item is independent
 - Split input into many chunks
 - Process each chunk separately on a different computer
- Challenges
 - Distributing work, load balancing, monitoring & restart
- Technologies
 - Condor, Sun Grid Engine, WDL
 - Amazon Simple Queue



3202 Genomes to go...



Samantha Zarate



bioinformatics II
output 4:
File: unaligned.sam
File: coordinates
File: FastaRef.fa
String: suspirium

bioinformatics II
output 5:
bamfile: shasta-10-201907051317.bam
--reference --targetmate2
-d 7000000
-n 10000000000
-r 10000000000
and
bioinformatics II
output 6:
File: shasta-10-201907051317.bam
File: shasta-10-201907051317.bcf
bioinformatics II
output 7:
bioformat: "bam2vcf.shasta-10-201907051317.bcf"
bioformat: "bcftools"
bioformat: "bcftools vcffilter -d 10000000000 -O vcf -o shasta-10-201907051317.vcf"
bioformat: "bcftools index shasta-10-201907051317.vcf"
bioformat: "bcftools stats -t=example101,example102,shasta-10-201907051317.vcf"



```

graph TD
    subgraph "SayItTwice"
        direction TB
        A["SayItTwice.input"] --> B[WriteGreeting]
        B --> C[ReadGreeting]
        C --> D["SayItTwice.outfile"]
    end

```

```

graph TD
    subgraph "SayItTwice"
        direction TB
        A["SayItTwice.input"] --> B[WriteGreeting]
        B --> C[ReadGreeting]
        C --> D["SayItTwice.outfile"]
    end

```

SayItTwice.input

WriteGreeting

ReadGreeting

SayItTwice.outfile

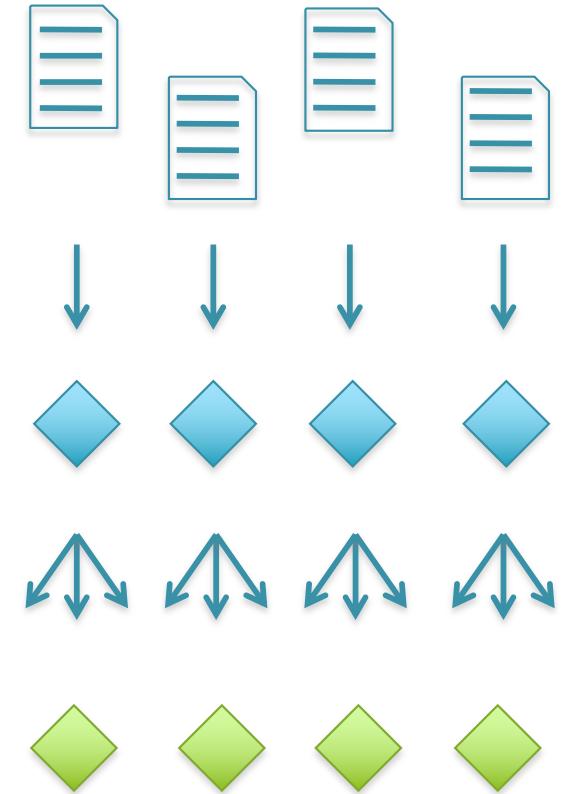
Loosley Coupled

aka the High School Dance



Loosely Coupled

- Divide and conquer
 - Independently process many items
 - Group partial results
 - Scan partial results into final answer



- Challenges
 - Batch computing challenges
 - + Shuffling of huge datasets
- Technologies
 - Hadoop, MapReduce, Dryad, WDL
 - Parallel Databases

