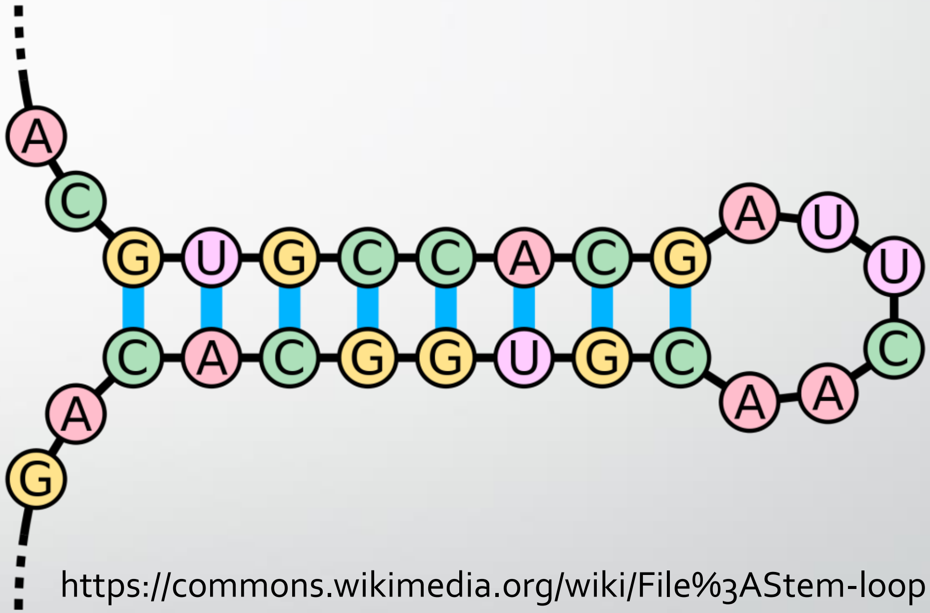
Adventures in Parallelization

On One of the Largest Supercomputers in The Country

Nathan Sloat

RNA Secondary Structure

- RNA
  - “Base” units: nucleotides (nt)
  - Four nucleotides: adenosine, cytosine, guanosine, uracil
- Folding
  - RNA can fold on itself, with nucleotide pairing stabilizing the folds
  - Pairing follows certain rules: Watson-Crick pairs, G-U pairs, and the “turn of three”



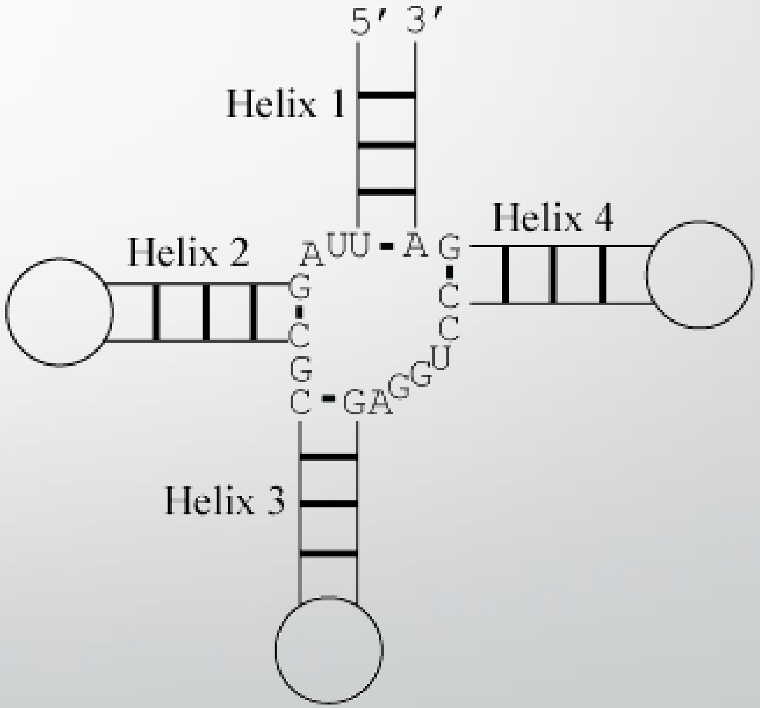
https://commons.wikimedia.org/wiki/File%3AStem-loop.svg

ACGUGCCACGAUUCAACGUGGCACAG

..((((((((......))))))))..

Previous Algorithms

- Wuchty
  - Output depends on the free energy window defined
  - Very slow, and insufficient due to the dependency on thermodynamics
- Crumple
  - Produces exhaustive solution set for given RNA sequence
  - Algorithm inefficient as sequence length increases
- Sliding Windows & Assembly
  - Scans a long sequence in windows, Crumpling each window & saving the best hairpin structure
  - In the end, all of the best structures combine
  - Cannot consider structures with multibranch loops



Multibranch Loop

Swellix

- Calculation for combinatorially complete solution set
- Not simulation, single-solution problem
- Sequence lengths
  - STMV 1058

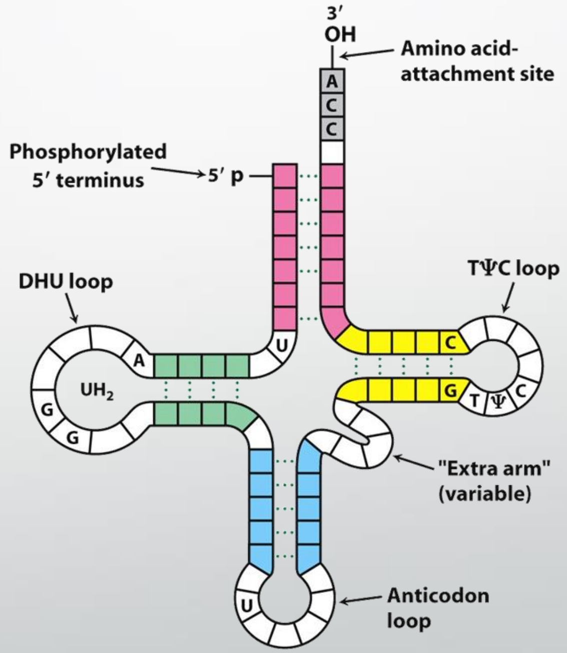
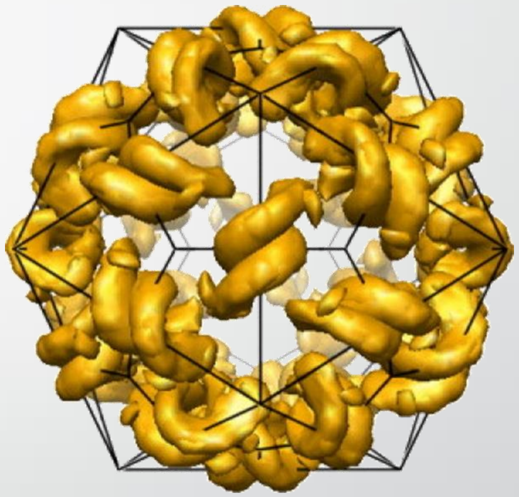


