A Fast Compact Indexing Algorithm for Managing Large-Scale Robinson-Foulds Distance Matrices

Beenish Jamil
bjamil@gmu.edu

Department of Computer Science & Engineering,
Texas A&M University, College Station, TX

Tiffani L. Williams
tlw@cse.tamu.edu

Introduction

Phylogenetic analysis can produce up to many thousands of trees, with each tree being a hypothesis for the evolutionary relationships between a group of organisms or taxa. The difference between these trees can be found using the Robinson-Foulds (RF) distance metric, a commonly used metric for phylogenetic analysis, and then stored in a matrix. However, there are currently no efficient methods for extracting data from large RF matrices due to I/O bottlenecks.

We propose an indexing algorithm to deal with one of the more time consuming of such searches: finding all occurrences of an RF value in the entire matrix.

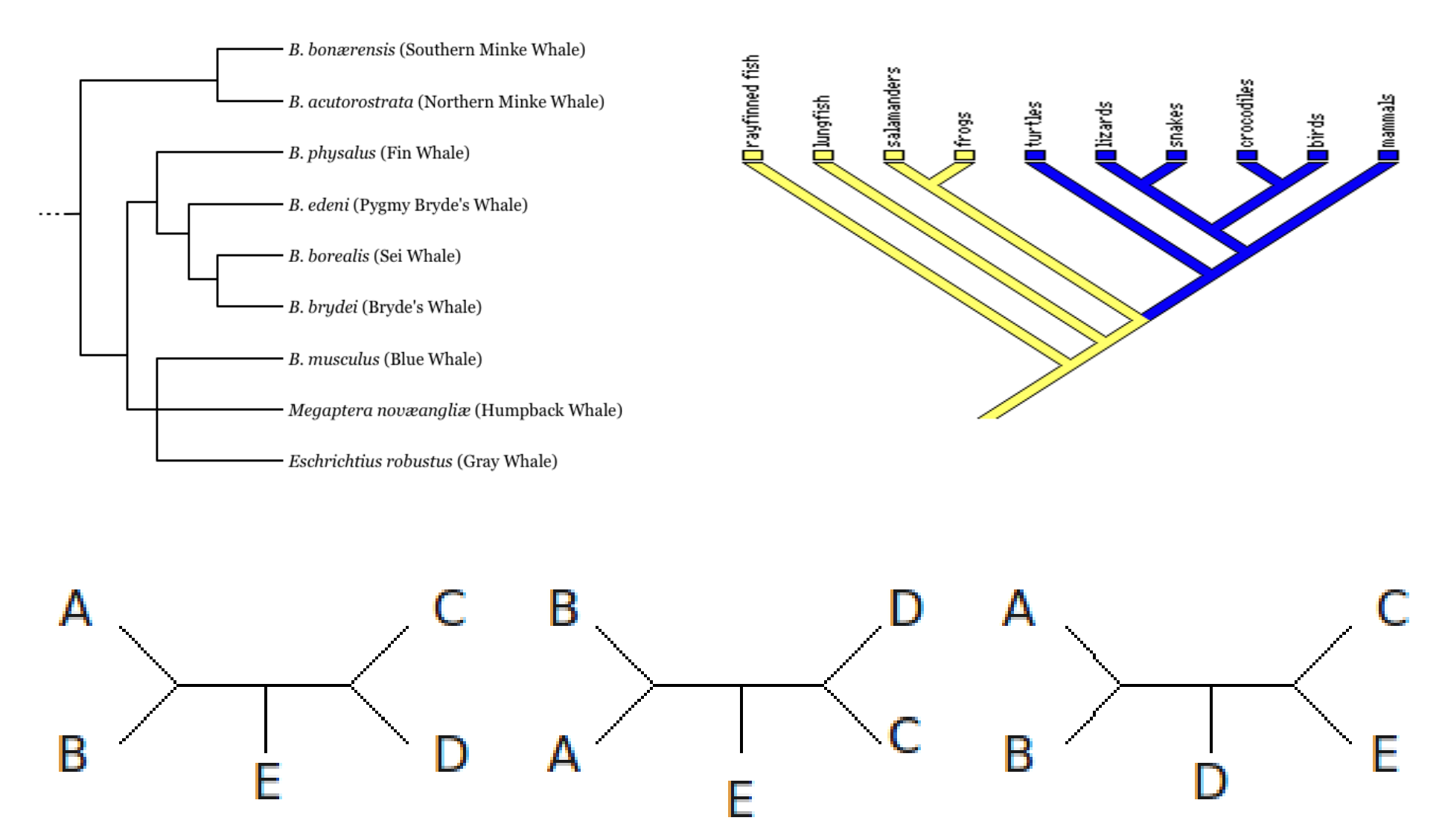

Figure 1: *Top*: Sample phylogenetic trees of whales (left), animals (right)
Bottom: From the left, the first and second trees are identical while the third one is 1 RF away from them.
Source: http://www.med.nyu.edu (top left) http://commons.wikimedia.org (top right) http://iysik.com/index.php?page=comparing-trees (bottom)