CloudBurst



1. Map: Catalog K-mers

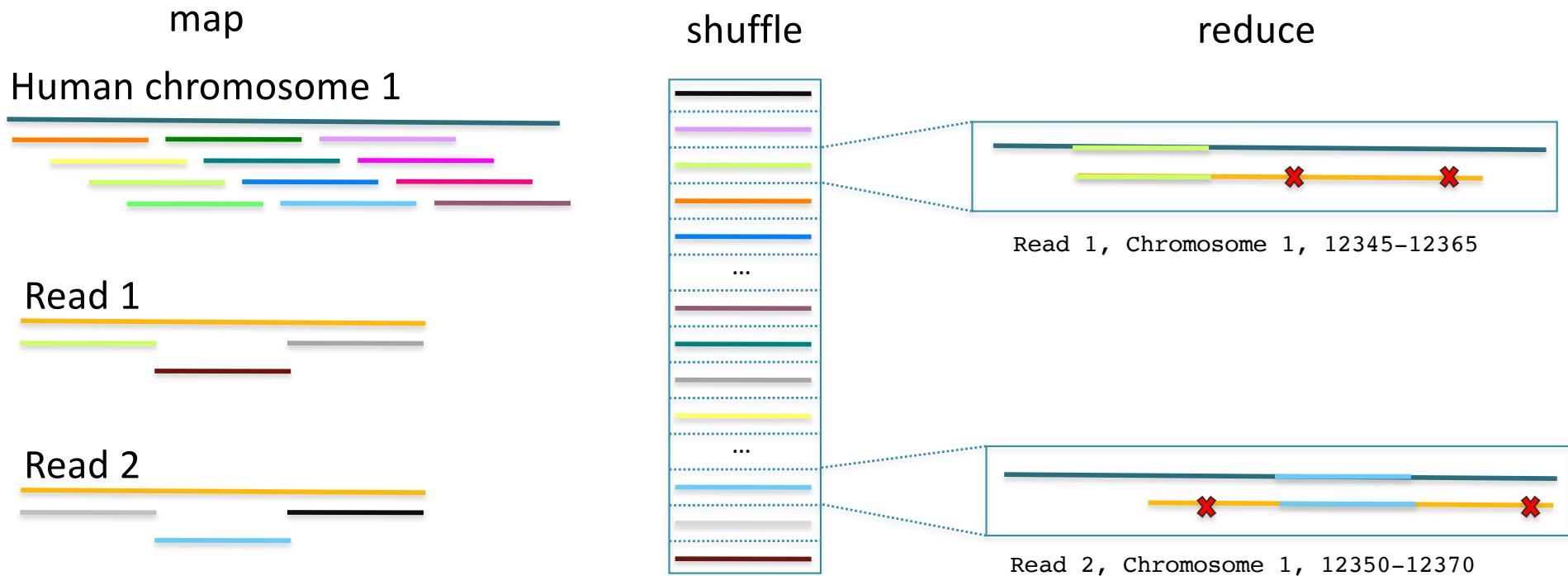
- Emit k-mers in the genome and reads

2. Shuffle: Collect Seeds

- Conceptually build a hash table of k-mers and their occurrences

3. Reduce: End-to-end alignment

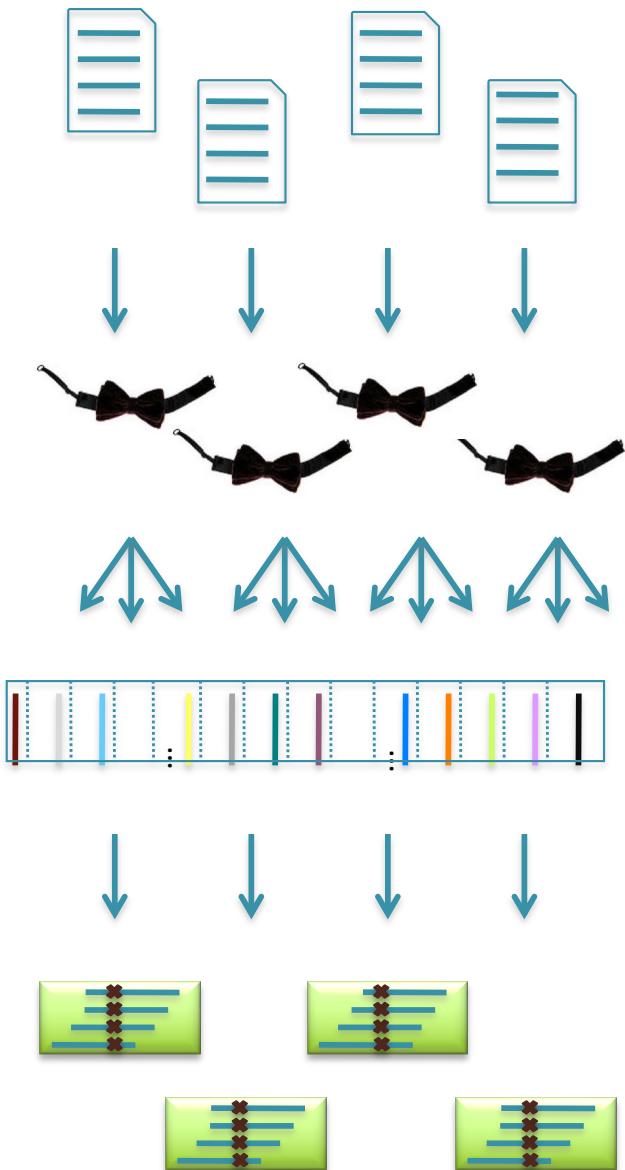
- If read aligns end-to-end with $\leq k$ errors, record the alignment



Crossbow

Align billions of reads and find SNPs

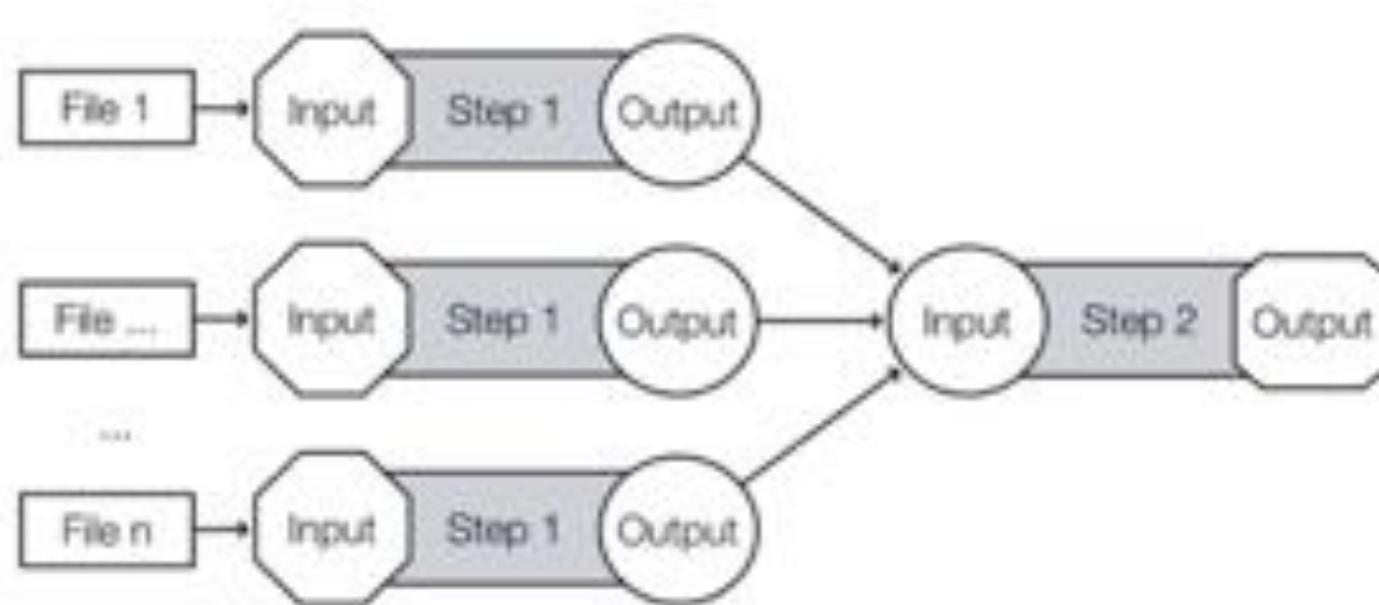
- Map: Bowtie (*Langmead et al., 2009*)
 - Find best alignment for each read
 - Emit (chromosome region, alignment)
- Shuffle: Hadoop
 - Group and sort alignments by region
- Reduce: SOAPsnp (*Li et al., 2009*)
 - Scan alignments for divergent columns
 - Accounts for sequencing error, known SNPs



Searching for SNPs with Cloud Computing.

