ViralSim: Agent-Based Modeling of Viruses through Populations

Inferring Transmission Clusters and Finding Fitness

Benjamin Cai

June 12, 2025

LASA CS Independent Study

Transmission Clusters and Networks

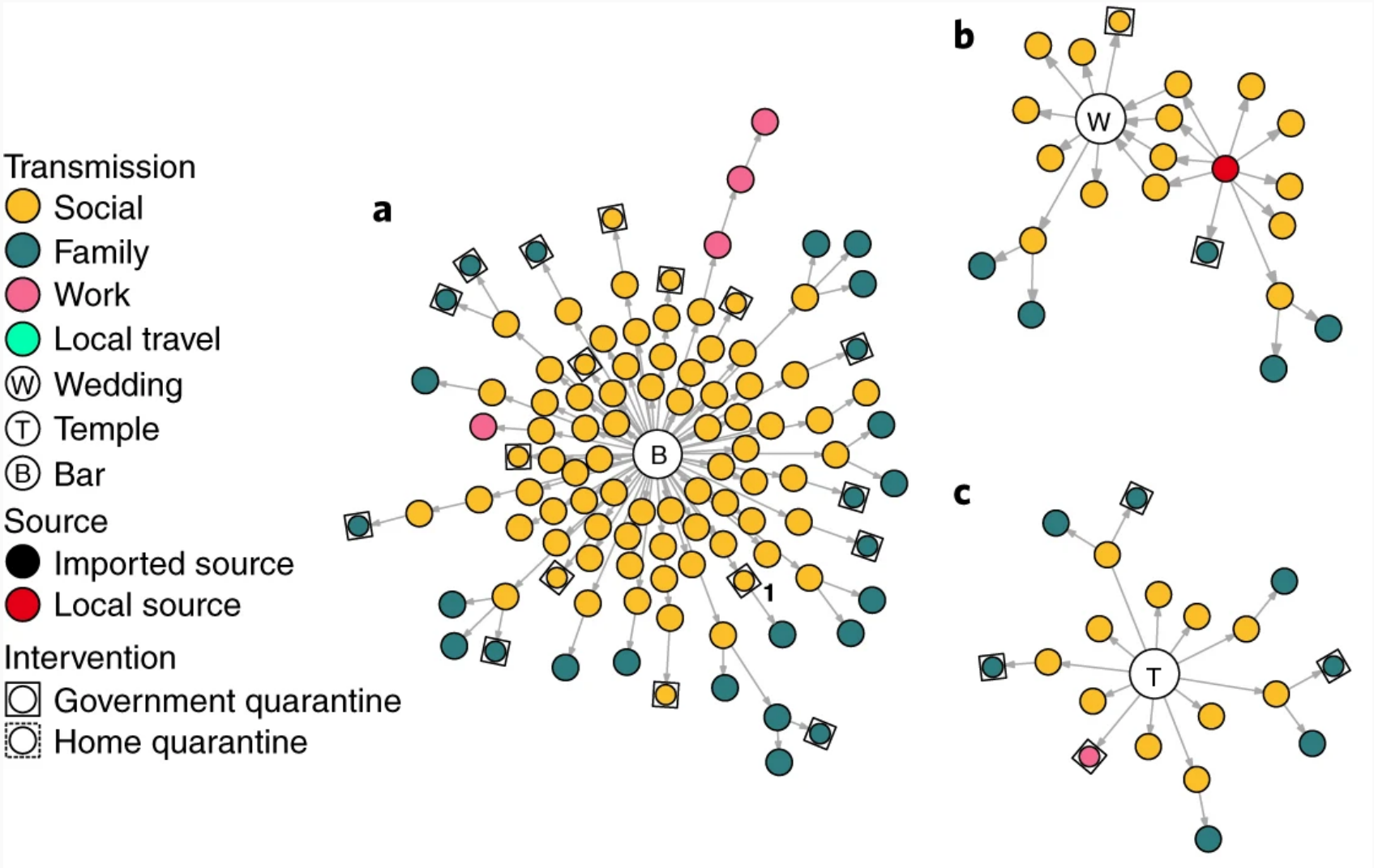


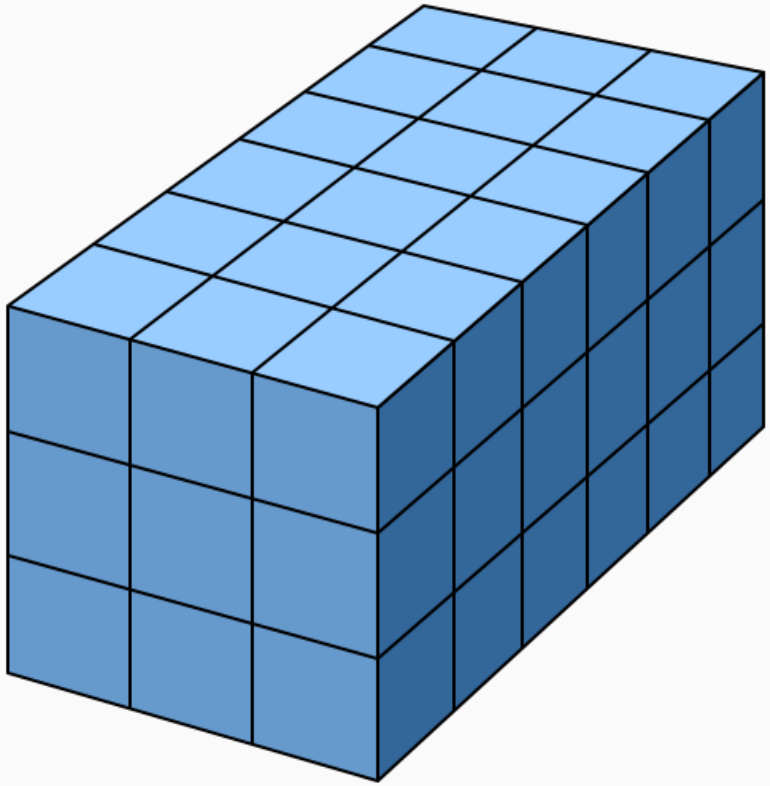
Figure 1: Hong Kong SARS-CoV-2 Clusters (Adam, et al.) [1]

Why Simulate?

1. See in real-time how viruses move and how pandemics start
2. Easily test different pandemic mitigation techniques
3. Compare analysis techniques (such as prediction) to what truly happened
4. Unique points for Agent-Based Models: Location and Fitness