Figure 30.3
Biochemistry, Seventh Edition
© 2012 W. H. Freeman and Company

tRNA



Folded STMV RNA

STMV from VIPERdb
http://viperdb.scripps.edu/info_page.php?VDB=4nia

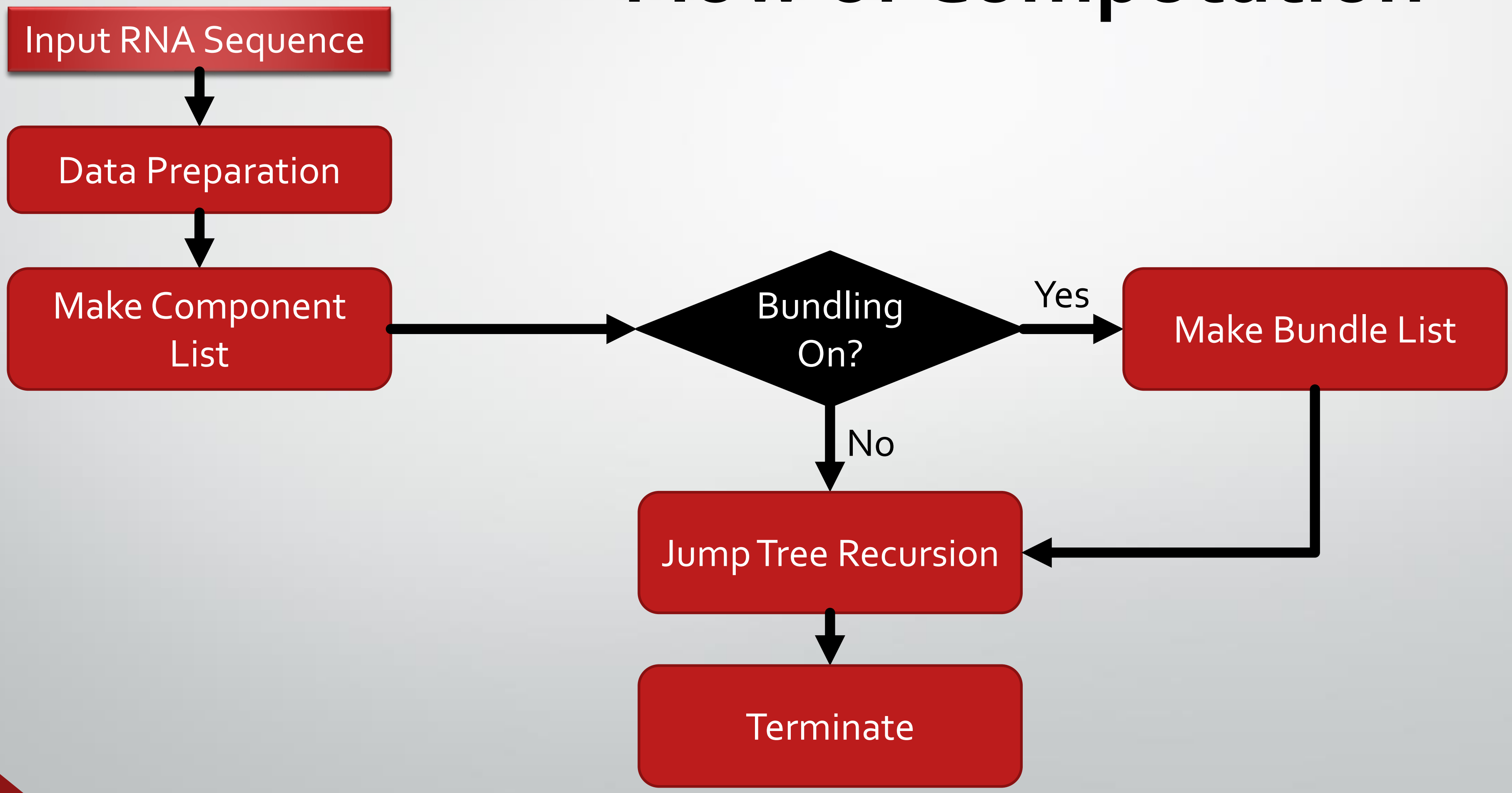
Human Endogenous Retroviral RNA (HERV)

- HIV, cancer, stem cells
- “Junk” DNA: 8% of human genome
- Binds many types of metal ions and proteins
- Likely has many functioning secondary structures
- 141 nt and up

Swellix is an Abstraction of Crumple

- Basic methodology
  - In essence, use the mechanics of Crumple but with respect to helices instead of base pairs
- Additional features
  - Bundling
  - Characterizing output structures with third party libraries
  - Motif searching
- Facilitates Multi-branch Loops

Flow of Computation



Basic Swellix Structure

1. Create Component List

Find all possible individual helices matching given constraints

GCUCUAAAAGAGAG

Min Helix Length 2 GCUCUAAAAGAGAG

.(..((.(........))

1.

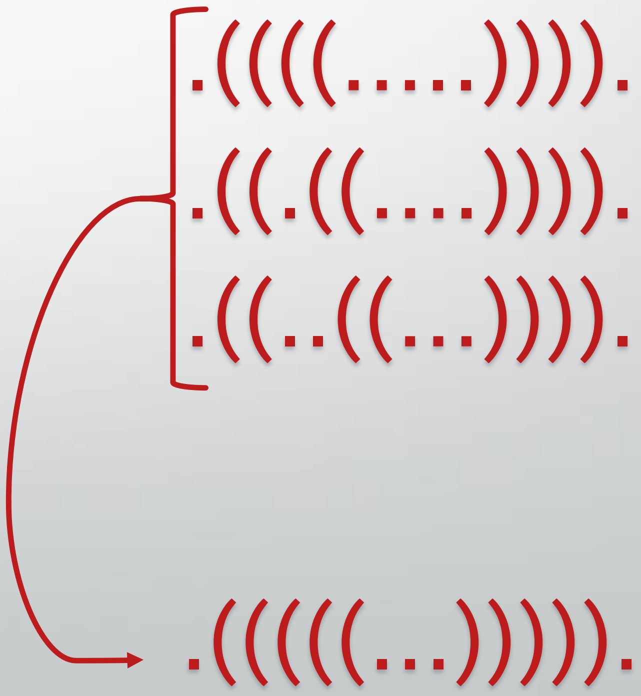
2.

Continue until finished

Basic Swellix Structure

2. Create Bundles

- Biggest difference from Crumple
- Greatly decreases size component list
- Groups similar helices and represents the whole group by one single structure from the bundle