Langmead B, Schatz MC, Lin J, Pop M, Salzberg SL (2009) *Genome Biology*. **10**:R134

Scatter-Gather



```
17 lines (in size) 288 bytes
```

```
1 version 1.4
2
3 my@lunar:/allenbond$ ./scatter()
4     scatter() is complete!
5         all scattering
6
7
8
9
10 use WriteString();
11 command {
12     write("Hello Scatterd world!");
13 }
14 output {
15     file output_greeting = stdout();
16 }
```

```
16 lines (in size) 164 bytes
```

```
1 version 1.4
2
3 my@lunar:/allenbond$ ./scatter()
4     scatter() is complete!
5         all scattering
6
7
8
9
10 use WriteString();
11 command {
12     write("Hello Scatterd world!");
13 }
14 output {
15     file output_greeting = stdout();
16 }
```

Tightly Coupled

aka the Ballroom Dance



Tightly Coupled

- Computation that cannot be partitioned
 - Graph Analysis
 - Molecular Dynamics
 - Population simulations
- Challenges
 - Loosely coupled challenges
 - + Parallel algorithms design
- Technologies
 - MPI
 - MapReduce, Dryad, Pregel

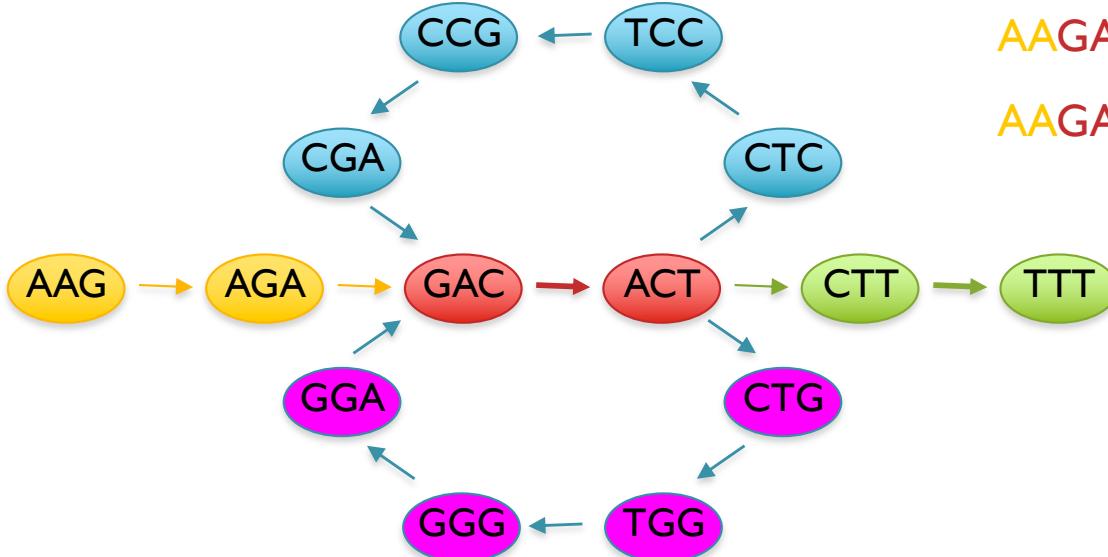


Parallel Genome Assembly

Reads

AAGA
ACTT
ACTC
ACTG
AGAG
CCGA
CGAC
CTCC
CTGG
CTTT
...

de Bruijn Graph



Potential Genomes

AAGACTCCGACTGGGACTTT
AAGACTGGGACTCCGACTTT

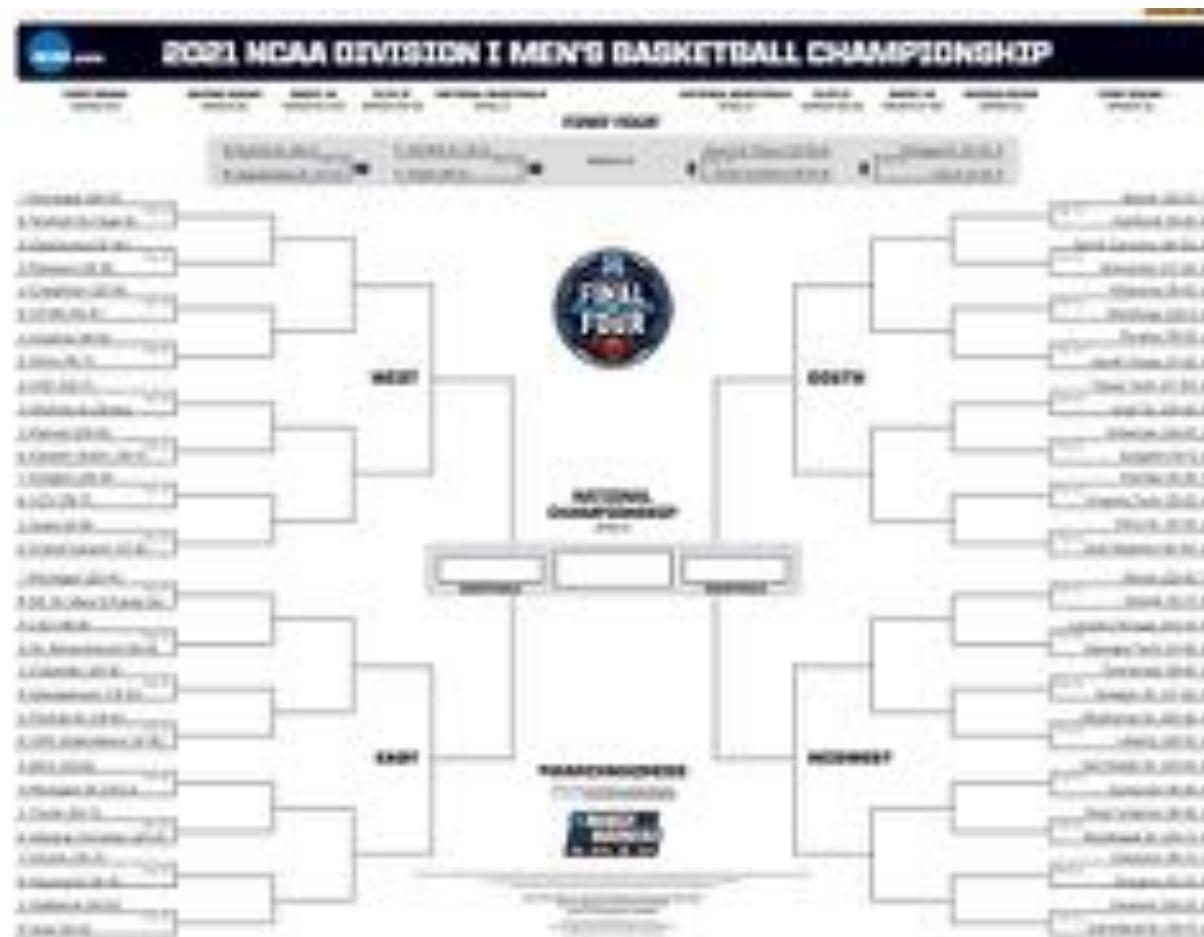
The de Bruijn graph for a human genome requires billions of nodes and edges, which may not fit on a single computer

- Assume there is only one node per computer, but each computer can send messages to its neighbors in the graph
- How can each node decide what to do to collapse the graph?