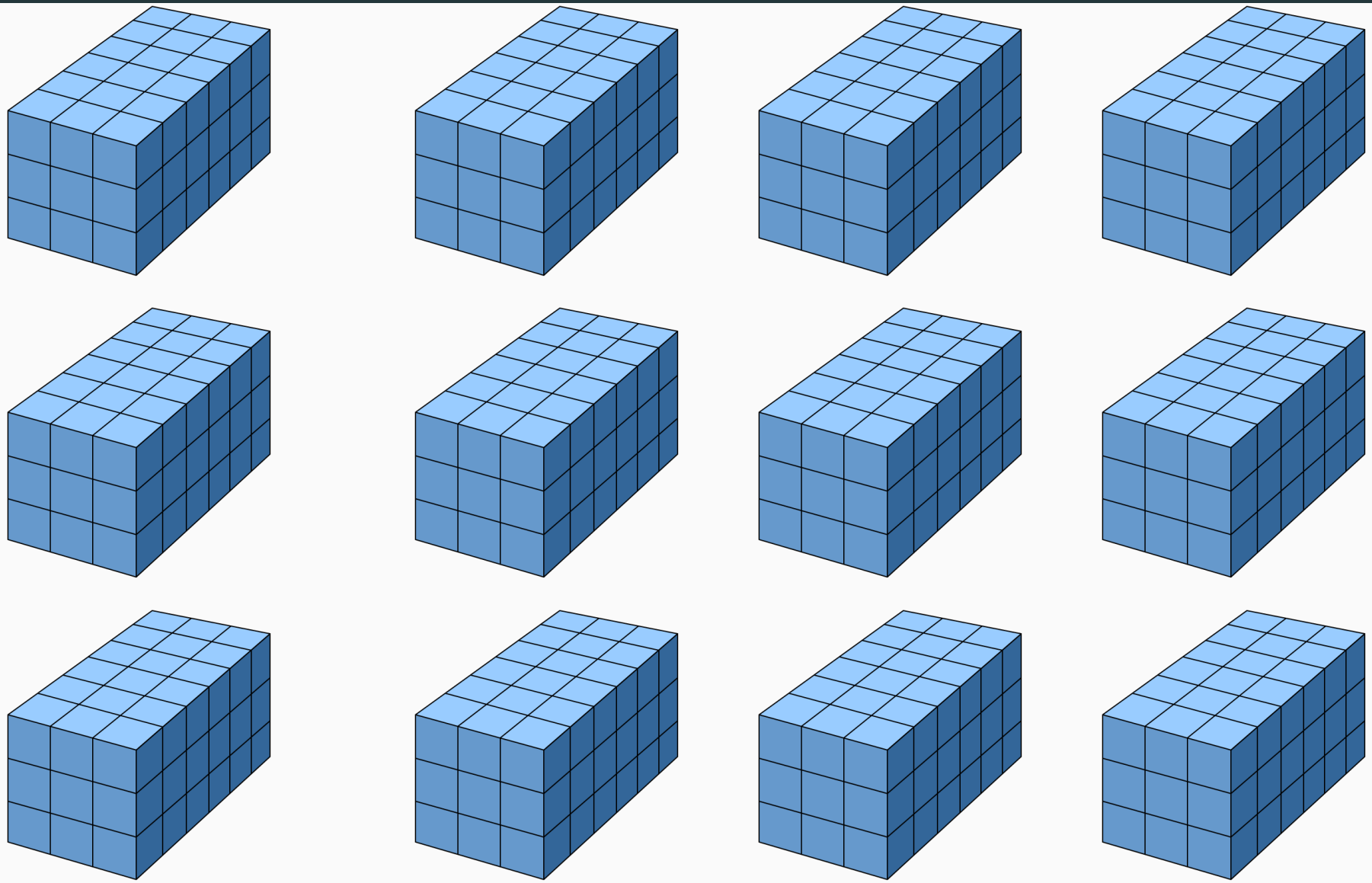
Simulation Overview

1. Hosts move around in locations (cities)
    - 1.1 Complex Movement
    - 1.2 List of Viruses
    - 1.3 List of Immunities
    - 1.4 Metadata like Age, gender, etc.



Example location that agents move in [2]

Movement between locations



Grid representing locations agents can move to [2]

Virus Overview

Seen Traits:

1. Sequence (A, T, C, G)
2. Infection Rate
3. Death and Recovery Rates
4. Asymptomatic Equivalents
5. Mutation Rate
6. Phylogeny (family only)

Unseen Traits:

1. Fitness

Figure 2: Fasta file

Mutation Overview

How Mutations Occur:

1. Chance for mutation if virus is within host
2. Immunity can be bypassed after X amount of mutations

Mutation Types:

1. Substitutions (more complex)
2. Indels (constant rate)
3. Recombination (constant rate)

Mutation Effects

```
void Virus::normalMutation() {
  infectionRate += Random::getRandomFloat(-0.01, 0.01);
  if (infectionRate > 0.9) {
    infectionRate -= Random::getRandomFloat(0,infectionRate/10);
  }
  deathRate += Random::getRandomFloat(-0.01, 0.01);
  if (deathRate < 0.1) {
    deathRate += Random::getRandomFloat(0, (1-deathRate)/10);
  }
  passthroughRate += Random::getRandomFloat(-deathRate/15, deathRate/15);
}
```

Figure 3: Example Mutation Code

Model for Substitution (Felsenstein 1981 Model)

Reasons:

1. Nucleotides change at different rates
2. Purines -> Pyrimidines vs Purine -> Purine

$$Q = \begin{pmatrix} & A & C & G & T \\ A & - & \pi_C & k\pi_G & \pi_T \\ C & \pi_A & - & \pi_G & k\pi_T \\ G & k\pi_A & \pi_C & - & \pi_T \\ T & \pi_A & k\pi_C & \pi_G & - \end{pmatrix}$$

A -> C is represented by π_C

'-' is the sum of all row values

Inputs and Outputs for ViralSim

Inputs:

1. Commands

Outputs:

1. Sequences
2. Sequence Metadata (namely location + traits)
3. True phylogeny

```
0 if (noVirus, addVirus 0 0 0.5 0.1 0.1 0.8)
50 AddVirus 0 0 0.5 0.1 0.1 0.8
60 AddLocation 0 10 10 10 100
300 AddVirus 0 0 0.5 0.1 0.1 0.8
600 AddVirus 0 0 0.5 0.1 0.1 0.8
```

Figure 4: Example commands with conditionals and times

Sequence Name	Location	Time	Age	AgentID	InfectionRate	DeathRate	PassthroughRate	MutationRate
V-JUGW-	L-IMWE	0	NA	NA	0.5	0.01	0.01	0.5
V-JUGW-M1	L-IMWE	2	2	56	0.494247	0.012237	0.010171	0.5
V-JUGW-M2	L-IMWE	3	3	4	0.491158	0.08571	0.006137	0.5
V-JUGW-M1M1	L-IMWE	3	3	56	0.494319	0.096996	0.013554	0.5
V-JUGW-M3	L-IMWE	4	4	4	0.498306	0.054547	0.007586	0.5
V-JUGW-M2M1	L-IMWE	4	4	4	0.495393	0.121319	0.00294	0.5
V-JUGW-M4	L-IMWE	4	4	56	0.509396	0.067196	0.012236	0.5
V-JUGW-M1M2	L-IMWE	4	4	56	0.498562	0.097253	0.009484	0.5
V-JUGW-M1M1M	L-IMWE	4	4	56	0.494079	0.103677	0.010205	0.5

Figure 5: Example output csv file

```
  commandMap = {
    {"AddVirus", addVirusCommand},
    {"AddLocation", addLocationCommand},
    {"if", ifCommand},
    {"Masking", masking},
    {"Quarantine", quarantine}
  };
```

Figure 7: How Commands are Read

```
void addNodeLeaf(const std::string root,
                 const std::string parent,
                 const std::string child,
                 const int timeElapsed) {
  std::string& phylogeny = phylogenies[root];
  size_t pos = phylogeny.find(parent);
  // DO NOT EVER IMPLEMENT CHECKS LIKE std::string::npos
  // It must be kept this way to find errors, otherwise they pass
  if (pos > 0 && phylogeny[pos - 1] == ')') {
    std::string toReplace = ")" + parent;
    std::string replacement = ", " + child + ":"
                            + std::to_string(timeElapsed)
                            + ")" + parent;
    size_t replacePos = phylogeny.find(toReplace, pos - 1);
    phylogeny.replace(replacePos, toReplace.length(), replacement);
  }
  else {
    std::string toReplace = parent;
    std::string replacement = "(" + child + ":"
                            + std::to_string(timeElapsed) + ")" + parent;
    phylogeny.replace(pos, toReplace.length(), replacement);
  }
}
```