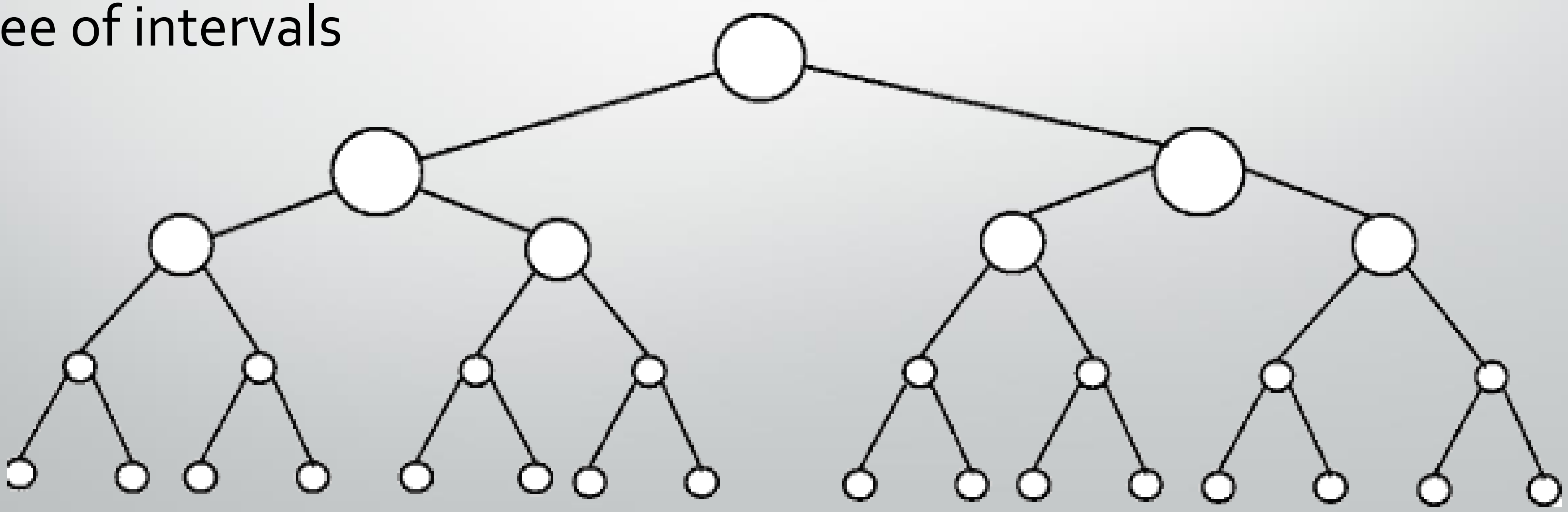
```
.((((.....)))).
.((.((....)))).
.((..((...)))).

.(((((...))))).
```

Basic Swellix Structure

3. Jump Tree Recursion

- Exhausts all possible combinations of components
- Tree of intervals

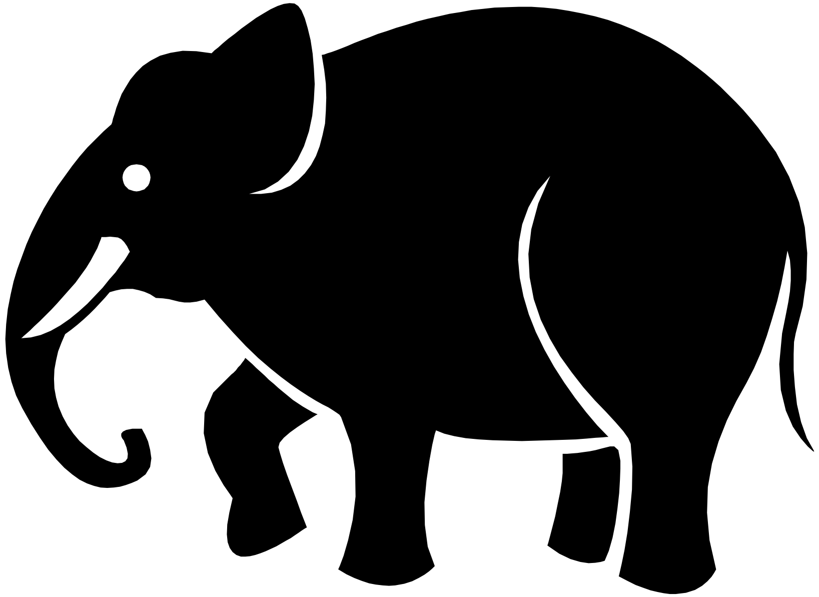


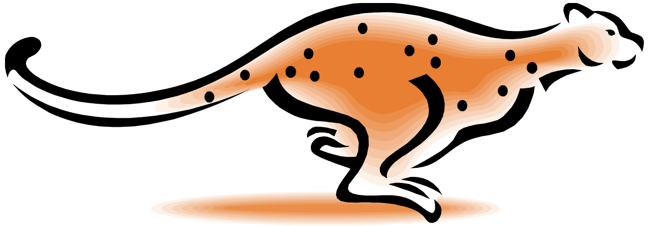
Key Benefits of Swellix

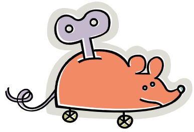
- Offers multiple methods of accelerating previous calculations
  - Bundling
  - “Helper” data structures
- Allows calculations that weren’t previously realistic
- Why supercomputing?

What is Supercomputing About?