Warmup Exercise

Who here was born closest to Sept 21?

- You can only compare to 1 other person at a time

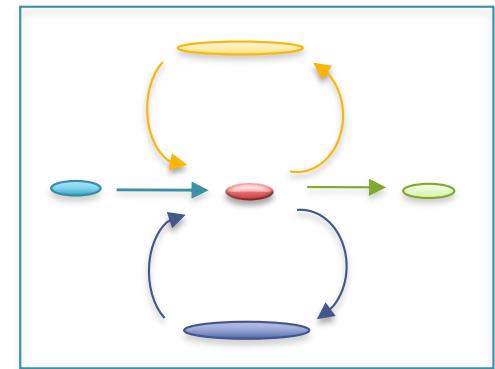
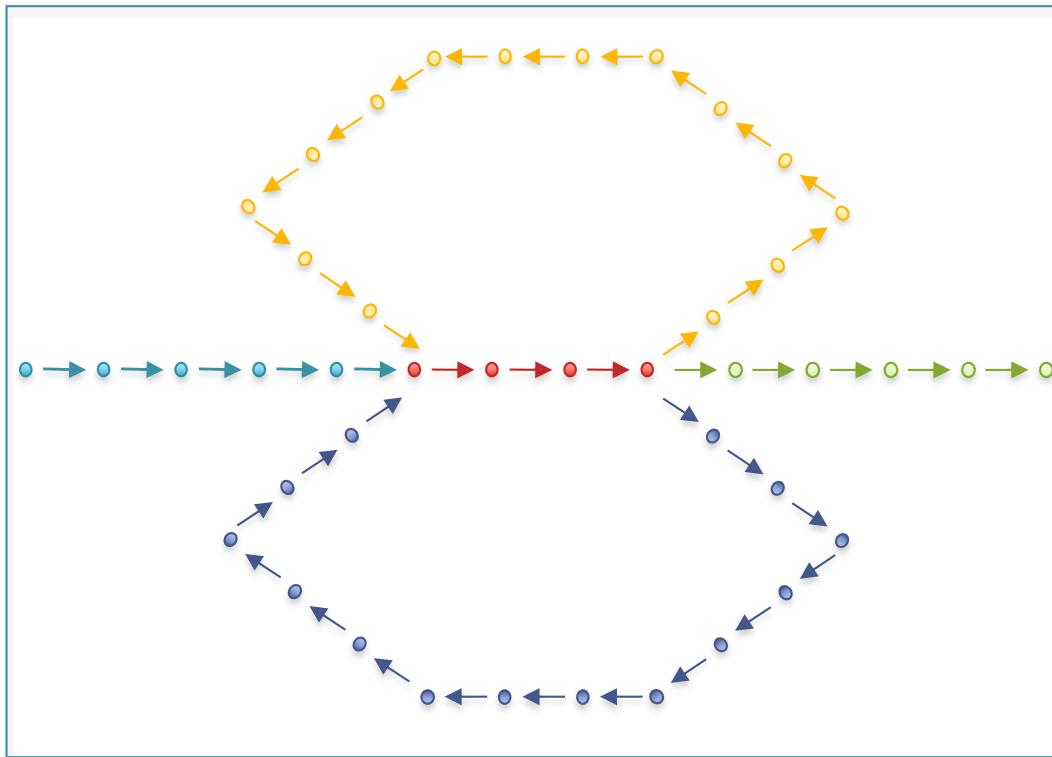


Find winner among 64 teams in just 6 rounds

$$\text{Why } 6? \log_2(64) = 6 \text{ (aka } 2^6 = 64\text{)}$$

Parallel Graph Compression

- After construction, many edges are unambiguous
 - Merge together compressible nodes
 - Graph physically distributed over hundreds of computers



Design Patterns for Efficient Graph Algorithms in MapReduce.

Lin, J., Schatz, M.C. (2010) Workshop on Mining and Learning with Graphs Workshop (KDD-2010)