Figure 6: Phylogeny as String

```
for (auto& virus : viruses) {
  if ((size(viruses) < 10)
      && (Random::getRandomFloat(0, 1) < virus.mutationRate)) {
    //std::cout << virus.toString(0) << std::endl;
    if (Random::getRandomFloat(0, 1) < 0.98) {
      Virus v = Virus(virus);
      v.mutate(step, location);
      FileWriting::writeMetadata(v.root, v.name, this->location,
                                 std::to_string(step),
                                 std::to_string(this->age),
                                 std::to_string(this->id),
                                 std::to_string(v.infectionRate),
                                 std::to_string(v.deathRate),
                                 std::to_string(v.passthroughRate),
                                 std::to_string(v.mutationRate));
      toAdd.push_back(v);
      if (Random::getRandomFloat(0,1) < 0.5) {
        auto virusIn = std::find_if(viruses.begin(), viruses.end(),
                                    [&virus](const Virus& v) {
                                        return virus.name == v.name;
                                    });
        toRemove.push_back(*virusIn);
      }
    }
    else {
      Virus v = Virus(virus);
      v.recombination(step, location,
                      viruses[Random::getRandomInt(0, viruses.size())].sequence);
```

Figure 8: Virus Calculations

Tree

Figure 9: TREE [4]

Tree (Phylogeny)

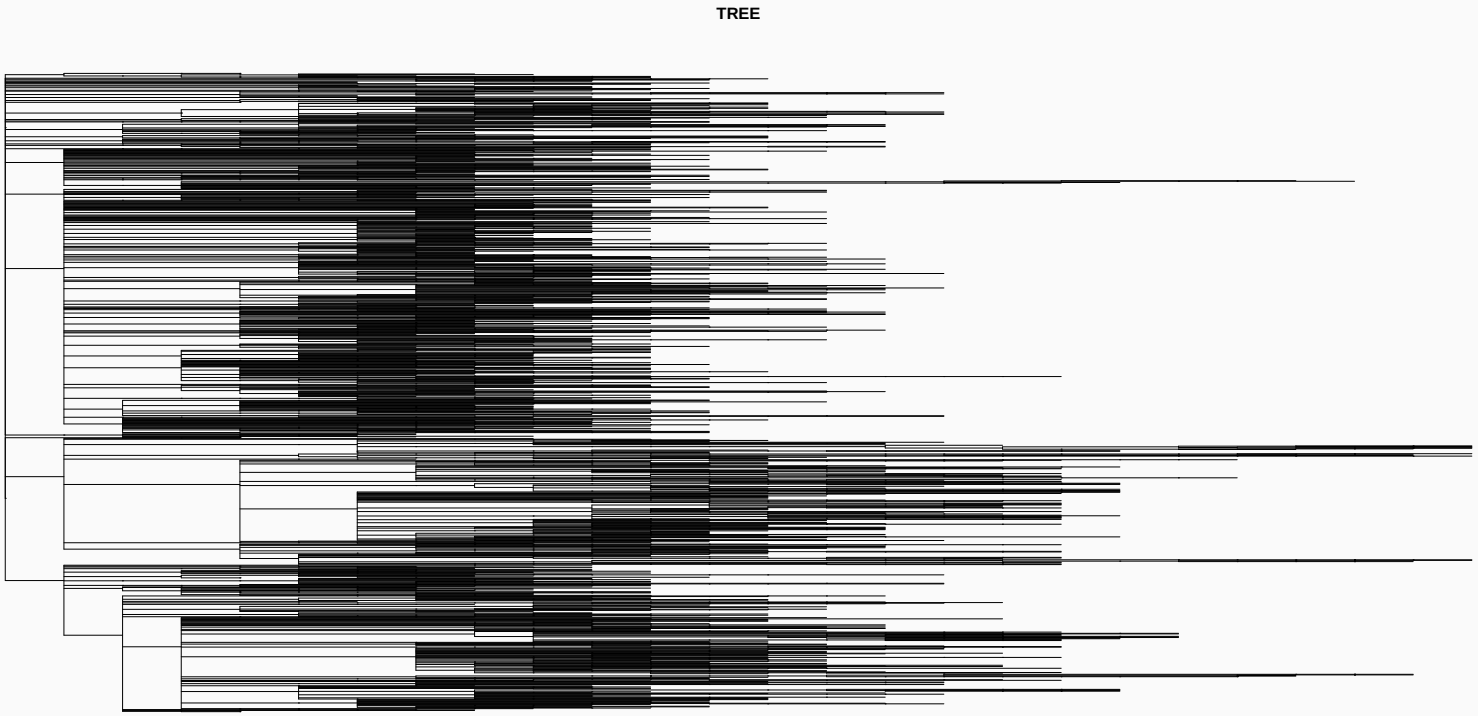


Figure 10: Example 1

Tree (Phylogeny)

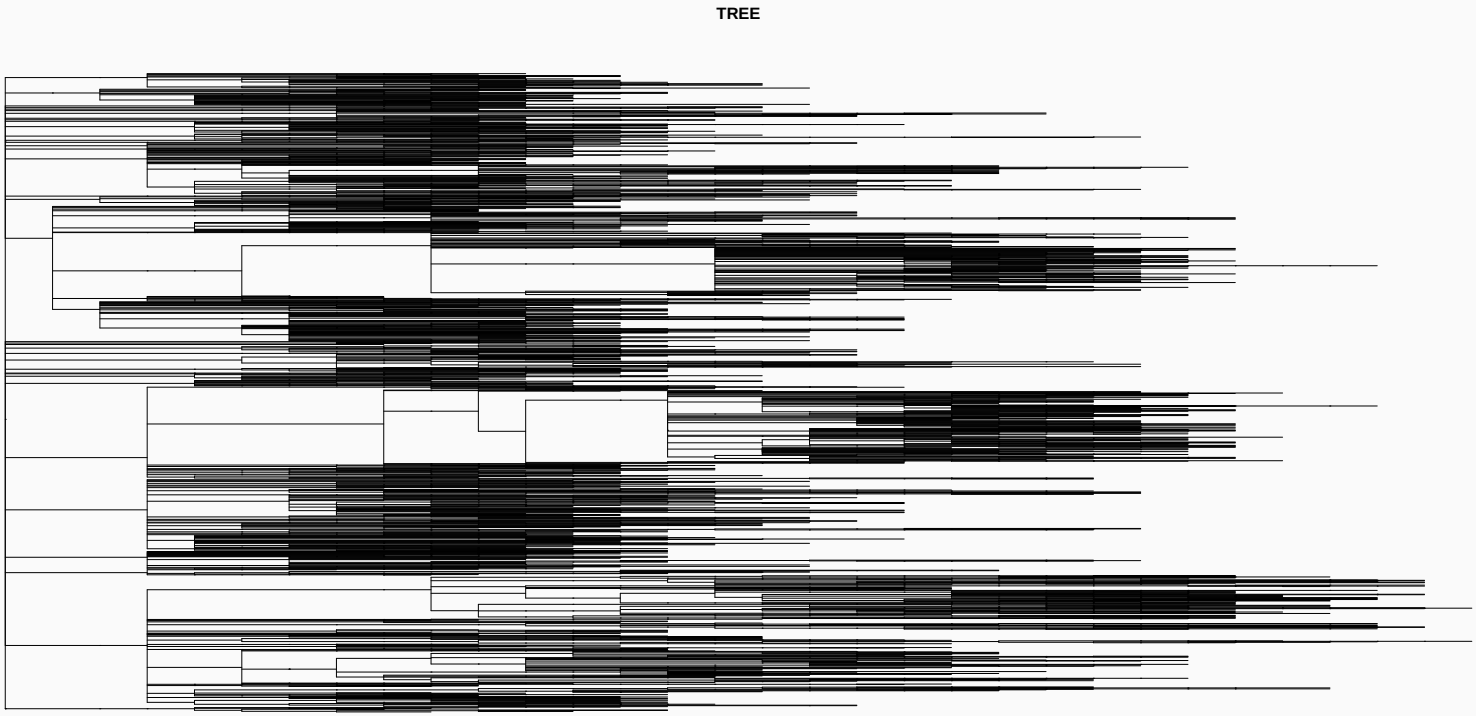


Figure 11: Example 2

Tree (Phylogeny)