Size **Speed**



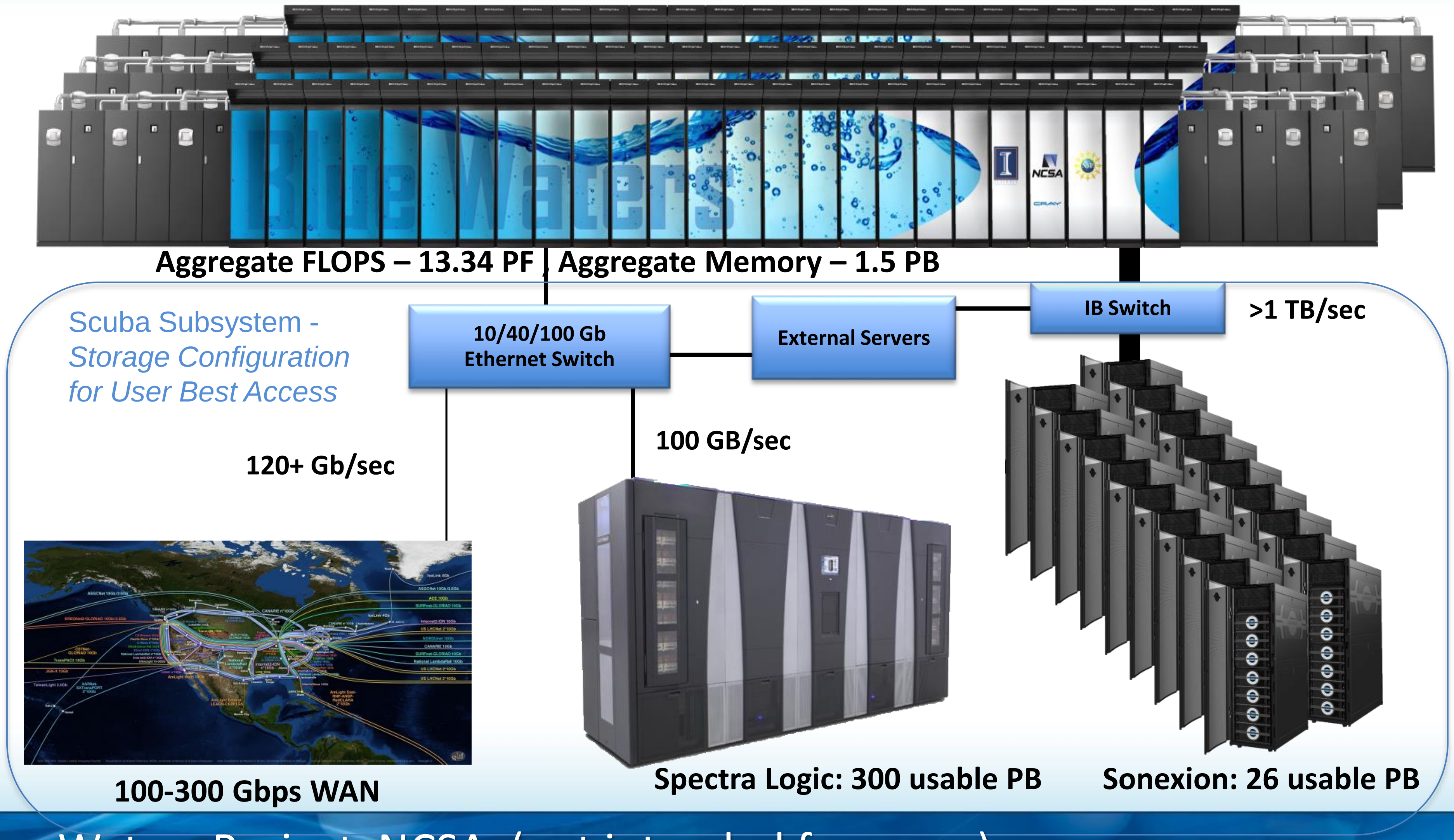




Laptop

Blue Waters

Blue Waters Computing System



Aggregate FLOPS – 13.34 PF | Aggregate Memory – 1.5 PB

Scuba Subsystem - *Storage Configuration for User Best Access*

10/40/100 Gb Ethernet Switch

External Servers

IB Switch

>1 TB/sec

100 GB/sec

120+ Gb/sec

100-300 Gbps WAN

Spectra Logic: 300 usable PB

Sonexion: 26 usable PB

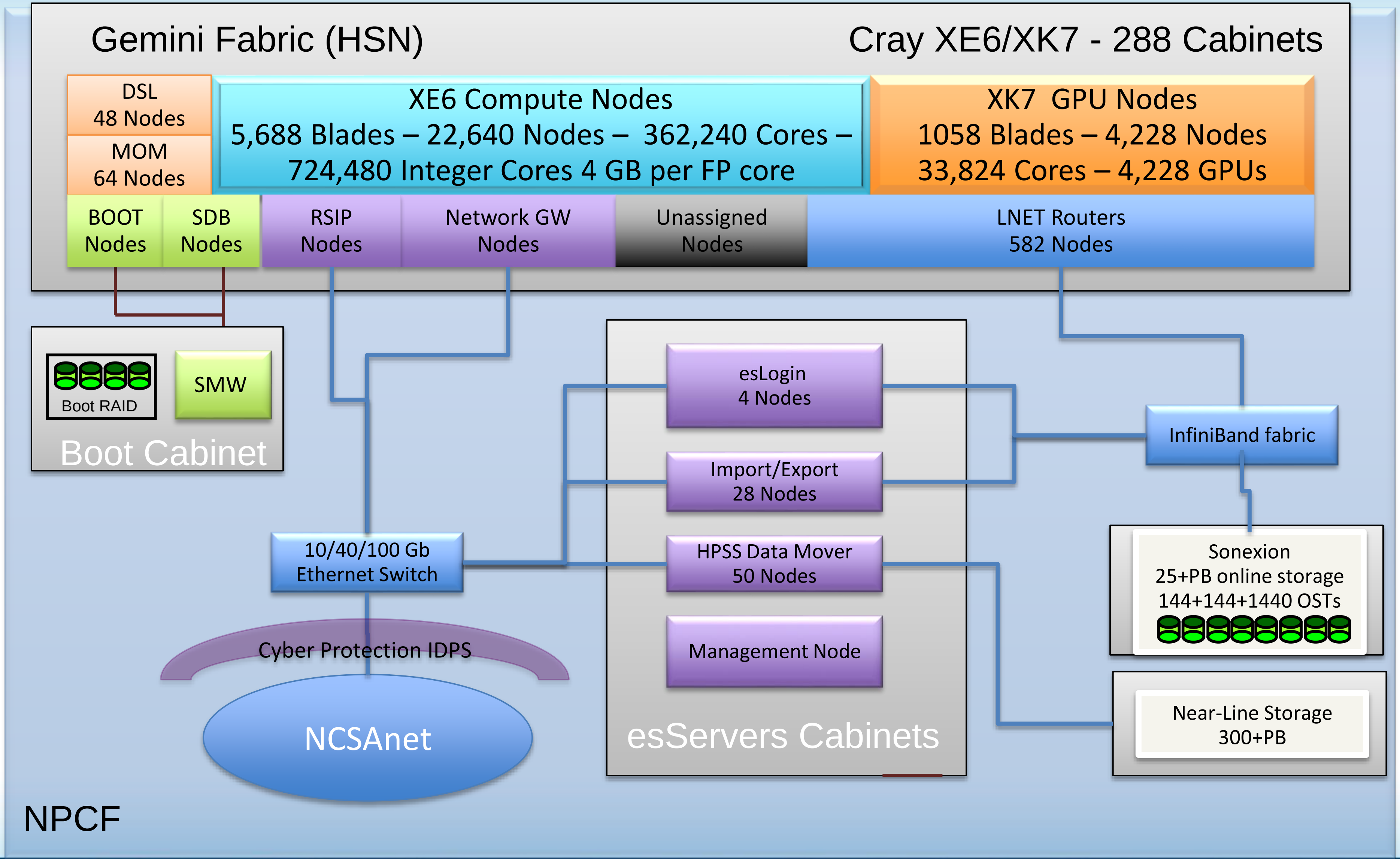
Courtesy: Blue Waters Project, NCSA (not intended for reuse)

National Petascale Computing Facility

- Modern Data Center
  - 90,000+ ft^2 total
  - 30,000 ft^2 6 foot raised floor
    20,000 ft^2 machine room gallery with no obstructions or structural support elements

- Energy Efficiency
  - LEED certified Gold
  - Power Utilization Efficiency, PUE = 1.1–1.2
  - 24 MW current capacity – expandable
  - Highly instrumented
  - Evaporative Tower Cooling System

Courtesy: Blue Waters Project, NCSA (not intended for reuse)



Courtesy: Blue Waters Project, NCSA (not intended for reuse)