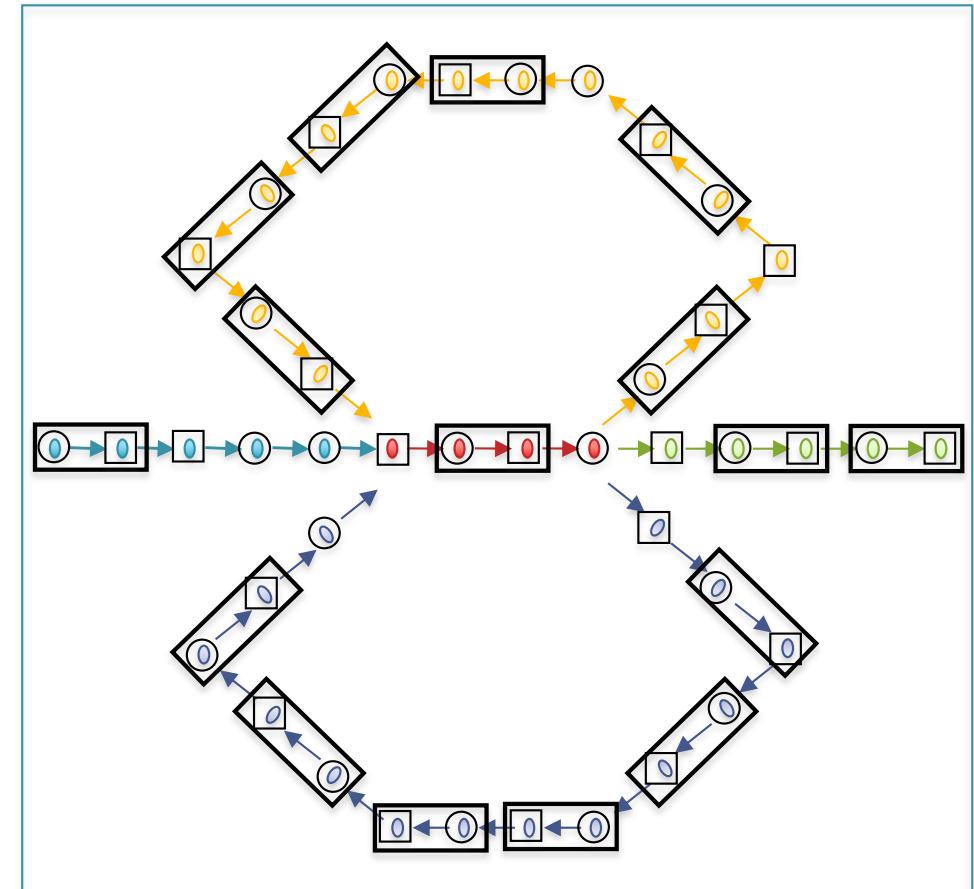
Fast Path Compression

Challenges

- Nodes stored on different computers
- Nodes can only access direct neighbors

Randomized List Ranking

- Randomly assign H / T to each compressible node
- Compress $H \rightarrow T$ links



Initial Graph: 42 nodes

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) ACM Symposium on Theory of Computation. 230-239.

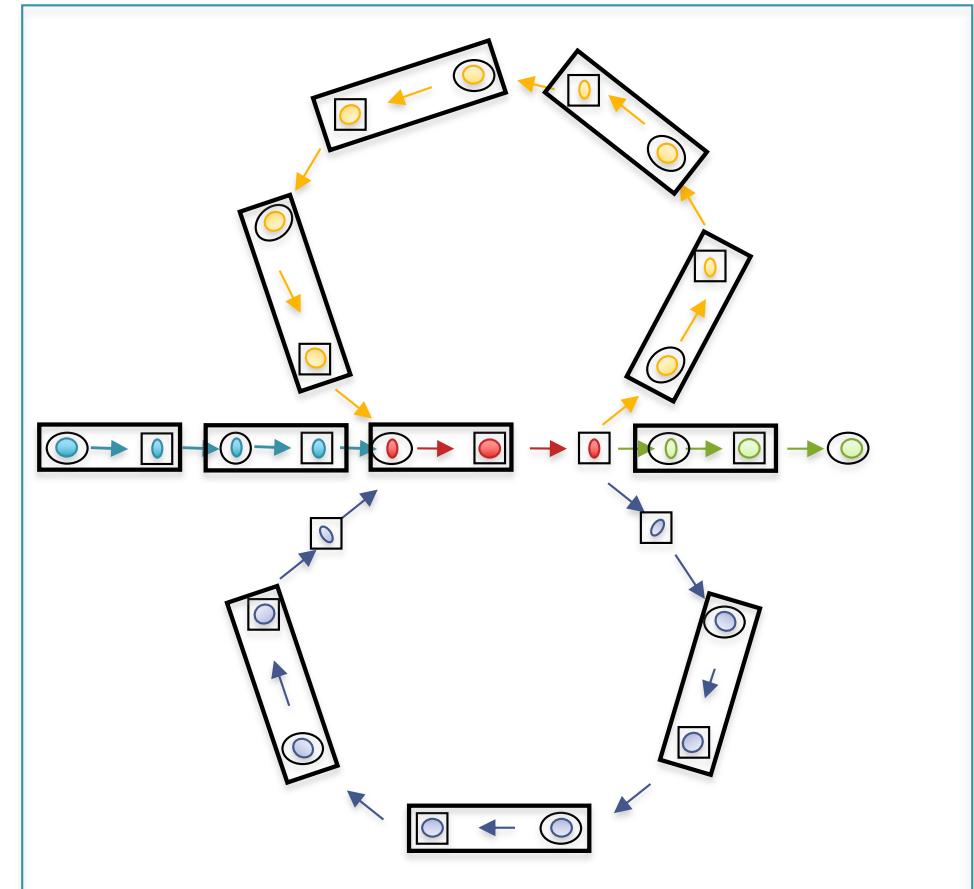
Fast Path Compression

Challenges

- Nodes stored on different computers
- Nodes can only access direct neighbors

Randomized List Ranking

- Randomly assign H / T to each compressible node
- Compress $H \rightarrow T$ links



Round 1: 26 nodes (38% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) ACM Symposium on Theory of Computation. 230-239.

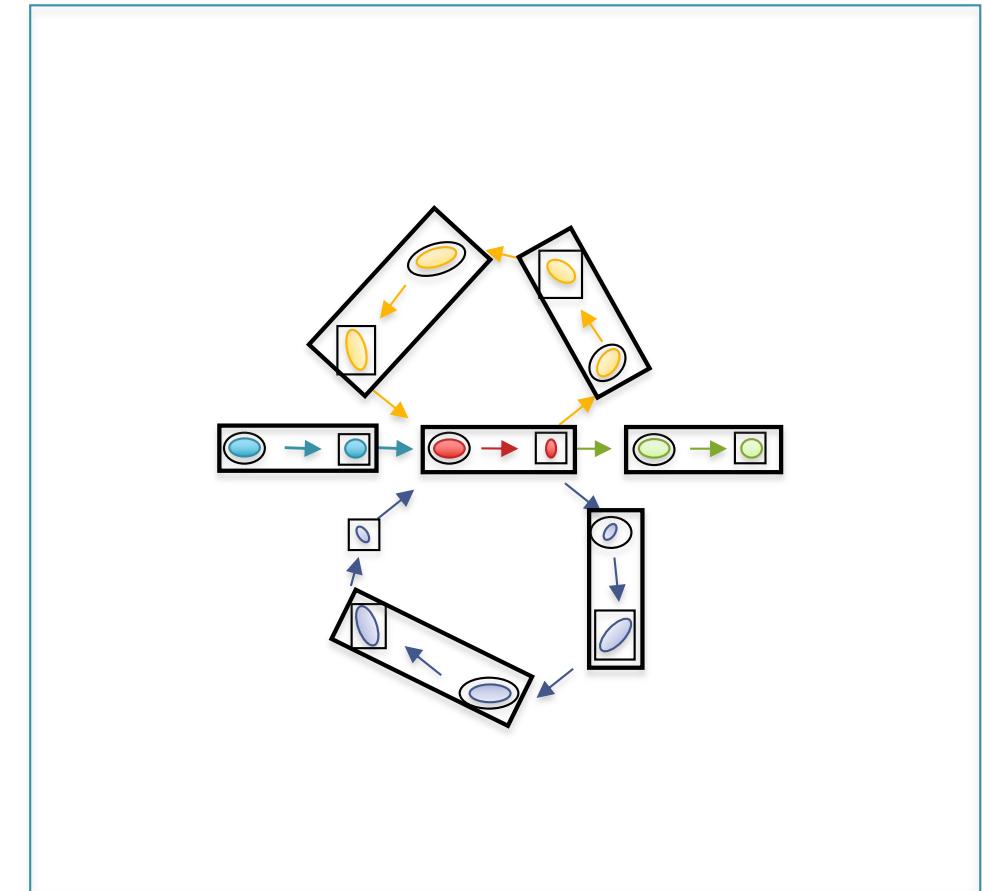
Fast Path Compression

Challenges

- Nodes stored on different computers
- Nodes can only access direct neighbors

Randomized List Ranking

- Randomly assign H/T to each compressible node
- Compress $\text{H} \rightarrow \text{T}$ links



