# 176 Literature Review: Inference of single-cell phylogenies from lineage tracing data using Cassiopeia

Joyce Lu, Joon Kim, Timothe Kasriel

April 2025

## 1 Graphical Abstract

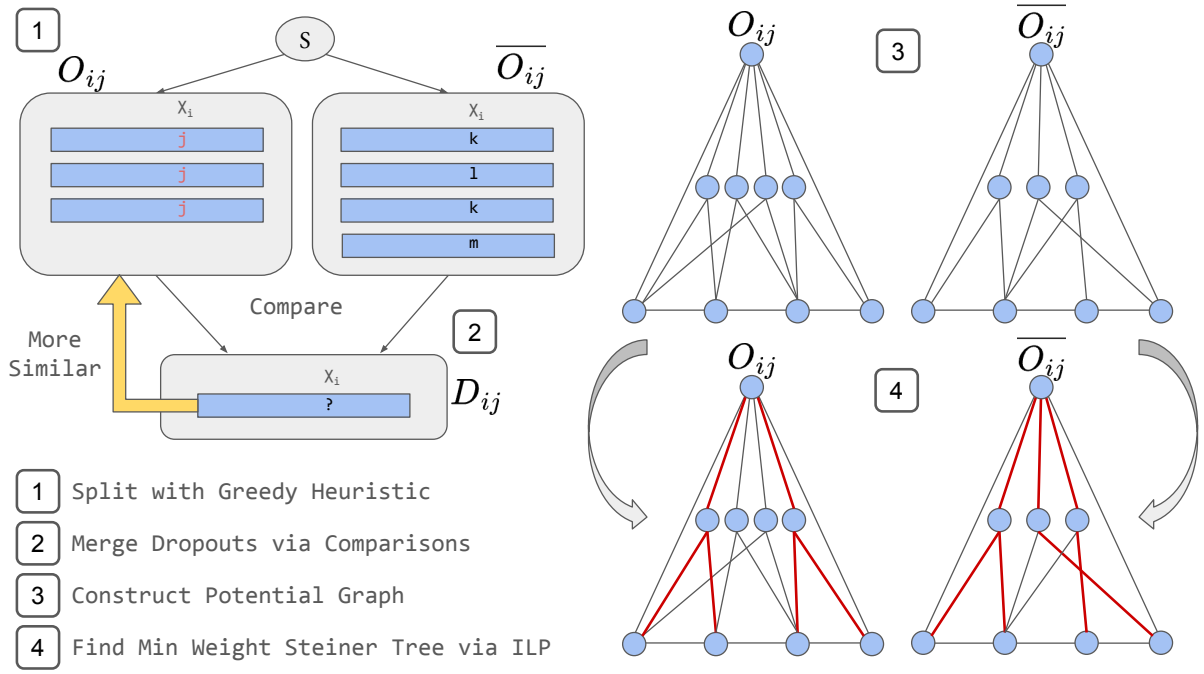


Figure 1: Workflow of Cassiopeia-Hybrid, which combines Cassiopeia-Greedy and Cassiopeia-ILP. The Greedy heuristic is repeated recursively (1 and 2) until each subtree is sufficiently small, then ILP solves for the exact most parsimonious Steiner Tree (3 and 4). Greedy helps cut down the runtime at the expense of some correctness, while ILP focuses on achieving the best performance on a reasonably small subproblem found by Greedy. Each of Greedy and ILP and also be used separately. 1) For a given dataset $S$, find the maximum occurring mutation $j$ in any one character $i$ and split based on its existence to $O_{ij}$ or $\overline{O_{ij}}$. 2) For data where character $i$ is dropped, categorize them into a separate set $D_{ij}$ and merge each to the more similar set in other characters. 3) Switch to Cassiopeia-ILP, and build a "Potential Graph" that only includes edges and intermediate nodes likely to be a part of the most parsimonious tree. 4) Run Integer Linear Programming to find the minimum solution to the Steiner Tree instance specified by the Potential Graph.

## 2 Motivation and Prior Work

Cassiopeia aims to accurately infer phylogenetic relationships from large-scale datasets. Recent developments in CRISPR/Cas9 gene editing and single-cell sequencing have enabled lineage tracing across many cells and multiple generations [11] [9]. One such procedure is described as follows:

- Induce a heritable Cas9-mediated insertion or deletion ("indel") at a target site,
- Sequence the resulting indels, and
- Infer cell relationships using an algorithm that reconstructs a phylogenetic tree from the indel data.