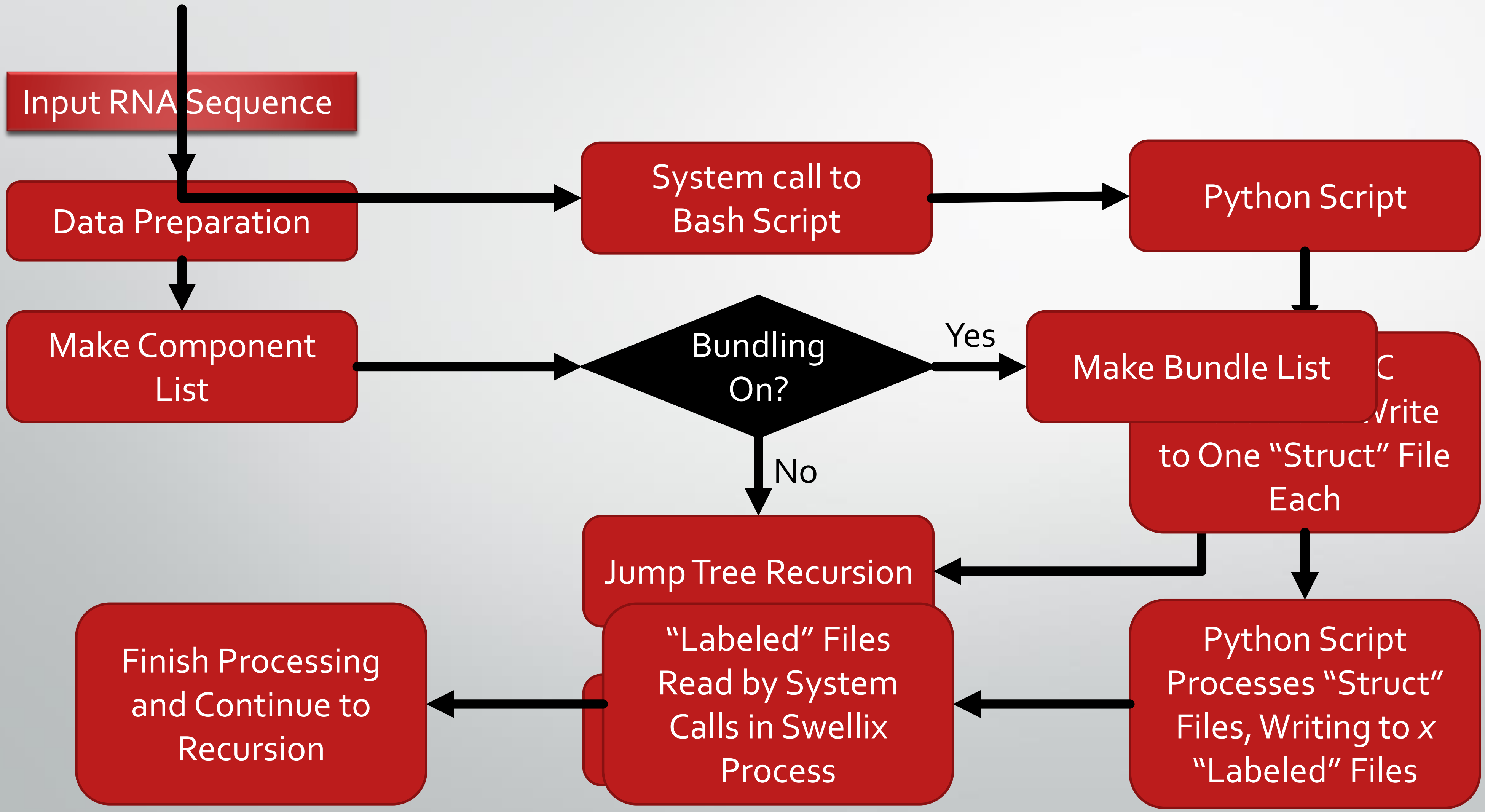
Parallelization

Strategies and Hurdles

Accounting for MPI Processes

- Existing code base nonuniform
  - External C executables, Python scripts
- Large dependency on I/O mid-computation
- Desire ability to predict workload on processing elements

Bundling Fragmentation



Input RNA Sequence

Data Preparation

Make Component List

System call to Bash Script

Python Script

Bundling On?

Yes

No

Make Bundle List

C

Vrite

to One “Struct” File Each

Jump Tree Recursion

Python Script Processes “Struct” Files, Writing to *x* “Labeled” Files

“Labeled” Files Read by System Calls in Swellix Process

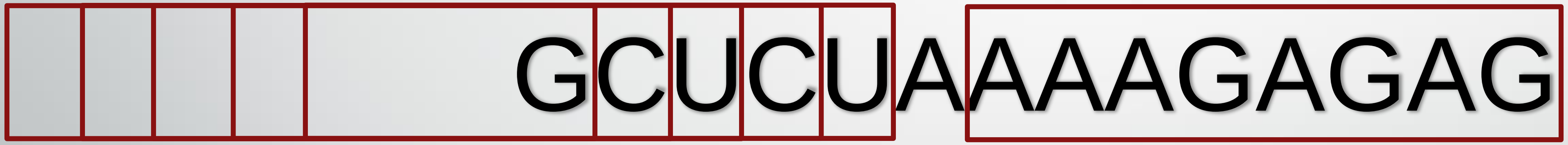
Finish Processing and Continue to Recursion

n = sequence length

Typically, *x* >> *n*

Bundling Is a Parallelization Candidate

- Bundling method – e.g. "sliding" interval size = 8
- For sequence length *n*, there are *n* unique intervals evaluated
- Independent

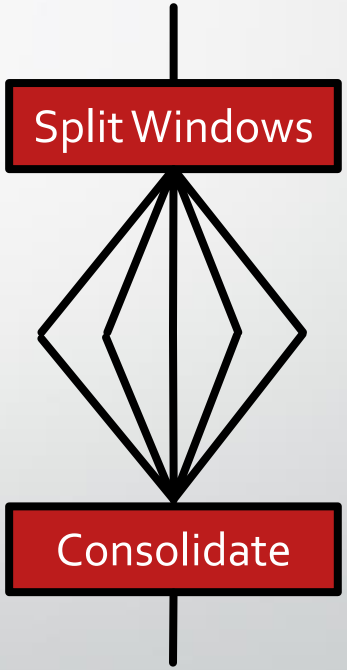


GCUCUAAAAGAGAG

1. G
2. GC
3. GCU
4. GCUC
5. GCUCU
6. GCUCUA
7. GCUCUAA
8. GCUCUAAA
9. CUCUAAAA
10. UCUAAAAG
11. CUAAAAGA
12. UAAAAGAG
13. AAAAGAGA
14. AAAGAGAG

Bundling Parallelization

- For *n* MPI processes, divide number of "windows," *x*, evenly across all *n* elements
- Construct output datatypes locally; then use MPI_Allgatherv to disburse all computed pieces to all other PEs
- Similar to OpenMP "parallel for" pragma