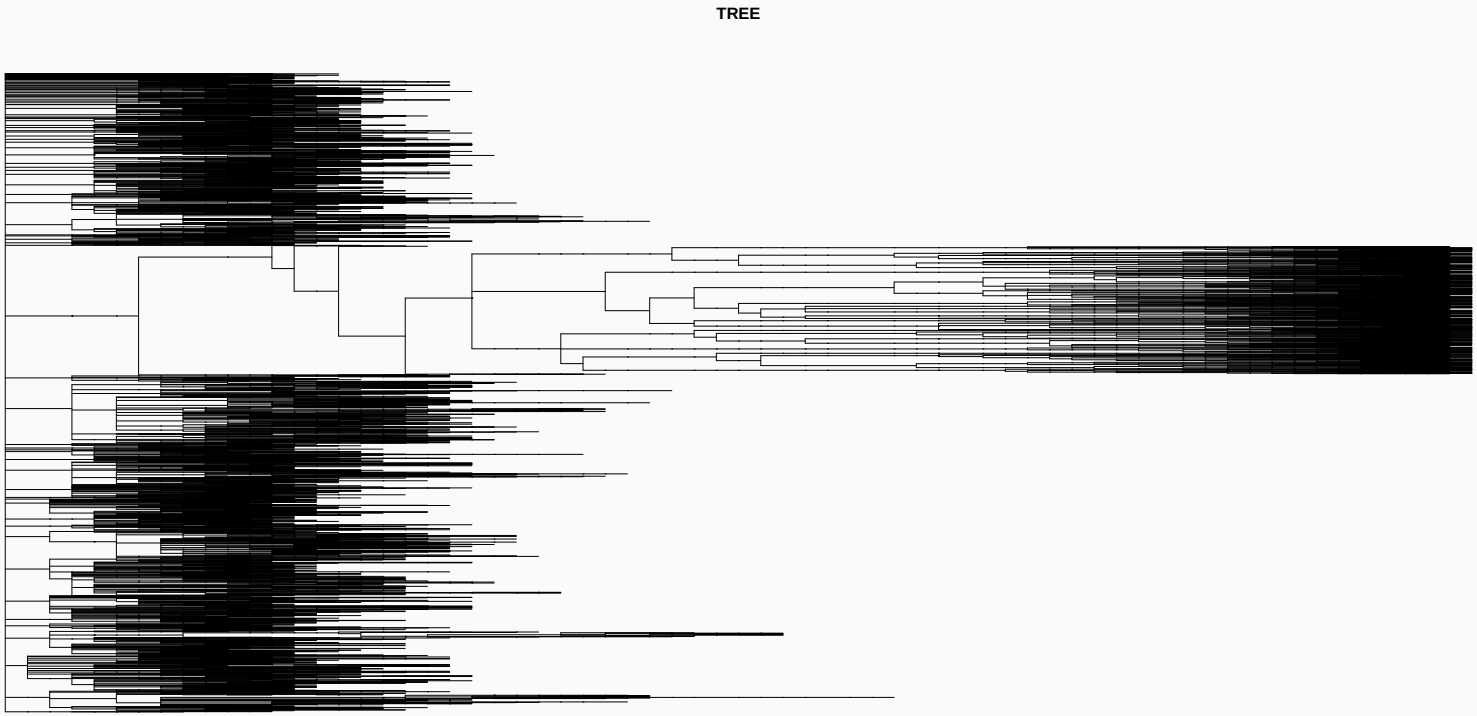


Figure 12: Example 3

Transmission Clusters and Networks

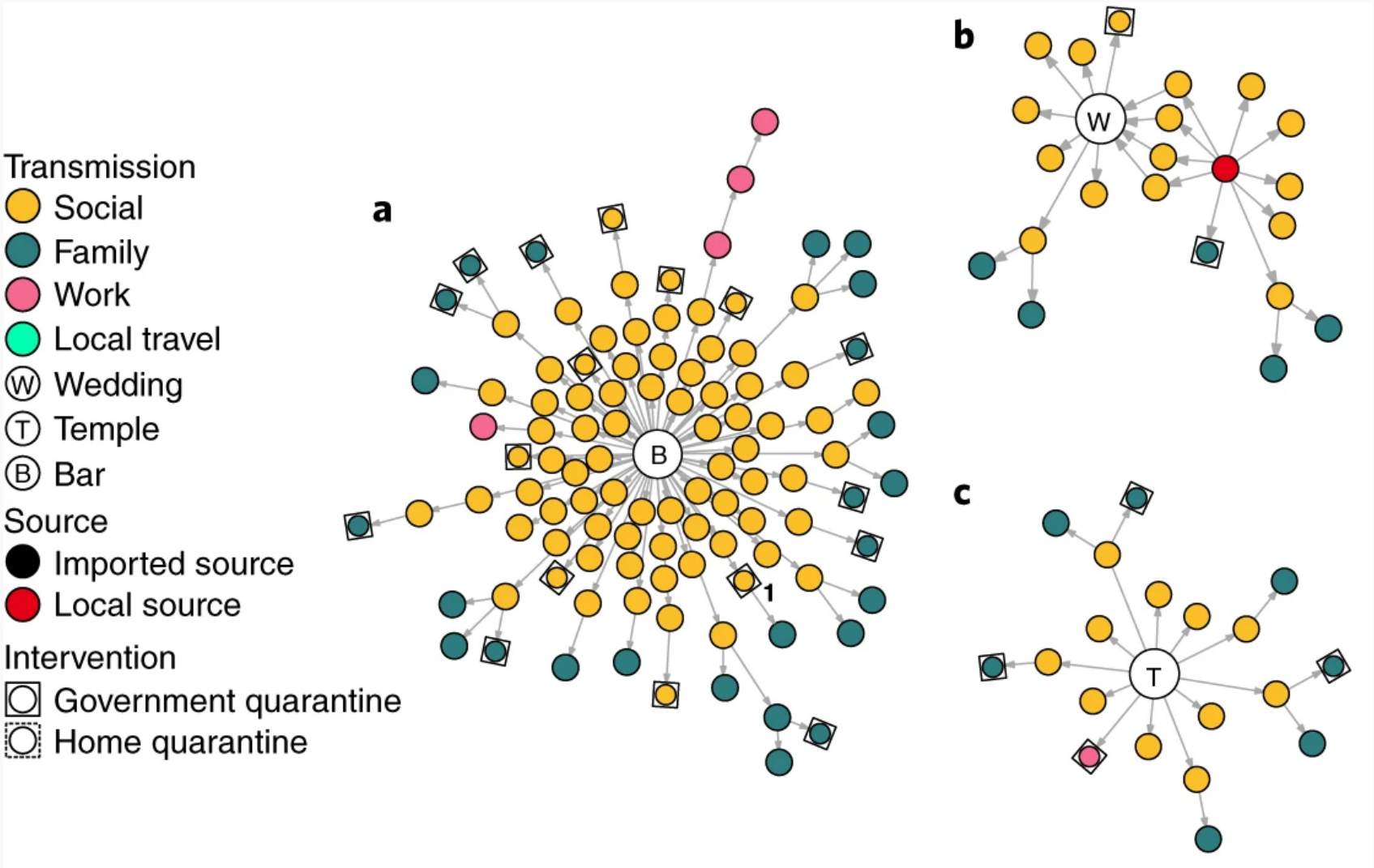


Figure 13: Hong Kong SARS-CoV-2 Clusters (Adam, et al.) [1]