In particular, while the coupling of CRISPR/Cas9 with single-cell readouts has made it possible to produce large-scale data, such data is highly complex, and existing phylogenetic reconstruction algorithms may not be well-suited for this problem. Traditional phylogenetic tree reconstruction methods include:

- *Distance-based methods:* Given a pairwise dissimilarity map, construct a phylogenetic tree that best reflects the distances. Examples include Neighbor Joining [12] and Phylogenetic Least Squares [2][5].
- *Character-based methods:* Given a set of characters, find the most parsimonious tree (i.e., the tree that minimizes the number of mutations). One such example is Camin-Sokal [1].

For the specific task of inferring single-cell phylogenies from lineage tracing data, we aim to develop a tree reconstruction algorithm that satisfies the following properties:

1. Scalability,
2. Robustness to dropouts (since lineage tracing experiments experience a non-negligible amount of missing data due to heritable or stochastic dropouts), and
3. Exploitation of lineage tracing-specific properties (mutation irreversibility: once a site is edited, it cannot be recut by Cas9, so all its descendants inherit the indel; the unedited state of founder cell: root of the phylogeny contains only uncut target sites).

From a theoretical standpoint, traditional algorithms were developed for small-scale data and may not be scalable. Furthermore, they were not developed for this use case and often fail to fully satisfy properties (2) and (3). Empirically, existing algorithms have not been extensively tested on lineage tracing data and may therefore be ineffective in handling lineage tracing from large-scale datasets.

To address those issues, the authors present Cassiopeia, a suite of algorithms satisfying the desired properties. They also develop a benchmarking resource via a simulation engine and an *in vitro* experiment to evaluate the effectiveness of phylogenetic tree reconstruction algorithms on lineage tracing data. The details of Cassiopeia are presented in section 3. Below, we briefly describe the benchmarking results.

**Benchmarking Resource**:

- *Simulation engine:* Lineages were simulated by varying parameters such as the number of characters, number of states, probability distribution of states, mutation rate per character, number of cell generations, and amount of missing data. Using experimental data, default

parameter values were estimated, and in each simulation run, only one parameter was varied from its initial value. Phylogeny reconstruction algorithms were evaluated using the combinatorial "triplets correct" metric, which measures at the proportion of cell triplets ordered correctly by the algorithm. [1]

- *In vitro reference experiment*: To establish a "ground truth" for experimental data, the researchers tracked the clonal expansions of 10,000 cells (11 clones over 21 days) using a lineage tracing technology [3]. Full details are found in the paper [8].

# 3 Computational Methodology

Cassiopeia is a parsimony-based phylogeny tree reconstruction algorithm that improves upon the neighbor-joining algorithm. Cassiopeia introduces three different methodologies for phylogenetic tree reconstruction: Cassiopeia-Greedy, Cassiopeia-ILP, and Cassiopeia-Hybrid. At a high level, Cassiopeia-Greedy is a fast but inaccurate method most useful for approximating "obvious" steps, while Cassiopeia-ILP will provide a guaranteed correct tree, should one exist, but involves solving an NP-Complete ILP problem. Cassiopeia-Hybrid combines both algorithms such that each algorithm makes up for the weaknesses of the other. In effect, Cassiopeia-Hybrid interpolates between the trade-offs of the Greedy and ILP algorithms.

Cassiopeia-Greedy is an adaptation of Dan Gusfield's greedy algorithm [6]. In Cassiopeia-Greedy, the tree is formed top-down. At every step, for every character $\chi_i$, all cells can be separated into a collection of sets $O_{ij}$ such that for all cells in $O_{ij}$, $\chi_i(x) = s_j$. One can then solve for the maximal size of this set, obtaining:

$$i, j = \arg\max_{ij} |O_{ij}|$$

If the prior probabilities of each character mutating to each possible state are known (or reasonably estimated), the heuristic can be naturally modified to:

$$i, j = \arg\min_{ij} p_i(s_0, s_j)^{|O_{ij}|}$$

where $p_i(s_0, s_j)$ are the prior probabilities of character $\chi_i$ mutating to state $s_j$. Intuitively, this takes into account that if an unlikely mutation is observed many times at the leaf nodes, it is very likely that all of those mutations happened very early in the splitting process rather than later. Formally, the paper gives an argument that the frequency of a particular mutation is negatively correlated with the number of times the mutation occurred in the ground truth phylogenetic tree, which gives rationale to the heuristics.

Although not necessary for a heuristic algorithm, Cassiopeia-Greedy has a nice property that on a dataset that admits a perfect phylogeny, it returns such a solution. The reasoning is that the most frequently observed mutation for a single character must be the earliest split for perfect phylogeny, and all subtrees follow by induction.