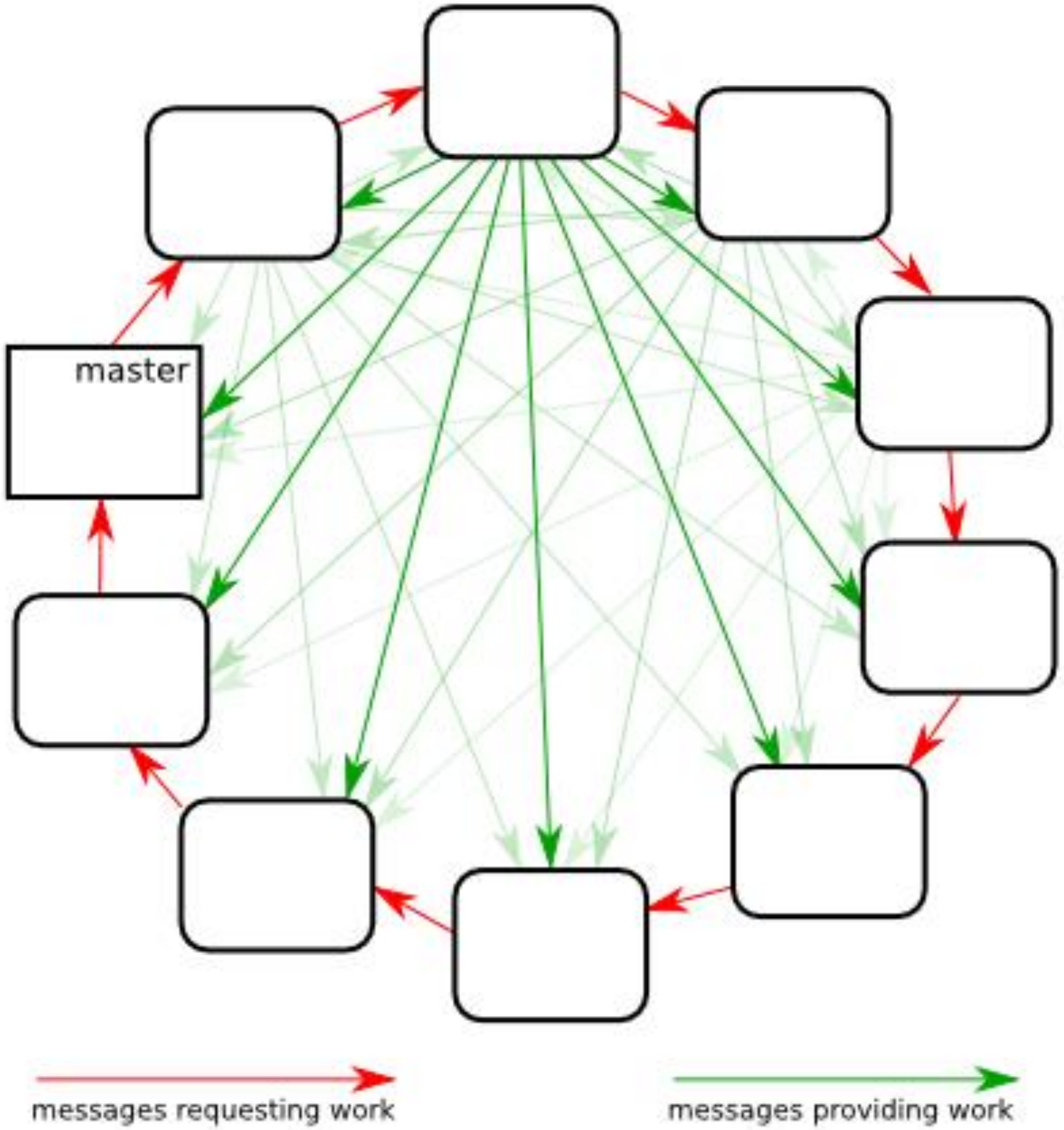
Recursion is Problematic

- Tree balance
- Unusual data to transmit

Preliminary Unbalanced Load Distribution with 64 PEs
Run on Boomer cluster

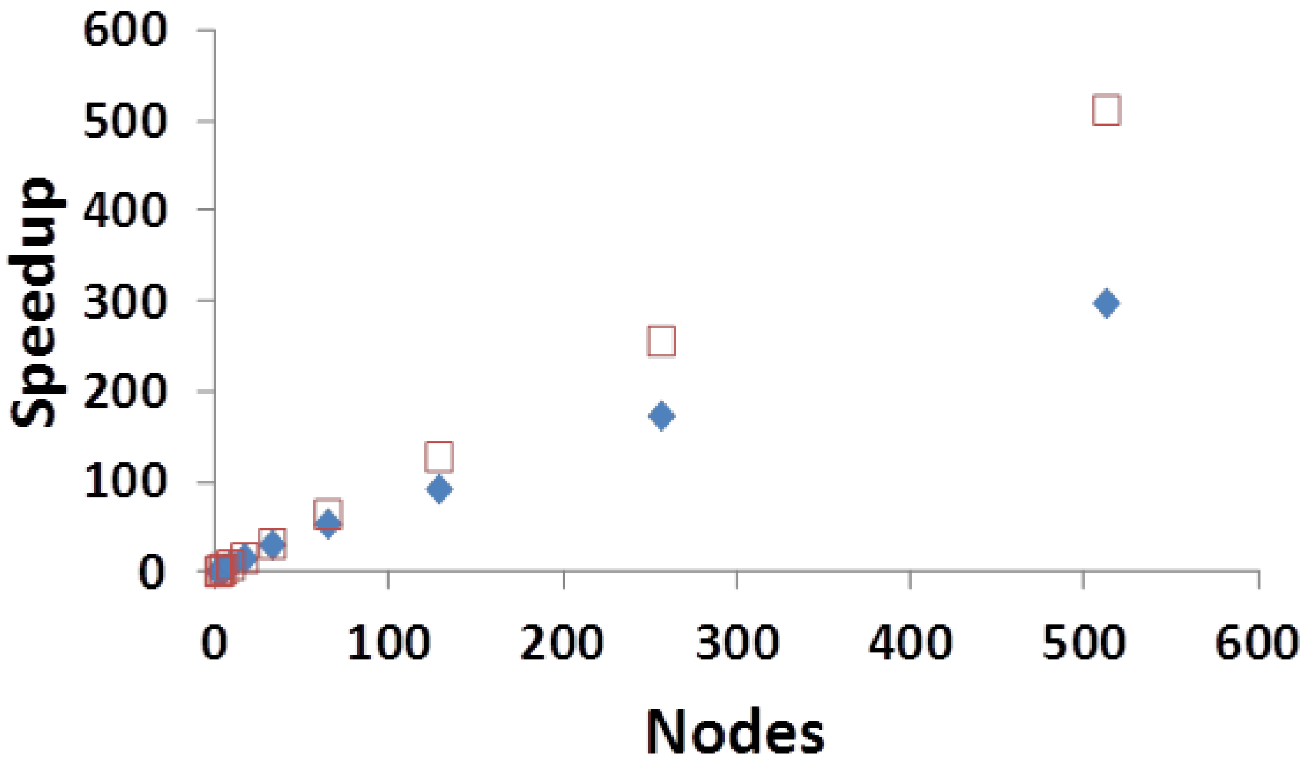
Sequence	Length (nt)	Total Structures	Serial Runtime (sec)	MPI Runtime (sec)
tRNA Azoarcus	197	2.122×10^{10}	88674.91	4122.37

A Parallel Algorithm Exists



Sam Bleckley
Jon Stone
PlosOne 2012 7(12): e52414.
doi:10.1371/journal.pone.0052414

