IOWA STATE UNIVERSITY
Department of Computer Science

Honors Poster Presentation
12/06/2020

Katherine Braught and David Fernández-Baca

Generating Decisive Sub-matrices for Phylogenetic Tree Construction

Background

A large problem when building phylogenetic trees is data incompleteness. A significant amount of missing data can lead to ambiguity in the structure of tree, leaving biologists unsure of the accuracy of the tree. However, even with incomplete data, an unambiguous phylogenetic tree can still be built depending on the makeup of data. Data can be represented using a data availability matrix which indicates whether sequencing data is available for a given taxon at a given loci. Each row represents an organism, and each column represents a loci.
Sanderson and Steele found that these matrices can be used to check if an unambiguous tree can be built. They showed that if a matrix is decisive, then you can confidently build a phylogenetic tree from that data. However, if a matrix is indecisive, we can find a decisive submatrix, which would correspond to a subset of organisms for which we can build a phylogenetic tree. Checking decisiveness of a matrix is equivalent to solving the no rainbow 4-coloring problem on the matrix's hypergraph. We can formulate this NP-complete problem as both an integer linear programming instance and satisfiability instance for efficient solving in practice. Once we find a decisive submatrix, a biologist can confidently construct a phylogenetic tree for those taxa.

SAT Formulation

To check for check for decisiveness, we formulate the no rainbow 4-coloring problem as a SAT problem. We use the following constraints and their corresponding Boolean formulas to check for a coloring. Let $x_{ia} = true \leftrightarrow x_i$ has color a. x_i represents one of the n taxa.
(1) Each taxa has only 1 color.

$$\bigwedge_{i\in[n]} x_{i1} \oplus x_{i2} \oplus x_{i3} \oplus x_{i4}$$

(2) Each color appears at least once in the coloring.

$$\bigwedge_{q\in[4]} \bigvee_{i\in[n]} x_{iq}$$

(3) No loci is rainbow colored. Let k be the number of loci and Y_j be the set of all taxa who have data for loci j.

$$\bigwedge_{j\in[k]} \bigvee_{q\in[4]} \bigwedge_{i:a_i\in Y_j} \neg x_{iq}$$

We also use an existing ILP formulation of the no rainbow 4-coloring.

Software Pipeline

We built a software pipeline that generates a decisive submatrix from an indecisive data availability matrix. The pipeline works as follows:

```
while the matrix is not decisive:
1.  add/remove rows using heuristic
2.  Generate an ILP or CNF Boolean formula
3.  Solve the formulation
```

We solve the ILP instances using the Gurobi ILP solver, a paid software available under academic license. We tested several SAT solvers from the PySAT library.

Data Set	# Loci	# Taxa
Allium	6	57
Asplenium	6	133
Caryophyllaceae	7	224
Eucalyptus	6	136
Euphorbia	7	131
Ficus	5	112
Iris	6	137
Meredith.mammals	6	169
Miadlikowska.fungi	9	1317
Primula	6	185
Rabosky.scincids	6	213
Ranunculus	7	170
Rhododendron	7	117
Rosaceae	7	529
Shi.bats	9	815
Solanum	7	187
Soltis.saxifragales	1	946
Szygium	5	106
Tolley.chameleons	6	202