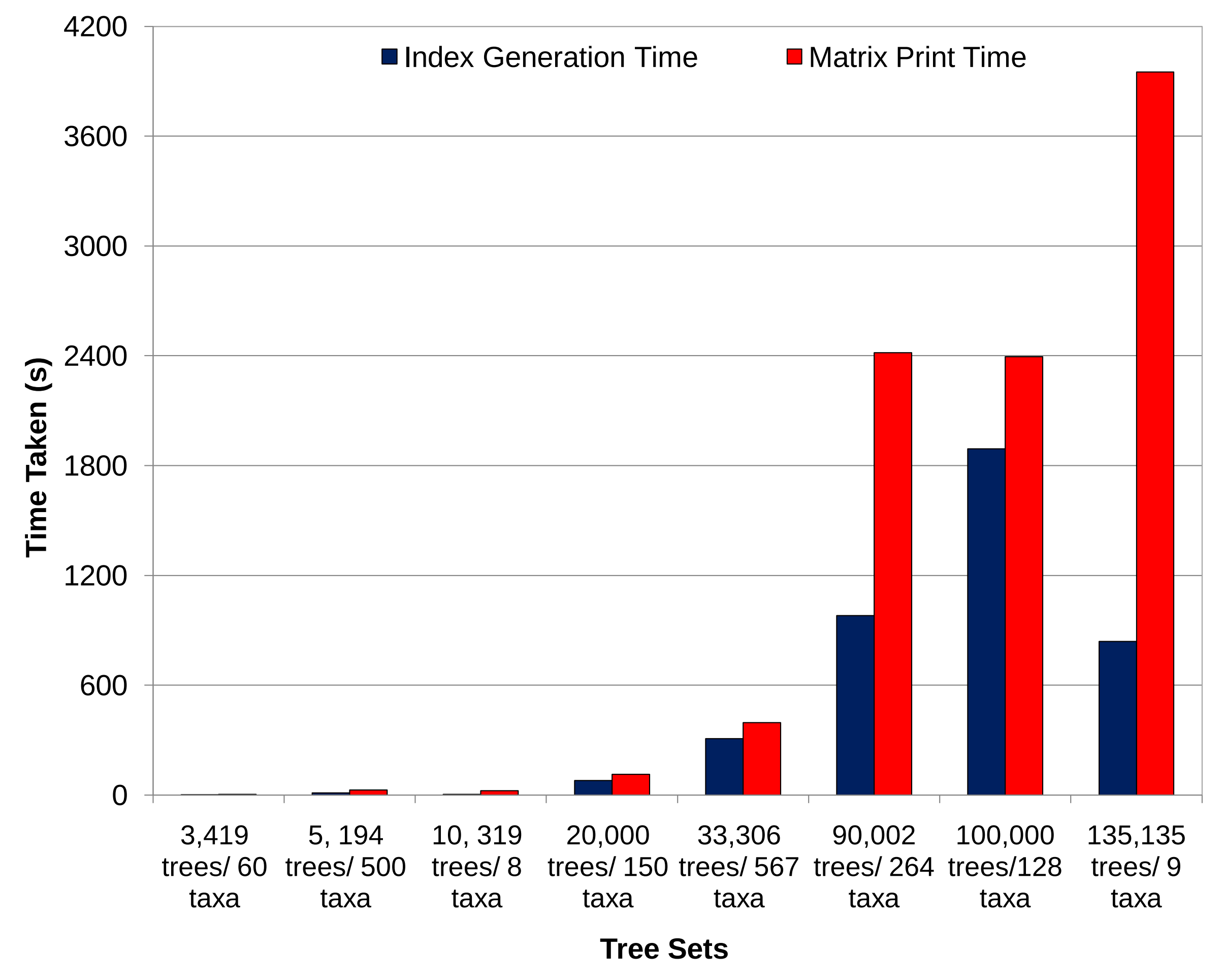

Figure 2: Relative Time For Generating an Index (blue) and Printing out the RF Matrix (red) for Various Tree Sets. This figure shows generating and storing the index usually takes less time for larger tree sets

The Algorithm

RF Matrix

	0	1	2	3	4	5
0	0	1	2	2	3	3
1	1	0	3	1	2	1
2	2	3	0	2	2	2
3	2	1	2	0	2	1
4	3	2	2	2	0	1
5	3	1	2	1	1	0

Cell Positions in the RF Matrix

	0	1	2	3	4	5
0	0	1	2	3	4	5
1	6	7	8	9	10	11
2	12	13	14	15	16	17
3	18	19	20	21	22	23
4	24	25	26	27	28	29
5	30	31	32	33	34	35

Index Files

RF 0	RF 1	RF 2	RF 3
	34-33	32	30
	31	27-25	24
	19	20	13
	6	18	
		12	

Results & Discussion

Index Generation Time:
- The slowest part of the algorithm is writing to file.