Round 2: 15 nodes (64% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) ACM Symposium on Theory of Computation. 230-239.

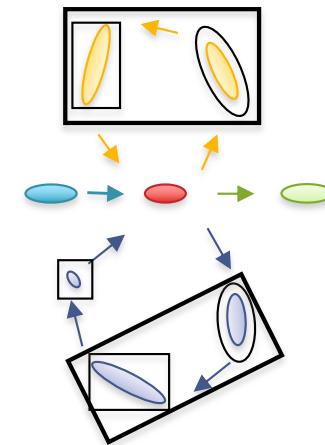
Fast Path Compression

Challenges

- Nodes stored on different computers
- Nodes can only access direct neighbors

Randomized List Ranking

- Randomly assign \textcircled{H} / \boxed{T} to each compressible node
- Compress $\textcircled{H} \rightarrow \boxed{T}$ links



Round 2: 8 nodes (81% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) ACM Symposium on Theory of Computation. 230-239.

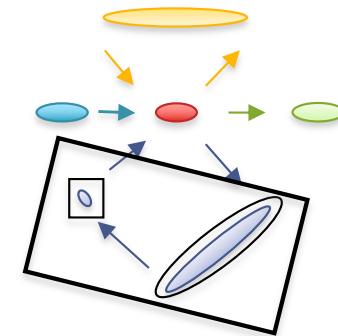
Fast Path Compression

Challenges

- Nodes stored on different computers
- Nodes can only access direct neighbors

Randomized List Ranking

- Randomly assign H/T to each compressible node
- Compress $\text{H} \rightarrow \text{T}$ links



Round 3: 6 nodes (86% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) ACM Symposium on Theory of Computation. 230-239.

Fast Path Compression

Challenges

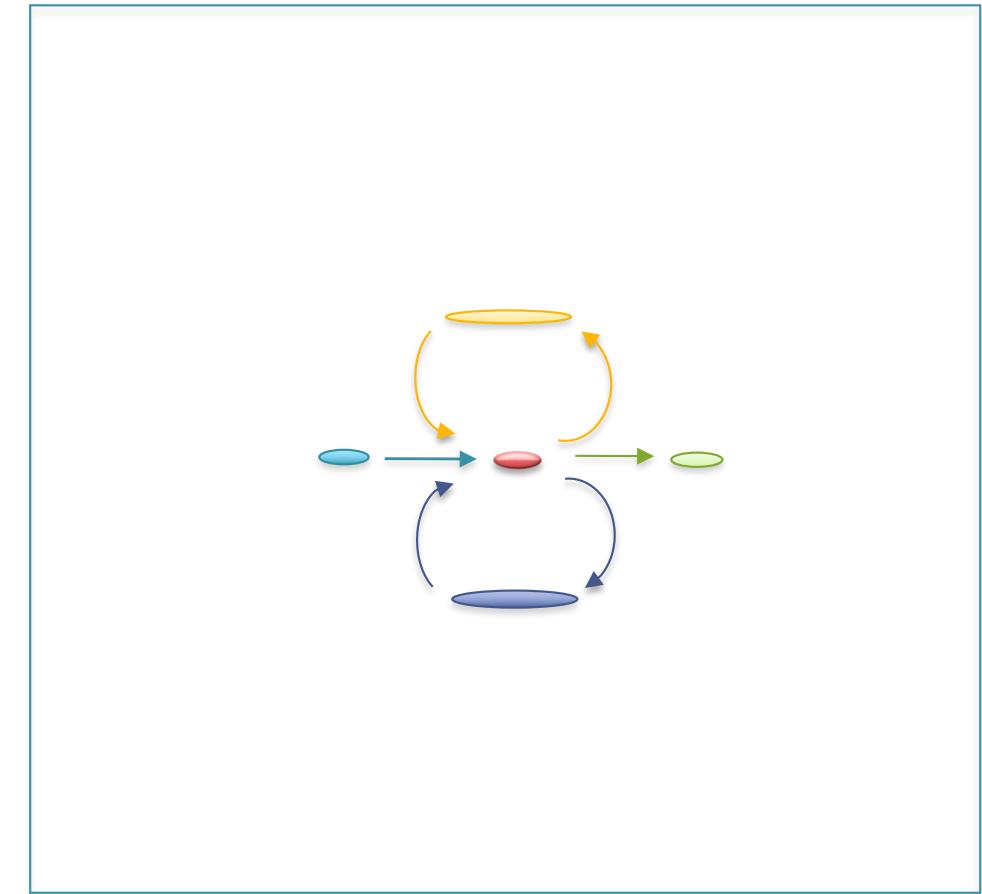
- Nodes stored on different computers
- Nodes can only access direct neighbors

Randomized List Ranking

- Randomly assign H/T to each compressible node
- Compress $\text{H} \rightarrow \text{T}$ links

Performance

- Compress all chains in $\log(S)$ rounds



Round 4: 5 nodes (88% savings)

Randomized Speed-ups in Parallel Computation.