Ring Node Topography

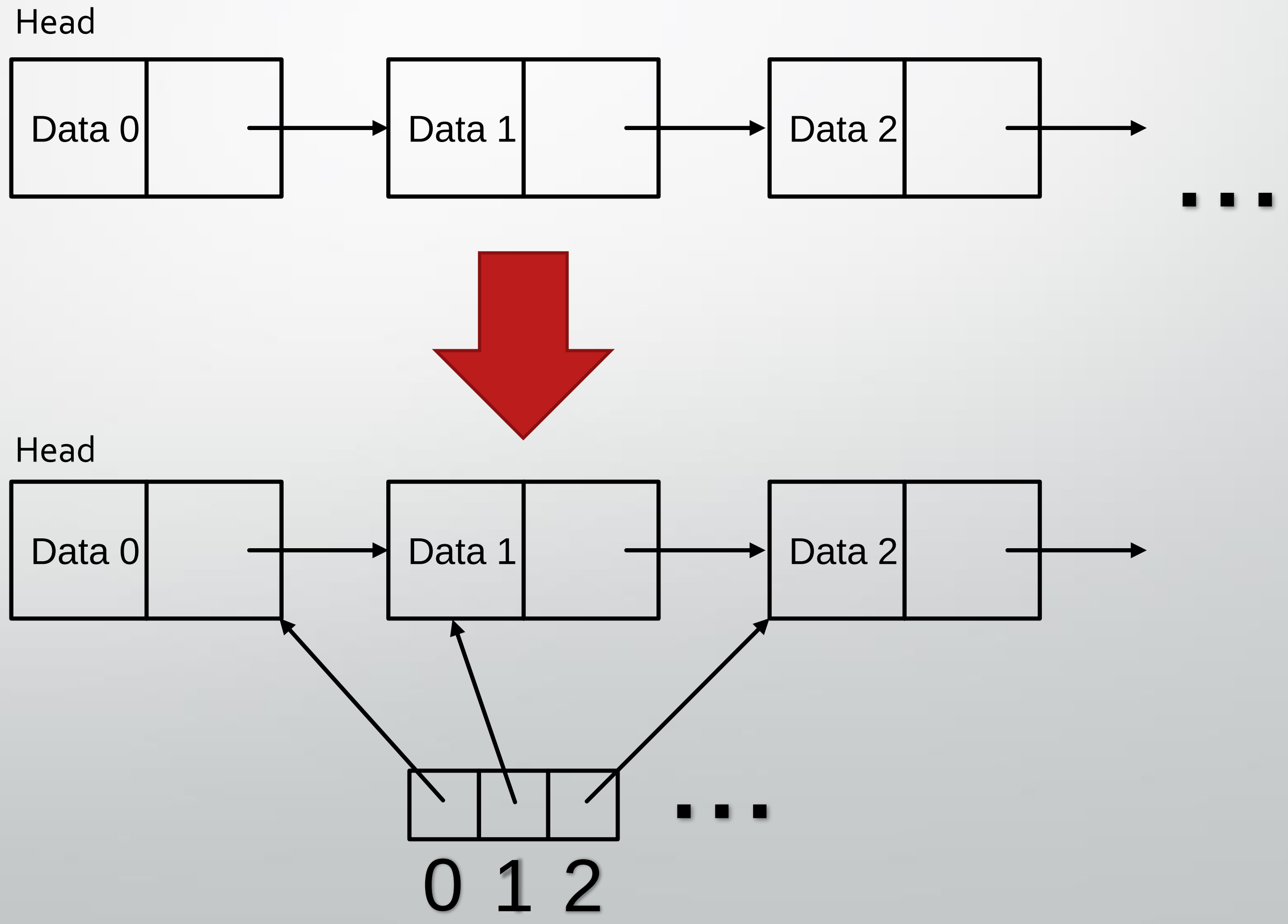


Number of Processes	Real Time (s)	Average Percent Work Per Process
4	42264	19.996 ± 10.001
8	21118	11.107 ± 3.956
16	10774	5.878 ± 1.488
32	5630	3.206 ± 0.570
64	3171	1.533 ± 0.285
128	1926	0.771 ± 0.180
256	999	0.384 ± 0.073

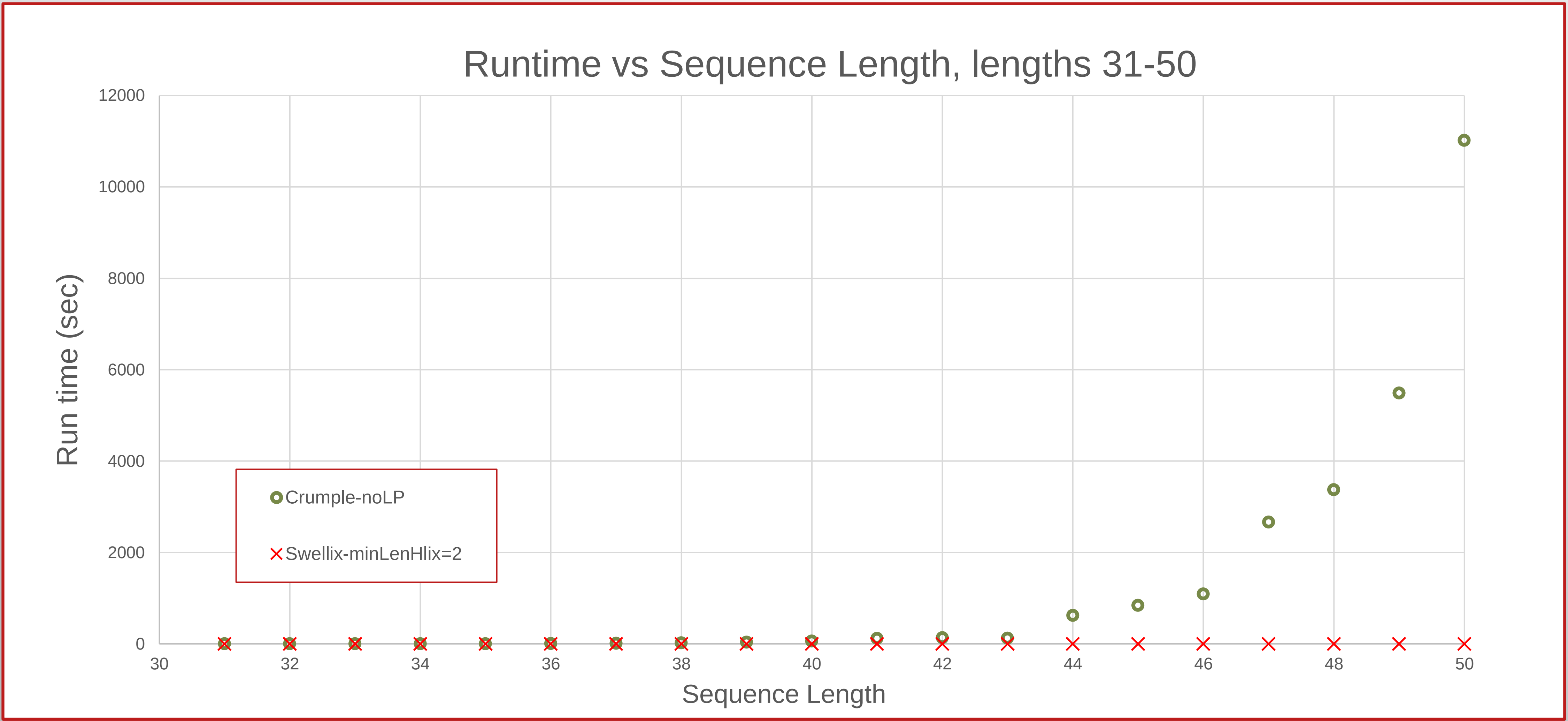
Sam Bleckley
Jon Stone
PlosOne 2012 7(12): e52414.
doi:10.1371/journal.pone.0052414