.
- All data for one RF distances is written to file whenever there are at least 100,000 values to write for that RF.
- The presence of identical trees and long runs of the same RF distance in the matrix allows more matrix cells to be processed before the data was written to the index files. This is because, of these values, only the start and end points of the runs are written to file.
- These less frequent writes help speed up the indexing process
- Program performance was best when there were a lot of consecutive runs and identical trees present and worst when there were few to none. .

Results & Discussion (continued)

Index Search Performance
- The algorithm was optimized for searches where only RF values of the desired trees were known.
- For such queries in the matrix, the entire matrix is searched, which can take up considerable time.
- The time required to return these values from the index is only the time required to read the index, parse the read values into their correct row/column parts and print them to screen.
- Although this, too, is limited by file I/O, the index file sizes of large RF matrices will always be less than the matrix size; therefore, the search space is still smaller for the index files. Consequently, less time is required to search the index

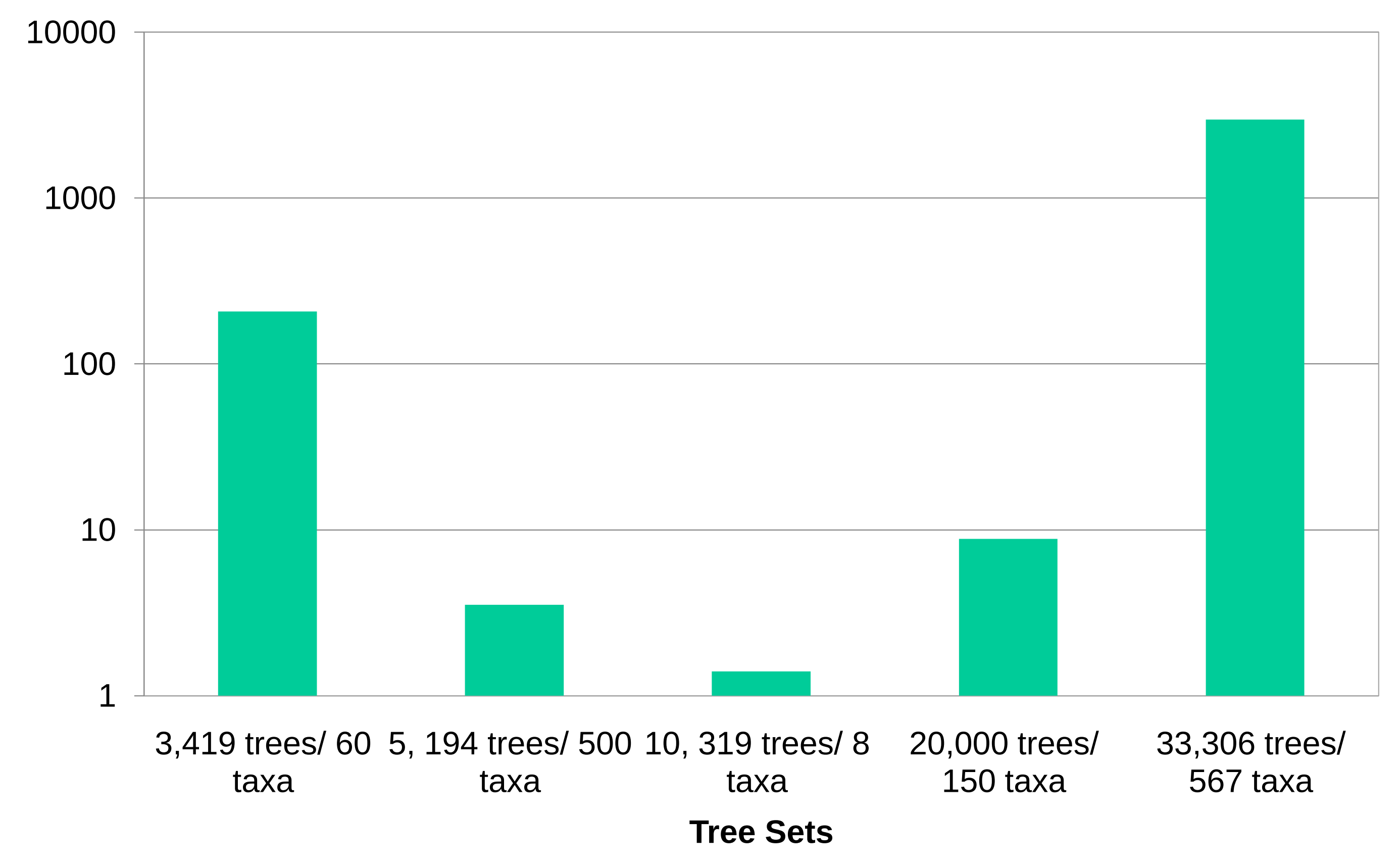

Figure 3: Speedup in Searching the Index over the RF Matrix. Searching the index was 1.4 times to more than 2,000 times faster than searching the matrix in our experiments.

Future Work

- Create efficient querying algorithms
- Introduce more diverse queries and modify this indexing algorithm to support them, if possible.
- Implement a method that reorders trees so that there are more/longer continuous runs of the same RF distance

Acknowledgements

I would like to thank the DREU program for making this research experience possible.