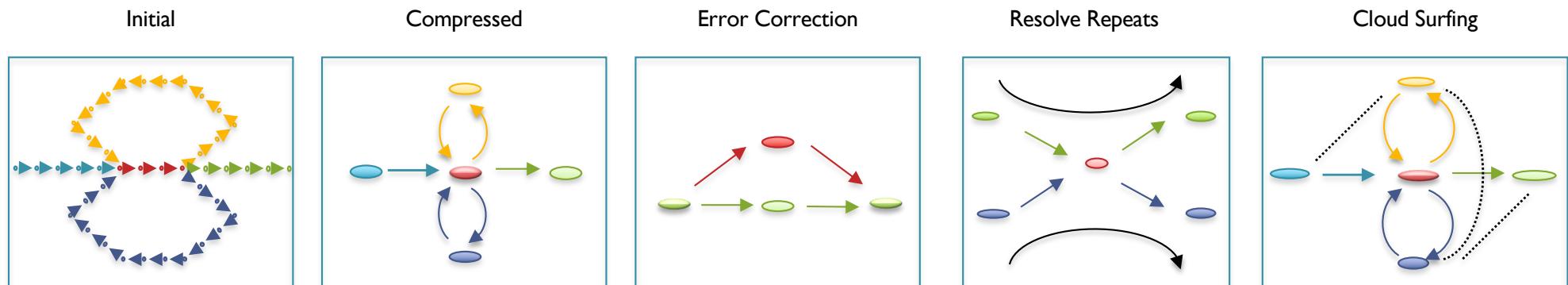
Vishkin U. (1984) ACM Symposium on Theory of Computation. 230-239.

Contrail



De novo Assembly of the Human Genome

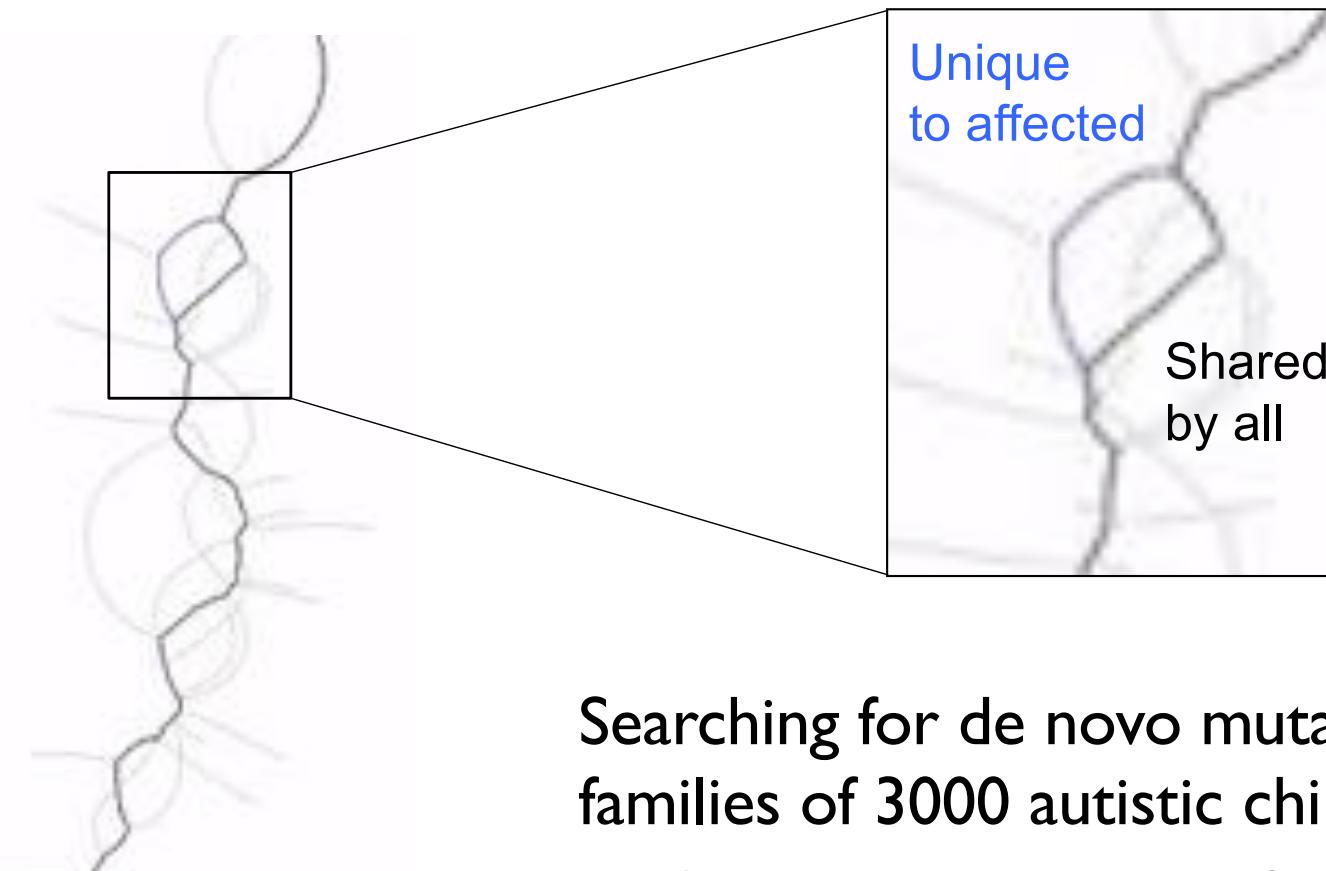
- *Genome:* African male NA18507 (SRA000271, Bentley et al., 2008)
- *Input:* 3.5B 36bp reads, 210bp insert (~40x coverage)



N	>7 B	>1 B	4.2 M	4.1 M	3.3 M
Max	27 bp	303 bp	20,594 bp	20,594 bp	20,594 bp
N50	27 bp	< 100 bp	995 bp	1,050 bp	1,427 bp

Assembly of Large Genomes with Cloud Computing.
Schatz MC (2010) PhD Thesis. University of Maryland

De novo mutations and de Bruijn Graphs



COLEC12
C->A

Searching for de novo mutations in the families of 3000 autistic children.

- Assemble together reads from mom, dad, affected & unaffected children
- Look for sequence paths unique to affected child

The contribution of de novo coding mutations to autism spectrum disorder
Iossifov et al. (2014) Nature doi:10.1038/nature13908



Learning More about AnVIL

The screenshots illustrate the AnVIL Learn portal's interface, which includes a sidebar with navigation links and a main content area with various sections:

- Getting Started with AnVIL:** This section provides an overview of the platform, including account setup, billing, and Terra integration.
- Overview of Galaxy in the Cloud:** This section shows a screenshot of the Galaxy interface running in the cloud.
- What is AnVIL?** This section defines AnVIL as "inverting the model of genomic data science" and compares traditional data storage with AnVIL's approach.