Dealing With the Datatypes

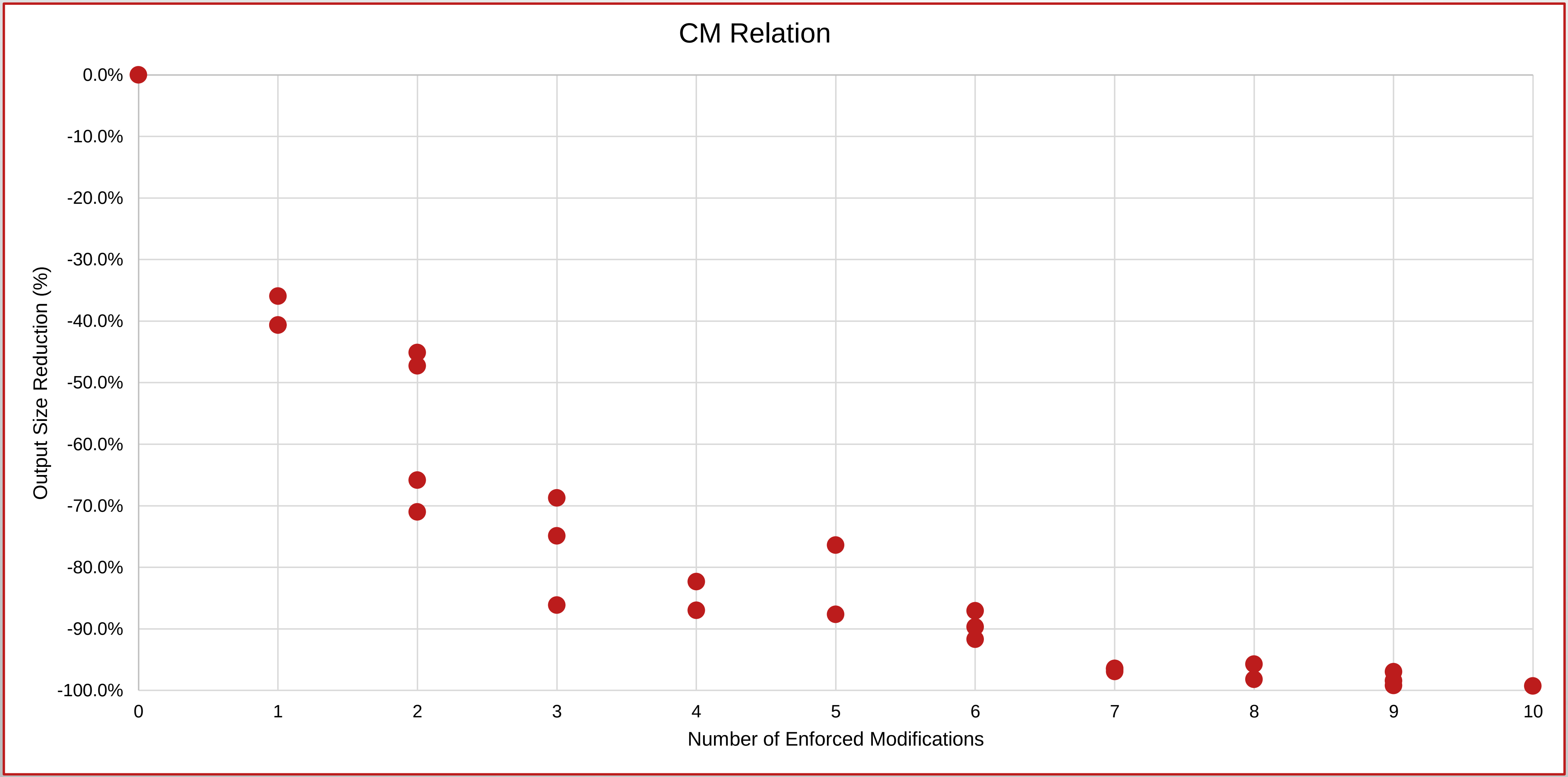
- Linked lists with MPI
- Serialization of recursion-specific structs
- Other approaches



Crumple vs Swellix

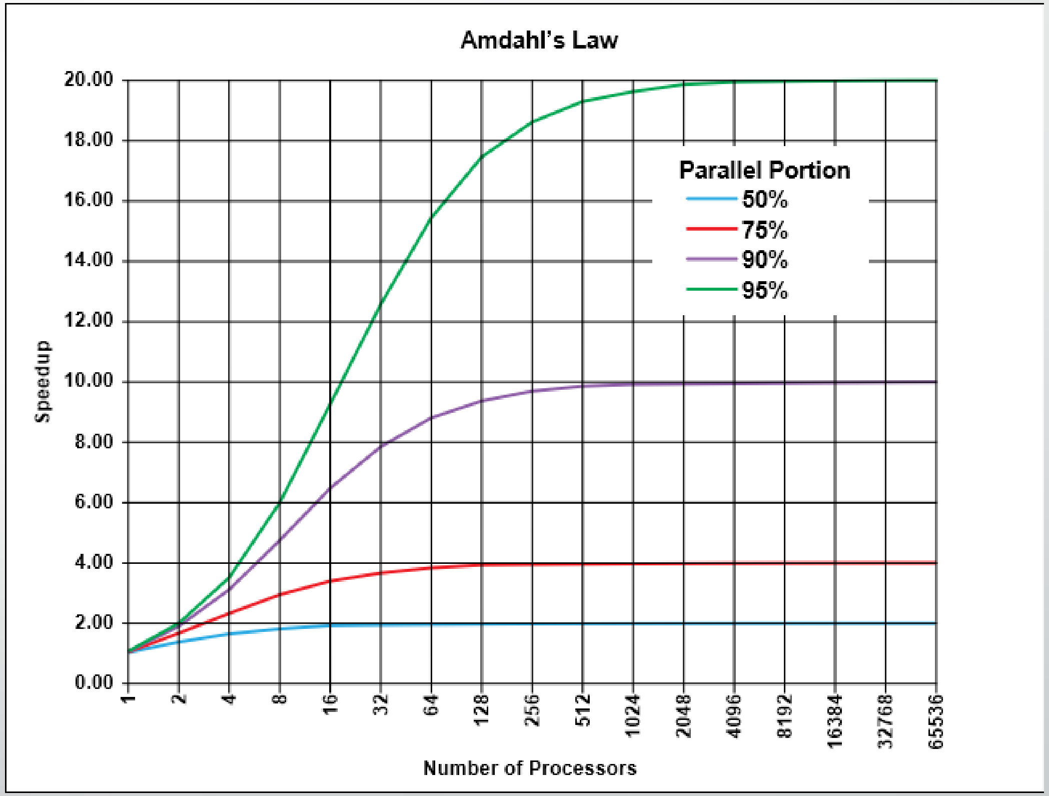


tRNA Conformational Space



Future plans

- Finish MPI implementation
- Accelerators
- OpenMP
- HERV-K
- STMV
- Further optimization of existing code



"AmdahlsLaw" https://commons.wikimedia.org/wiki/File:AmdahlsLaw.svg#/media/File:AmdahlsLaw.svg

Schroeder Lab Members

Front row: Ella Parsons, Lynneah McCarrell, Cari Quick, Kimberly Ughamadhu, Xiaobo Gu

Middle row: Susan Schroeder, Amr Elongdhakly, Alyssa Hill, Thanh Truong

Back row: Wade Craig, Jake Schillo, Nathan Sloat

Special Thanks to:

OSCER Director Henry Neeman, Joshua Alexander

Ryan Liu, Jon Stone, Sam Bleckley, Ivan Guerra, Isaac Sung for previous development of Swellix and other code

The computing for this project was performed at the OU Supercomputing Center for Education & Research and on the Blue Waters supercomputer.

Questions?