Table 1: Sizes of initial data sets

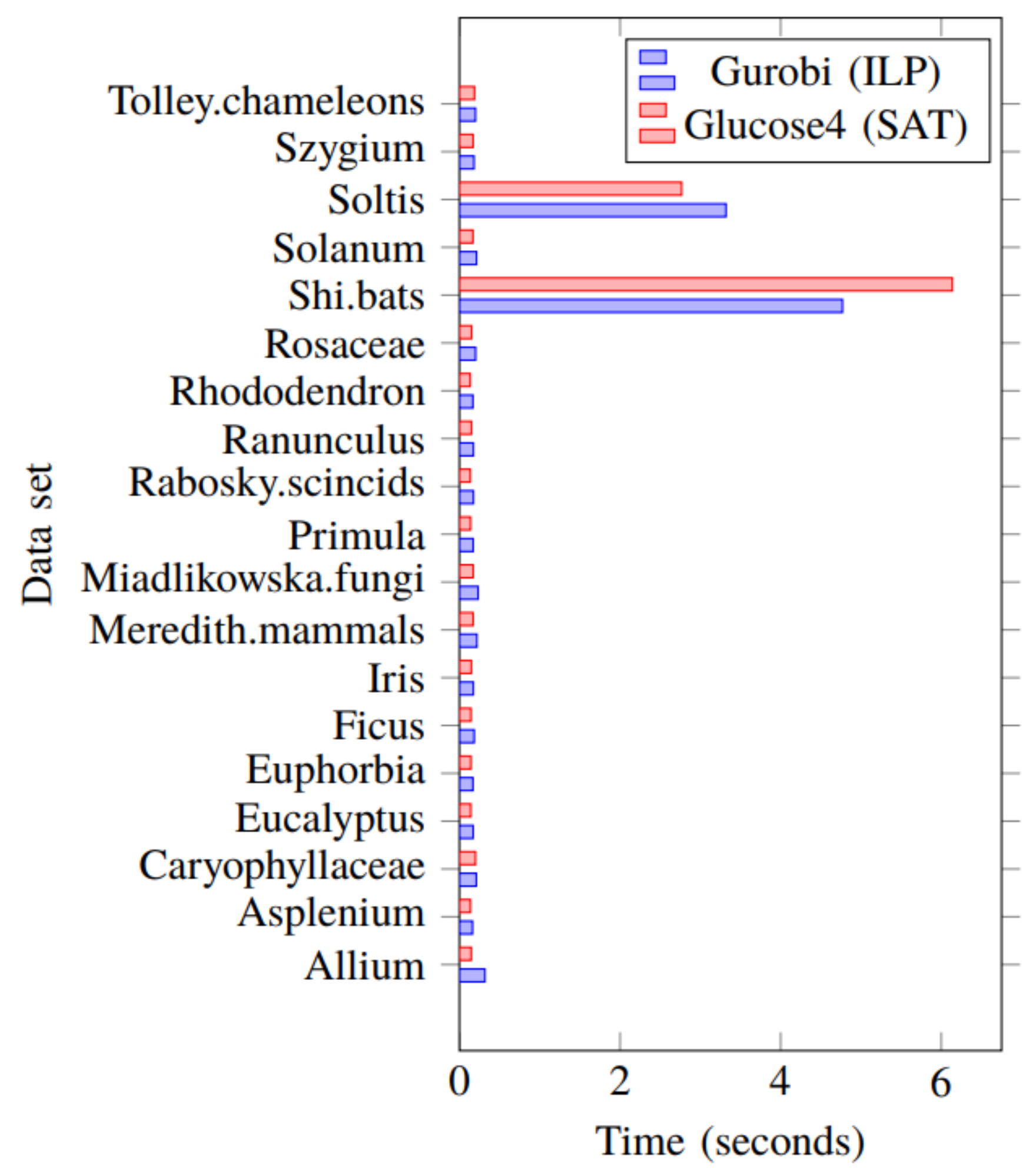

Figure 1: Time to generate submatrix using remove method

	Remove Method			Add Method		
Data Set	Final Size	Gurobi time (sec)	Glucose 4 time (sec)	Final Size	Gurobi time (sec)	Glucose 4 time (sec)
Allium	6	0.316	0.146	11	0.269	0.192
Asplenium	32	0.162	0.136	132	0.147	0.164
Caryophyllaceae	69	0.209	0.199	124	x	0.296
Eucalyptus	3	0.17	0.14	134	0.155	0.165
Euphorbia	82	0.167	0.142	127	0.163	0.169
Ficus	26	0.183	0.143	95	0.238	0.201
Iris	25	0.172	0.147	129	0.16	0.17
Meredith.mammals	168	0.217	0.17	168	0.318	0.291
Miadlikowska.fungi	320	0.231	0.17	1192	x	5.349
Primula	29	0.17	0.138	171	0.235	0.21
Rabosky.scincids	207	0.174	0.135	208	0.158	0.182
Ranunculus	15	0.173	0.148	164	0.194	0.2
Rhododendron	63	0.167	0.133	87	0.184	0.181
Rosaceae	64	0.2	0.15	415	x	1.769
Shi.bats	35	4.769	6.137	x	x	x
Solanum	27	0.212	0.167	151	x	0.361
Soltis.saxifragales	16	3.319	2.766	x	x	x
Szygium	85	0.18	0.169	94	0.136	0.164
Tolley.chameleons	188	0.199	0.186	194	x	0.27

Table 2: Timing and size results for all experiments

Experiment

We ran the pipeline on real data provided by Dobrin, Zwickl, and Sanderson. The pipeline was always run on the same computer with four cores and a Linux operating system with no background programs running. From these experiments, we wanted to know:
1: Are the theoretical results feasible for use in practice?
2: What problem formulation should be used?
3: What heuristic yields the best sub-matrices?
The heuristics were tested were basic:
Remove method: Repeatedly remove a taxon that has the fewest number of non-zero entries until a decisive submatrix is found.
Add method: Remove taxa as described in the remove method and add back sets of taxa in the powerset of initially removed taxa until the largest decisive sub-matrix is found.
We ran four types of experiments: add method with ILP, add method with SAT, remove method with ILP, and remove method with SAT.

Results

Timing Results
We tested several SAT solvers and found that Glucose 4 was generally the fastest, so we use those results to compare to the ILP results.
We see that in most cases, ILP and SAT formulations run in comparable time.