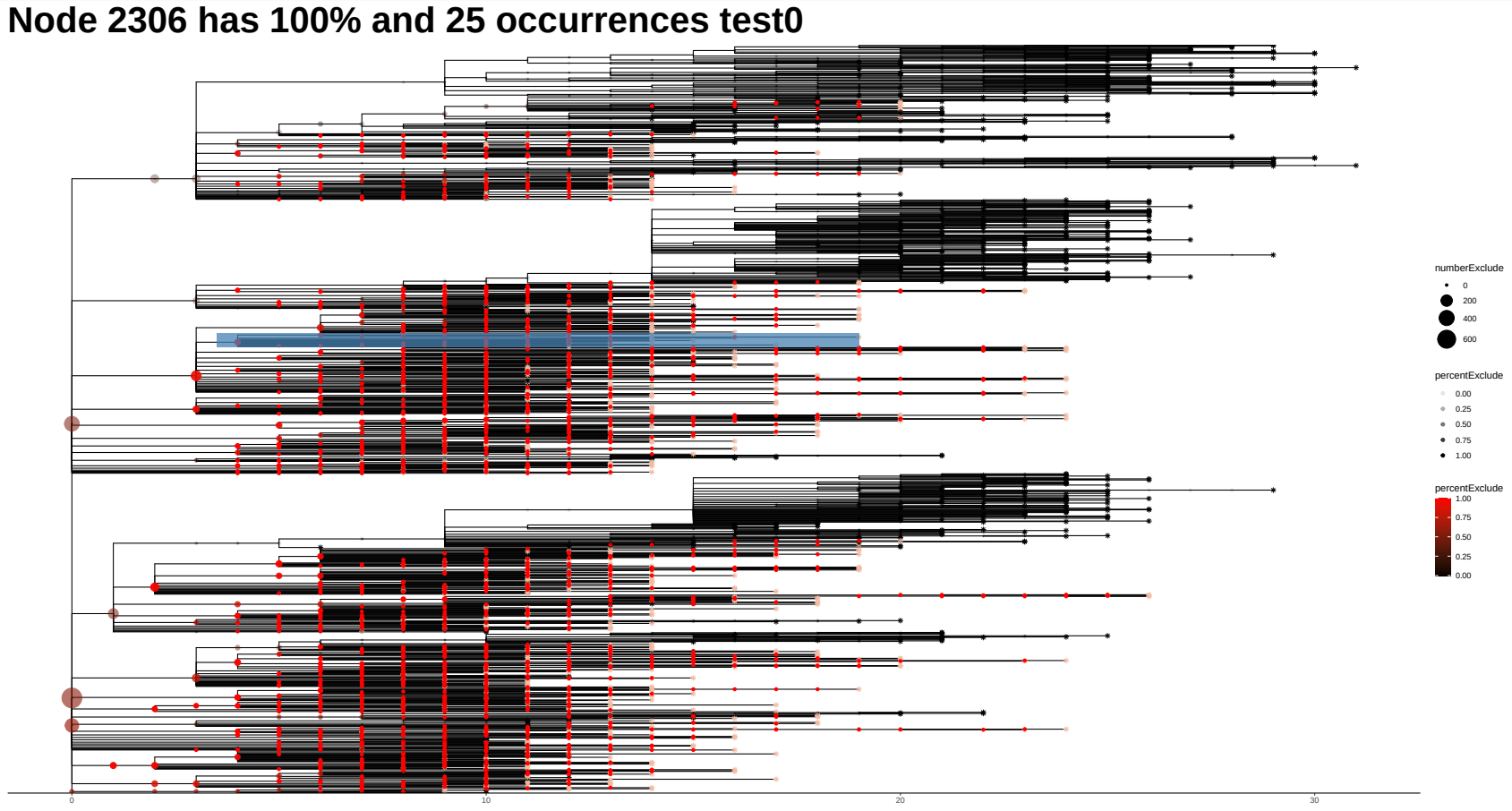
How to find Transmission Clusters

1. Start at a parent with children
2. Find the ratio of number of children from location A and the number of children not from location A
3. Check with a threshold (like 100:1) and repeat for all parent nodes

```
orderTransmissionClusters <- function(TC, limit, number, percent) { ## Returns data necessary for node zoom in
  TC <- TC[order(TC$percentExclude, TC$numberExclude, decreasing = TRUE),] ## Order TC by percent
  endList <- data.frame(node = c(), percent = c(), number = c())
  nodes <- vector("list", length = 10)
  percents <- vector("list", length = 10)
  numbers <- vector("list", length = 10)
  count <- 1
  for(i in 1:nrow(TC)) {
    row <- TC[i,]
    if (row$numberExclude > number && row$percentExclude > percent){ ## For each row, if there are more strains than 7, keep it
      nodes[count] <- row$node
      percents[count] <- row$percentExclude
      numbers[count] <- row$numberExclude
      count <- count + 1
    }
    if (count > limit){ ## Once reach limit, exit loop
      break
    }
  }
  endList <- data.frame(node = unlist(nodes), percent = unlist(percents), number = unlist(numbers))
  return(endList) ## Return as dataframe
}
```

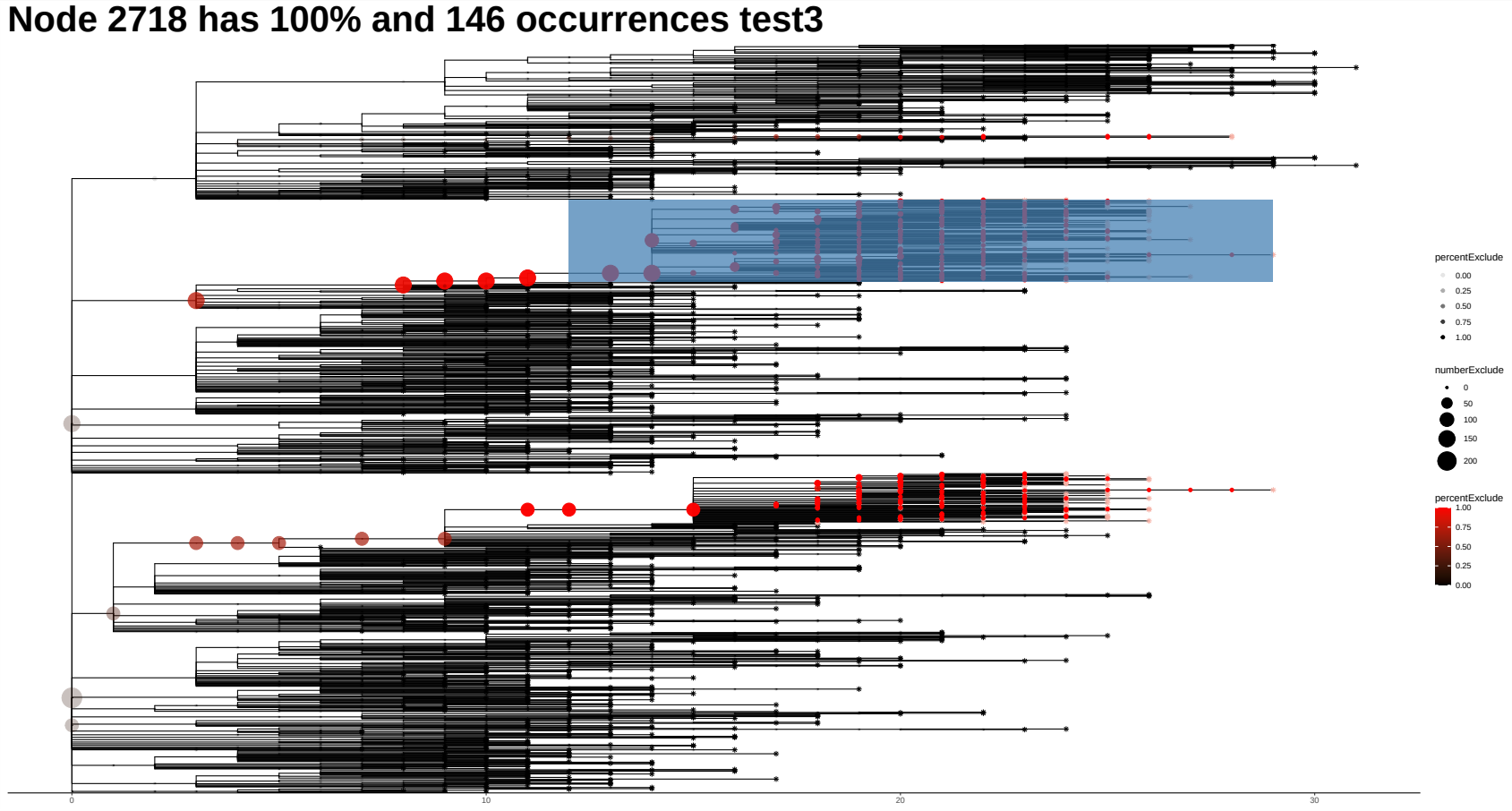
Figure 14: Main Component of TC Calculation