Portal: <http://anvilproject.org/>

Mailing List: help@lists.anvilproject.org

Discourse: <http://help.anvil-project.org>



Learning More about WDL

The image displays three screenshots of web-based resources for learning Workflow Description Language (WDL):

- learn-wdl (https://github.com/openwdl/learn-wdl)**: This page provides an introduction to WDL, including its purpose (specifying data processing workflows), history (originally developed for genomics), and educational materials for learning to read and write WDL scripts. It also outlines "Learn WDL in 3 Steps":
 - 1. Set up the environment
 - 2. Run a local WDL script in Docker
 - 3. Review reference material
- miniwdl (https://github.com/chanzuckerberg/miniwdl)**: This page details the installation of miniwdl, which is a local runner & developer toolkit for Python 3.6+. It includes instructions for installing via pip or conda, and verifying the installation. It also shows how to run a WDL script using Docker and provides a diagram illustrating the relationship between inputs and outputs.
- Dockstore search results**: A screenshot of the Dockstore interface showing a list of WDL tools. One tool, "Workflow Language v.0.0", is highlighted with a green checkmark. The results include various WDL tools categorized by type (e.g., Bioinformatics, Genomics, Proteomics) and provider (e.g., BioRxiv, GitHub, Zenodo).

<https://github.com/openwdl/learn-wdl/>

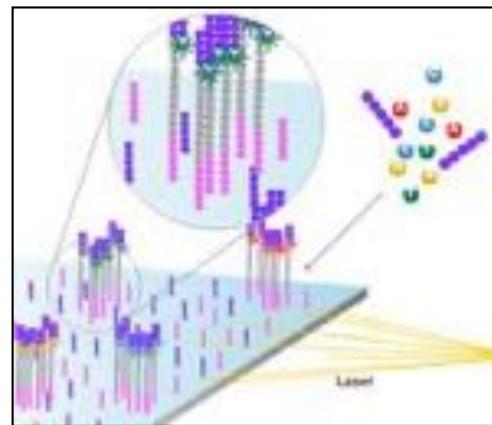
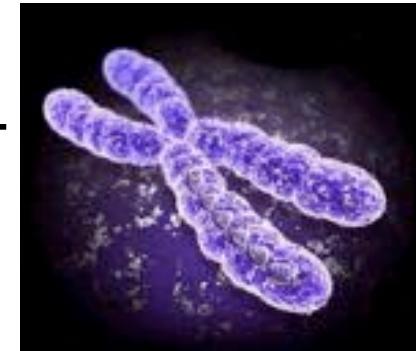
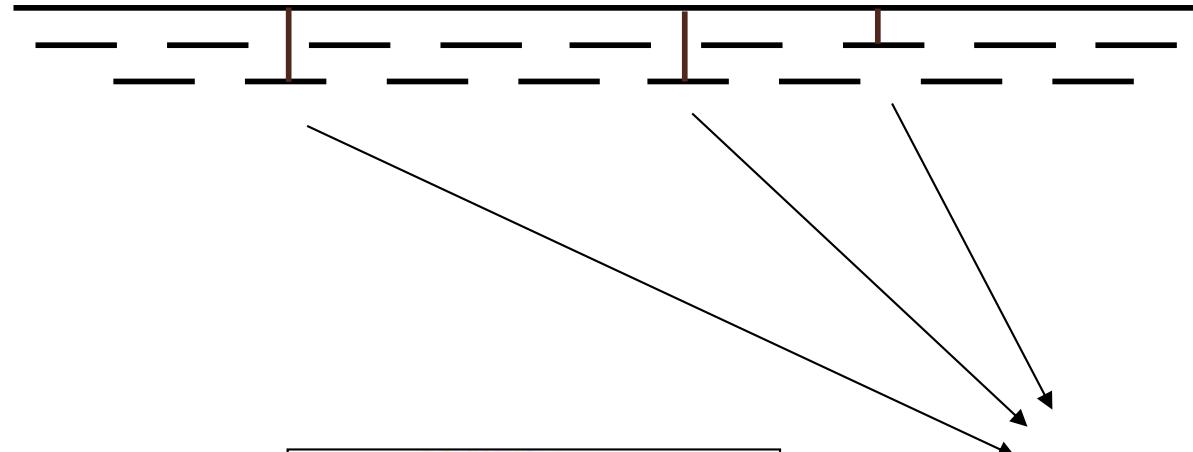
<https://github.com/chanzuckerberg/miniwdl/>

<https://dockstore.org/>

Read Mapping

Personal Genomics

How does your genome compare to the reference?

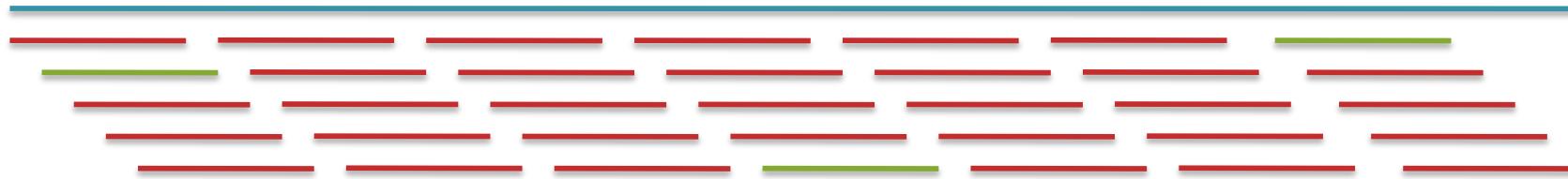


Heart Disease

Cancer

Presidential smile

Brute Force Analysis



- Brute Force:
 - At every possible offset in the genome:
 - Do all of the characters of the query match?
- Analysis
 - Simple, easy to understand
 - Genome length = n [3B]
 - Query length = m [7]
 - Comparisons: $(n-m+1) * m$ [21B]
- Overall runtime: $O(nm)$
 - [How long would it take if we double the genome size, read length?]
 - [How long would it take if we double both?]

Brute Force Reflections

Why check every position?

- GATTACA can't possibly start at position 15

[WHY?]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...
T	G	A	T	T	A	C	A	G	A	T	T	A	C	C	...
										G A T T A C C A					

- Improve runtime to $O(n + m)$ [3B + 7]
 - If we double both, it just takes twice as long
 - Knuth-Morris-Pratt, 1977
 - Boyer-Moyer, 1977, 1991
- For one-off scans, this is the best we can do (optimal performance)
 - We have to read every character of the genome, and every character of the query
 - For short queries, runtime is dominated by the length of the genome