We also see that on smaller matrices, the add and remove methods run in similar time, but more programs time out when using the add method.

Size Results
The add method consistently produces larger decisive submatrices when it finished. Several of the submatrices produced by the remove method for organisms like fungi and chameleons were relatively large submatrices.

Other Results
When analyzing the resulting submatrices, we saw that all the matrices produced using the add method were trivial, meaning they all shared one gene in common. Many of the matrices produced by the add results were also trivial, but several were non-trivial. Some of the non-trivial matrices had good coverage over several families of organisms.

Conclusion

Formulation
Use the SAT formulation in future software - SAT and ILP run in comparable time, and SAT solving software is available open source and free to anyone.

Heuristic
Avoid the add method – this method works poorly on large matrices, runs slower, and produces trivial matrices.
Remove method shows feasibility – though remove method often finds trivial matrices, it shows it is possible to find non-trivial submatrices with good coverage using these ideas.

Future Work
Find a better heuristic – we need a theoretically backed way to choose rows to eliminate to get more non-trivial results.

References

Alexey Ignatiev and Antonio Morgado and Joao Marques-Silva(2018). PySAT: A Python Toolkit for Prototyping with SAT Oracles.In SAT (pp. 428–437).
B. H. Dobrin, D. J. Zwickl, and M. J. Sanderson, "The prevalence of terraced treescapes in analyses of phylogenetic data sets," BMC Evolutionary Biology, vol. 18, p. 46, 2018.
Parvini, G., Braught, K., Fernández-Baca, D.: Checking phylogenetic decisiveness in theory and in practice (February 2020), arXivpreprinthttps://arxiv.org/abs/2002.09722
Steel, M., Sanderson, M.J.: Characterizing phylogenetically decisive taxon coverage. Applied Mathematics Letters23(1), 82–86(2010)

Acknowledgements: I would like to thank my mentor, David Fernández-Baca, for his guidance and mentorship throughout this entire project. I would also like to thank his graduate student, Ghazaleh Parvini, and Dan Gusfield for their helpful collaboration.