Tree (Phylogeny)



Simulation exported tree annotated for location ’test0’

Tree (Phylogeny)



Simulation exported tree annotated for location ’test3’

Main Test

1. Real-World Tree-Building vs True Tree

How Tree-Building Starts

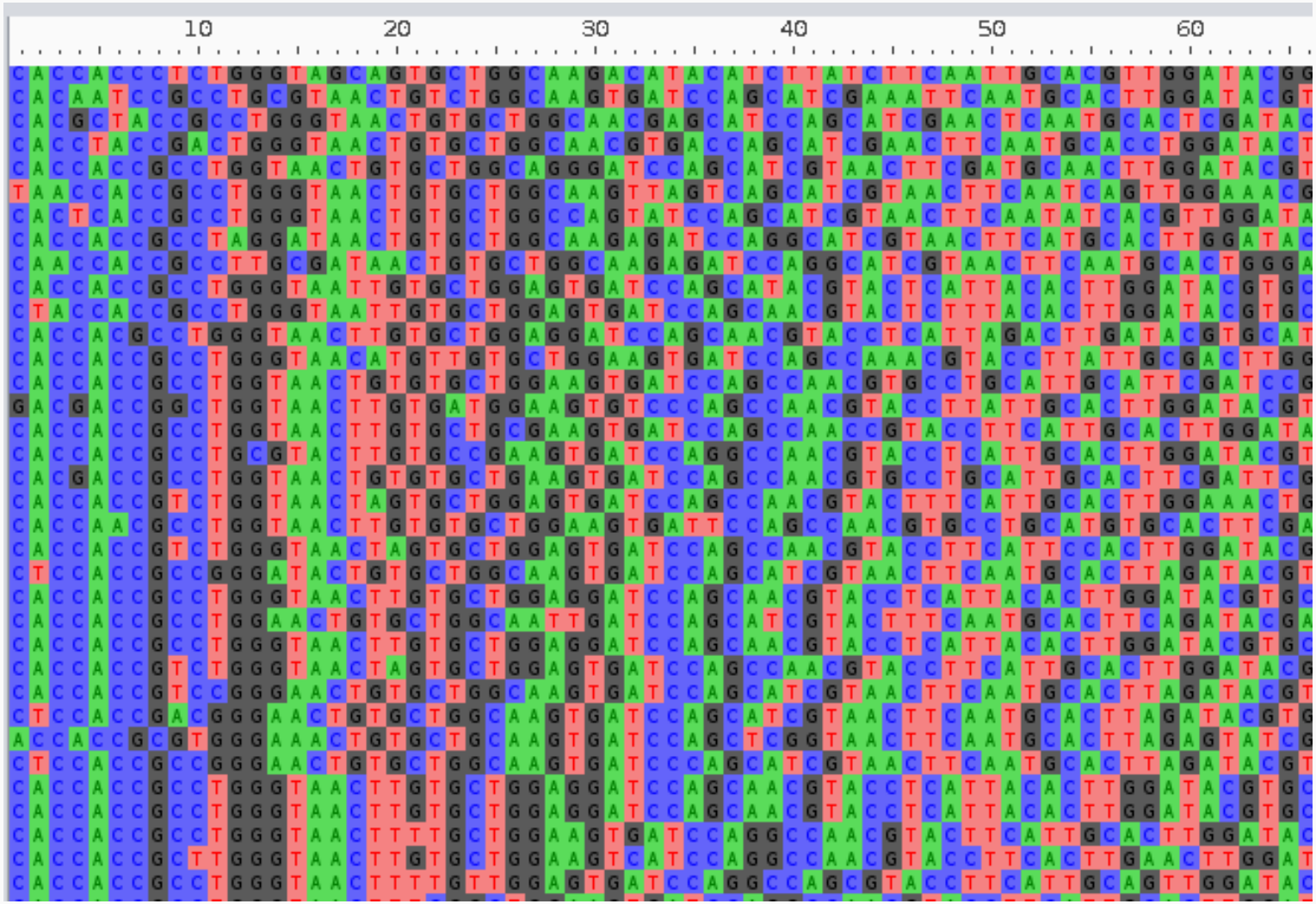
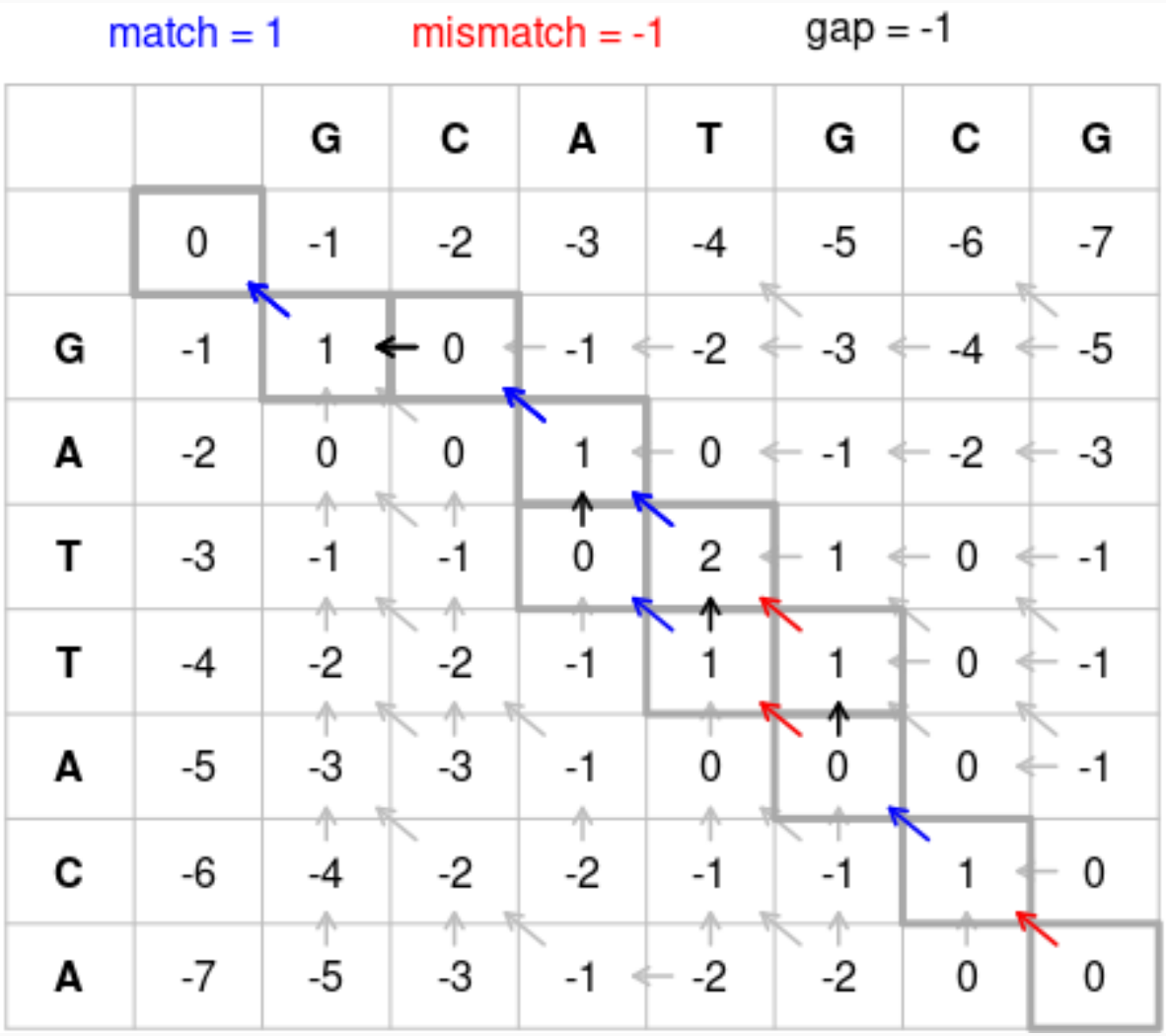