Cassiopeia relaxes the assumptions that Gusfield's algorithm makes, notably that mutations only occur once, that the characters are binary, and that the data is without dropouts. To accommodate cells whose character state $\chi_i$ is unknown (dropouts), Cassiopeia first separates known cells into $O_{ij}$ and $\overline{O_{ij}}$ and puts dropout cells into a third set, $D_{ij}$. Then, for every cell in $D_{ij}$, Casseopeia finds the average percentage of mutated states shared between the cell and $O_{ij}$, and between the

[1] Overall, Cassiopeia produced more accurate and parsimonious trees than Neighbor Joining and Camin-Sokal.

cell and $\overline{O_{ij}}$. This approximates how "similar" a cell is to these two populations, to provide a best guess as to which state this cell should contain. The cell is then merged with the more similar set.

Cassiopeia-ILP solves a Steiner Tree problem using Integer Linear Programming to reconstruct a feasible tree. A potential graph is given to the ILP to reduce the search space to a reasonably small Steiner Tree instance. It should be noted that **ILP is an NP-complete problem**. To get a sense of the NP-Completeness, if we were to naively find the minimum weight phylogenetic tree, we would be searching over all $O(2^{mn})$ nodes (where $m$ :=number of characters, $n$ :=number of states) that construct a hypercube of all possible strings. However, the structure of the phylogenetic tree gives a natural heuristic to reduce the search space by a considerable amount.

The potential graph of the Steiner Tree is derived by a simple rule: If the edit distance between two leaf nodes is below a certain threshold $d$, add the latest common ancestor (LCA) and the directed edges of its children to the potential graph. Disagreeing characters from the two children are replaced by the unmutated state. This iterative process eventually converges to a single root node, given an appropriate $d$. The paper notes that the threshold hyperparameter $d$ affects the performance of the algorithm. To do this, the authors implemented a procedure to dynamically search for the optimal value of $d$ while constructing the potential graph, given the maximum allowed LCA length $k$ and the maximum neighborhood size $N$. These values serve as a proxy for the runtime or memory constraints for a particular instantiation of the algorithm.

Once the potential graph is decided, we can translate the Steiner Tree problem to an ILP instance and solve it using a well-defined software. The exact formulation of the ILP is not particularly interesting for the analysis of phylogenetic algorithms and is excluded.

However, both algorithms suffer from their flaws. Although Cassiopeia-Greedy is an excellent heuristic in early mutations, it struggles with accuracy during later branches. Meanwhile, Cassiopeia-ILP requires solving an NP-Complete problem. While feasible for small trees, this suffers from exponential runtime growth and is therefore a poor choice for larger trees. To compensate for these flaws, the authors have Cassiopeia-Hybrid, which involves performing Cassiopeia-Greedy up to some threshold value to split the data into a set of disjoint subtrees, and then performing the ILP on each of these subtrees. Empirically, the paper reports a runtime of a few hours on a dataset consisting of several thousand cells. Summarizing:

- Cassiopeia-Greedy recursively splits cells into two groups by identifying the maximally occurring mutation for any character and state pair while considering the dropouts for that particular character. This correctly reconstructs a perfect phylogeny solution **if one exists**.
- If information about the prior probabilities of each mutation happening is known, Cassiopeia-Greedy can exploit it while conducting the tree search.
- Cassiopeia-ILP solves a Steiner Tree problem using Integer Linear Programming to reconstruct a feasible tree. A potential graph is given to the ILP to reduce the search space to a reasonably small Steiner Tree instance. It should be noted that **ILP is an NP-complete problem**.
- Cassiopeia-Hybrid first uses the greedy algorithm up to a threshold depth, then exactly solves the remaining subtree reconstruction via ILP. In practice, this balances out the trade-off between the inexactness of Greedy and the runtime of ILP.

# 4 Applications

The paper mentions three potential applications and extensions of Cassiopeia:

1. Cassiopeia can be used for more general lineage-tracing applications. For demonstration, the paper experimented with zebrafish data generated from GESTALT technology. Cassiopeia-ILP consistently found the most parsimonious solution compared to the Camin-Sokal and Neighbor Joining algorithms. [2]

2. Cassiopeia is expected to interact well with future developments in base editing technology, with lower dropout rates and an increase in the number of edit sites. The caveat is that base editing would limit the number of characters to 4 (A, C, G, T). Nevertheless, an ablation simulation of the trade-off between the number of characters with states suggests that having more characters with fewer states could be desirable for Cassiopeia.

3. Cassiopeia can be extremely effective in settings where the mutation rate for each character (target sites) can be engineered to known values across a wide range. With this additional knowledge of prior probabilities of each indel, Cassiopeia can better distinguish between early and late mutations. Specifically, the paper simulates "Phased Recorders" to show that such dispersion of mutation probabilities increases the inference quality of Cassiopeia.

## 5 Impact and Limitations

Due to the use of a hybrid method, Cassiopeia is adaptable for a large variety of situations which would require perfect phylogeny. While there have not yet been direct applications of the work, it has been postulated to be used in studies interested in cellular evolution, such as this cancer study[4]. Other works have also attempted to build on Cassiopeia to improve its use cases, through the usage of machine learning [7] in order to optimize for barcode entropy.