Figure 15: Simulated Sequences in ALIVIEW

Tree-Building: Sequence Alignment



		G	C	A	T	G	C	G
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	1	0	-1	-2	-3	-4	-5
A	-2	0	0	1	0	-1	-2	-3
T	-3	-1	-1	0	2	1	0	-1
T	-4	-2	-2	-1	1	1	0	-1
A	-5	-3	-3	-1	0	0	0	-1
C	-6	-4	-2	-2	-1	-1	1	0
A	-7	-5	-3	-1	-2	-2	0	0

Figure 16: Needleman-Wunsch Pairwise Alignment (Wikipedia) [3]

Tree-Building: Maximum Parsimony

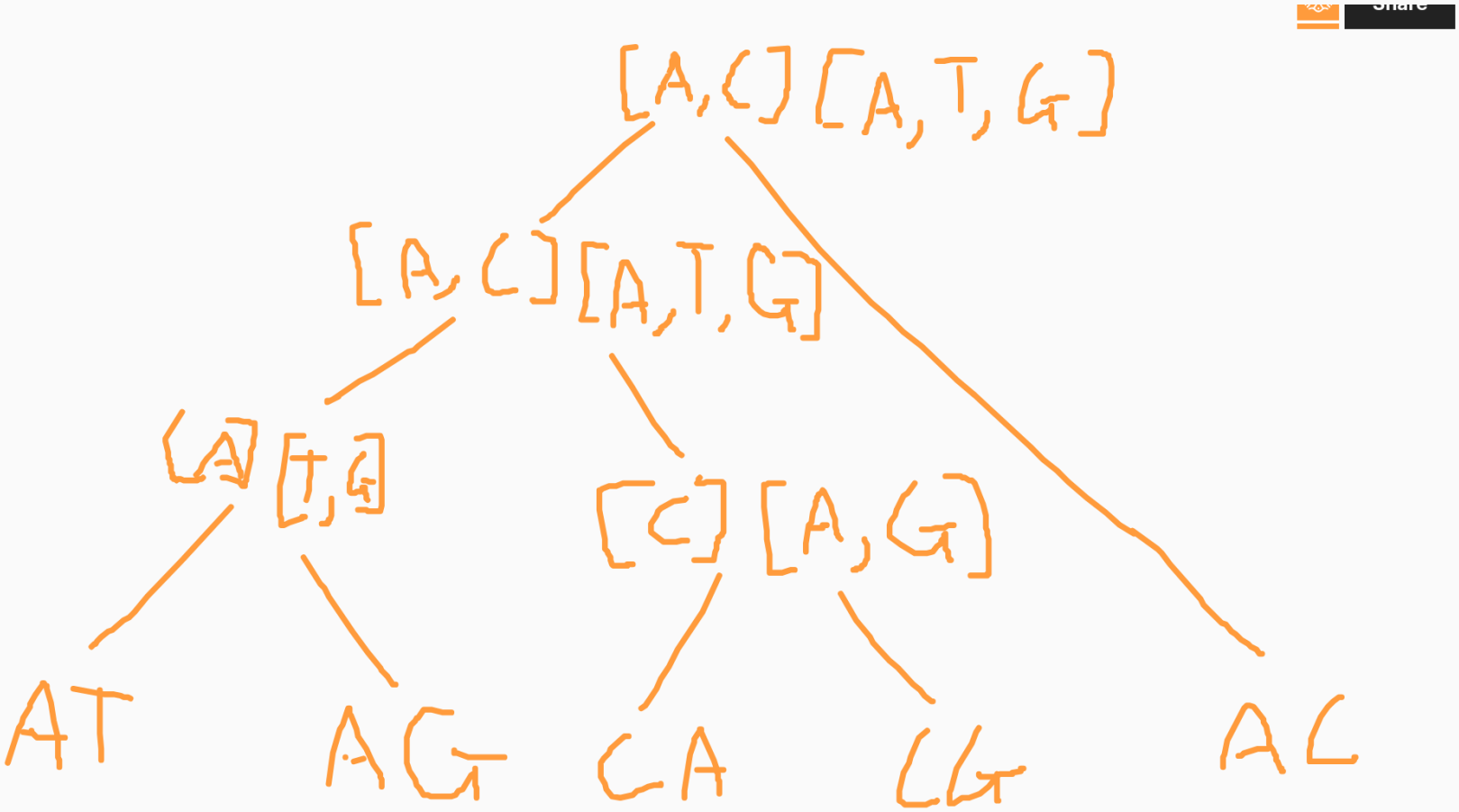
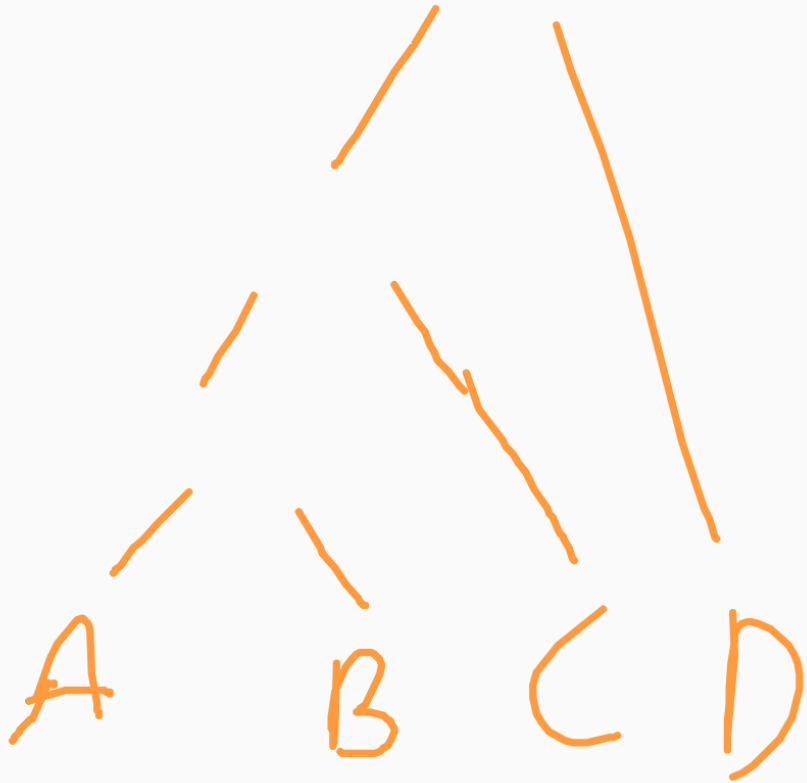