Cassiopeia provides an effective general-case solution. However, multiple studies have pointed to its reliance on only mutation data from a single point in time, meaning that it needs to be adapted or modified for specialized domains. For instance, in the case of B cells, which can go through class switch recombination, a modified algorithm needs to be used[13]. In some cases, the problem is more fundamental. This work is a character-based phylogeny construction algorithm, which is only one of three general approaches to solving this problem, and in some applications this may not be the most effective solution. In the case of LinTIMaT[14], the authors found that a statistical approach worked better in the cases where the data from multiple experiments needed to be combined together.

Another area of study in expanding Cassiopeia also relies on expanding its reach if we have additional information. In the case of Moslin[10], they seek to make an algorithm for tracking cellular phylogenies where they have sampled cells over time during the experiment.

## References


[1] J. H. Camin and R. R. Sokal. A method for deducing branching sequences in phylogeny. *Evolution*, 19(3):311–326, 1965.

[2] L. L. Cavalli-Sforza and A. W. F. Edwards. Phylogenetic analysis: models and estimation procedures. *Evolution*, 21(3):550–570, 1967.

[3] M. M. et al. Chan. Molecular recording of mammalian embryogenesis. *Nature*, 570(7759):77–82, 2019.


[2] However, it should be noted that there was no mention of restrictions on the runtime of ILP.

[4] Michelle Chan-Seng-Yue, Jaeseung C. Kim, Gavin W. Wilson, Karen Ng, Eugenia Flores Figueroa, Grainne M. O'Kane, Ashton A. Connor, Robert E. Denroche, Robert C. Grant, Jessica McLeod, Julie M. Wilson, Gun Ho Jang, Amy Zhang, Anna Dodd, Sheng-Ben Liang, Ayelet Borgida, Dianne Chadwick, Sangeetha Kalimuthu, Ilinca Lungu, John M. S. Bartlett, Paul M. Krzyzanowski, Vandana Sandhu, Hervé Tiriac, Fieke E. M. Froeling, Joanna M. Karasinska, James T. Topham, Daniel J. Renouf, David F. Schaeffer, Steven J. M. Jones, Marco A. Marra, Janessa Laskin, Runjan Chetty, Lincoln D. Stein, George Zogopoulos, Benjamin Haibe-Kains, Peter J. Campbell, David A. Tuveson, Jennifer J. Knox, Sandra E. Fischer, Steven Gallinger, and Faiyaz Notta. Transcription phenotypes of pancreatic cancer are driven by genomic events during tumor evolution. *Nature Genetics*, 52(2):231–240, January 2020.

[5] W. M. Fitch and E. Margoliash. Construction of phylogenetic trees. *Science*, 155(3760):279–284, 1967.

[6] Dan Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21(1):19–28, 1991.

[7] Nicholas W. Hughes, Yuanhao Qu, Jiaqi Zhang, Weijing Tang, Justin Pierce, Chengkun Wang, Aditi Agrawal, Maurizio Morri, Norma Neff, Monte M. Winslow, Mengdi Wang, and Le Cong. Machine-learning-optimized cas12a barcoding enables the recovery of single-cell lineages and transcriptional profiles. *Molecular Cell*, 82(16):3103–3118.e8, August 2022.

[8] Matthew G. Jones, Alex Khodaverdian, Jeffrey J. Quinn, Michelle M. Chan, Jeffrey A. Hussmann, Robert Wang, Chenling Xu, Jonathan S. Weissman, and Nir Yosef. Inference of single-cell phylogenies from lineage tracing data using cassiopeia. *Genome Biology*, 21(1):92, 2020.

[9] L. Kester and A. van Oudenaarden. Single-cell transcriptomics meets lineage tracing. *Cell Stem Cell*, 23(2):166–179, 2018.

[10] Marius Lange, Zoe Piran, Michal Klein, Bastiaan Spanjaard, Dominik Klein, Jan Philipp Junker, Fabian J. Theis, and Mor Nitzan. Mapping lineage-traced cells across time points with moslin. *Genome Biology*, 25(1), October 2024.

[11] A. McKenna and J. A. Gagnon. Recording development with single cell dynamic lineage tracing. *Development*, 146(12), 2019.

[12] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.

[13] Leah L. Weber, Derek Reiman, Mrinmoy S. Roddur, Yuanyuan Qi, Mohammed El-Kebir, and Aly A. Khan. Isotype-aware inference of b cell clonal lineage trees from single-cell sequencing data. *Cell Genomics*, 4(9):100637, September 2024.

[14] Hamim Zafar, Chieh Lin, and Ziv Bar-Joseph. Single-cell lineage tracing by integrating crispr-cas9 mutations with transcriptomic data. *Nature Communications*, 11(1), June 2020.