Figure 17: Example of Maximum Parsimony Calculation

Tree-Building: Maximum Likelihood

1. V(i,x) is likelihood of subtree rooted at i when parent takes value x
2. Maximize likelihood
3. Change tree topology



Tree example

SARS-CoV-2 Tree annotated for California

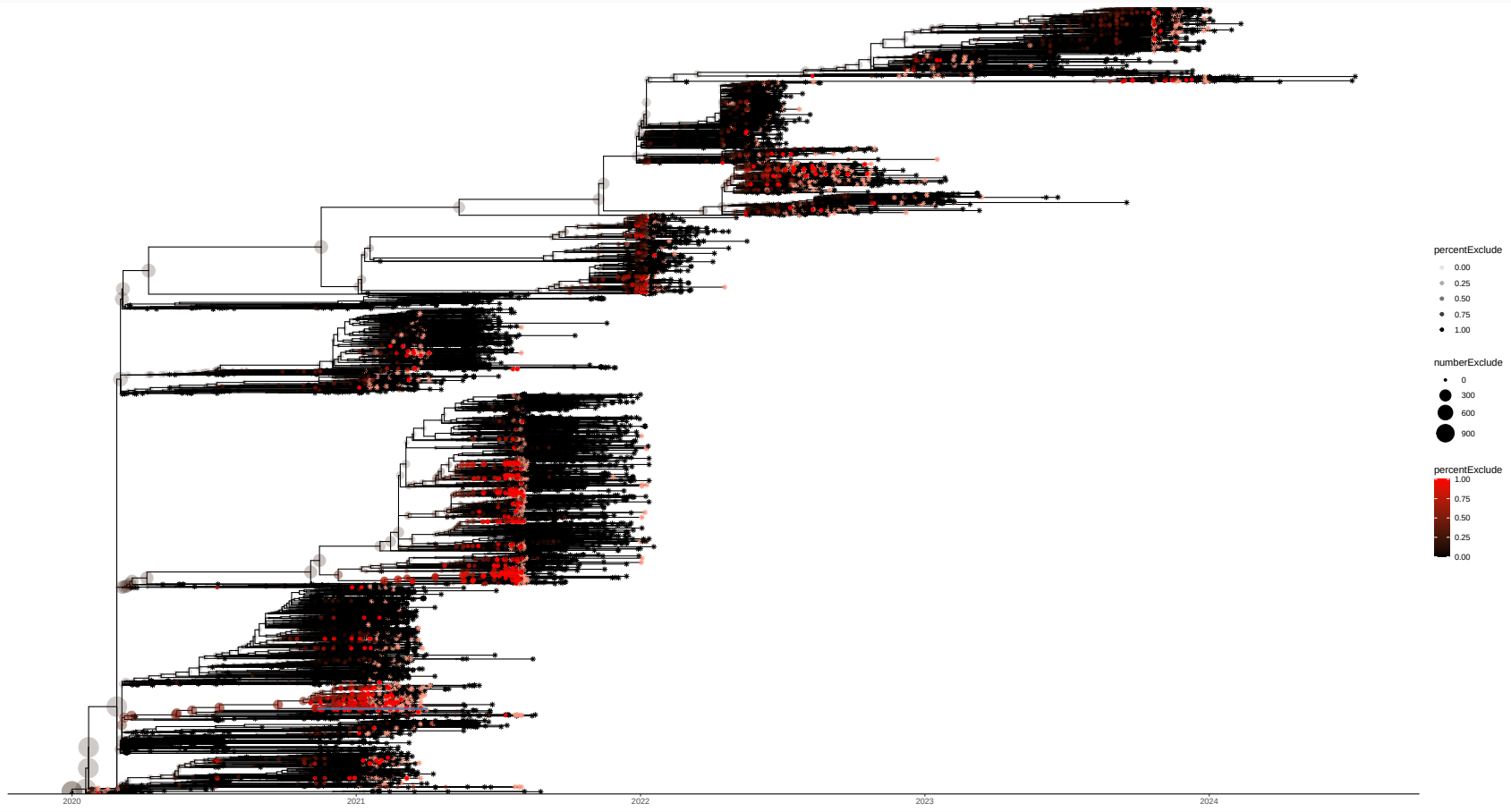


Figure 18: US SARS-CoV-2 phylogeny

California Largest Transmission Cluster

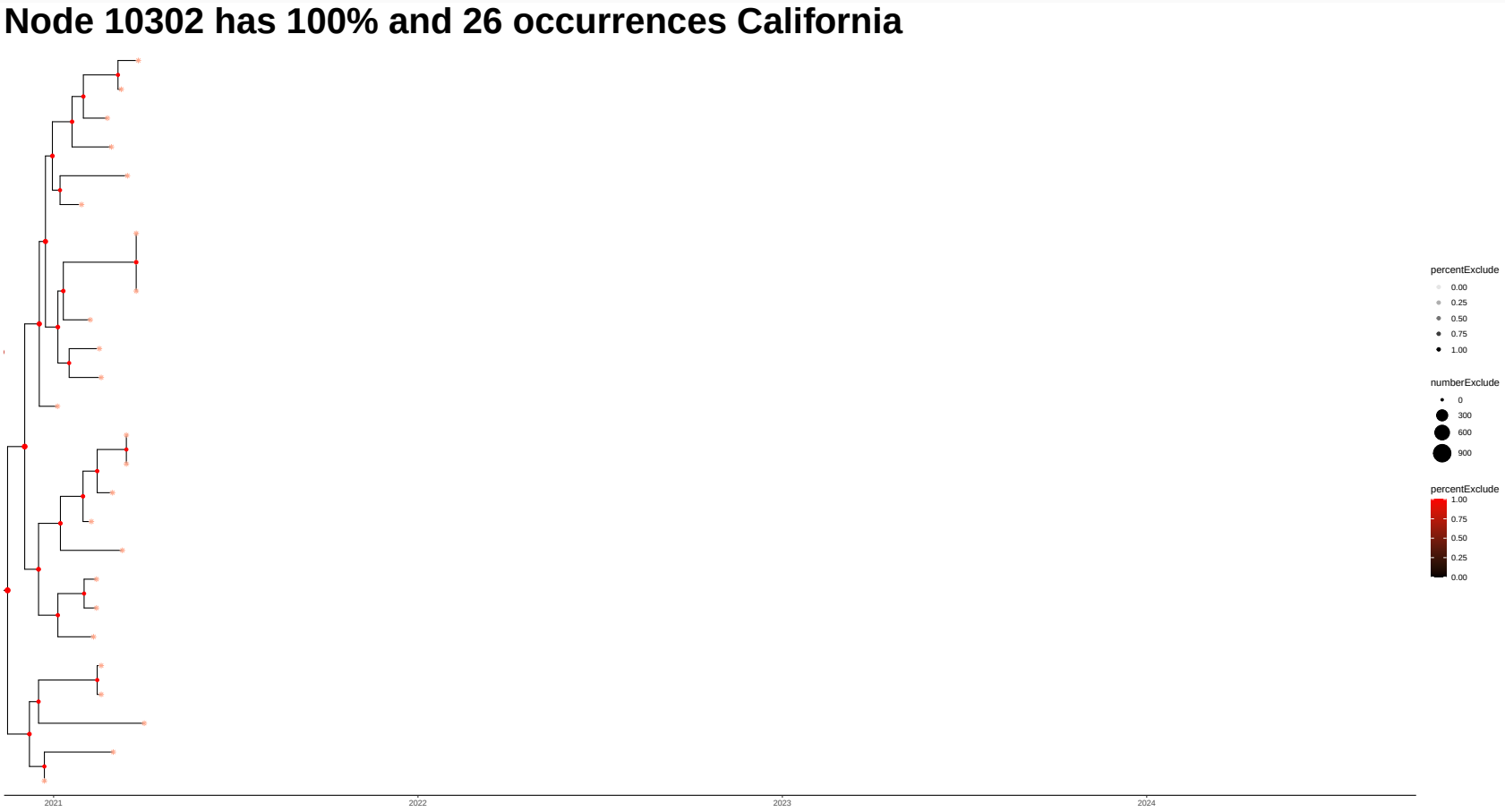


Figure 19: Largest Cluster, 26 occurrences, 100%

Transmission Clusters

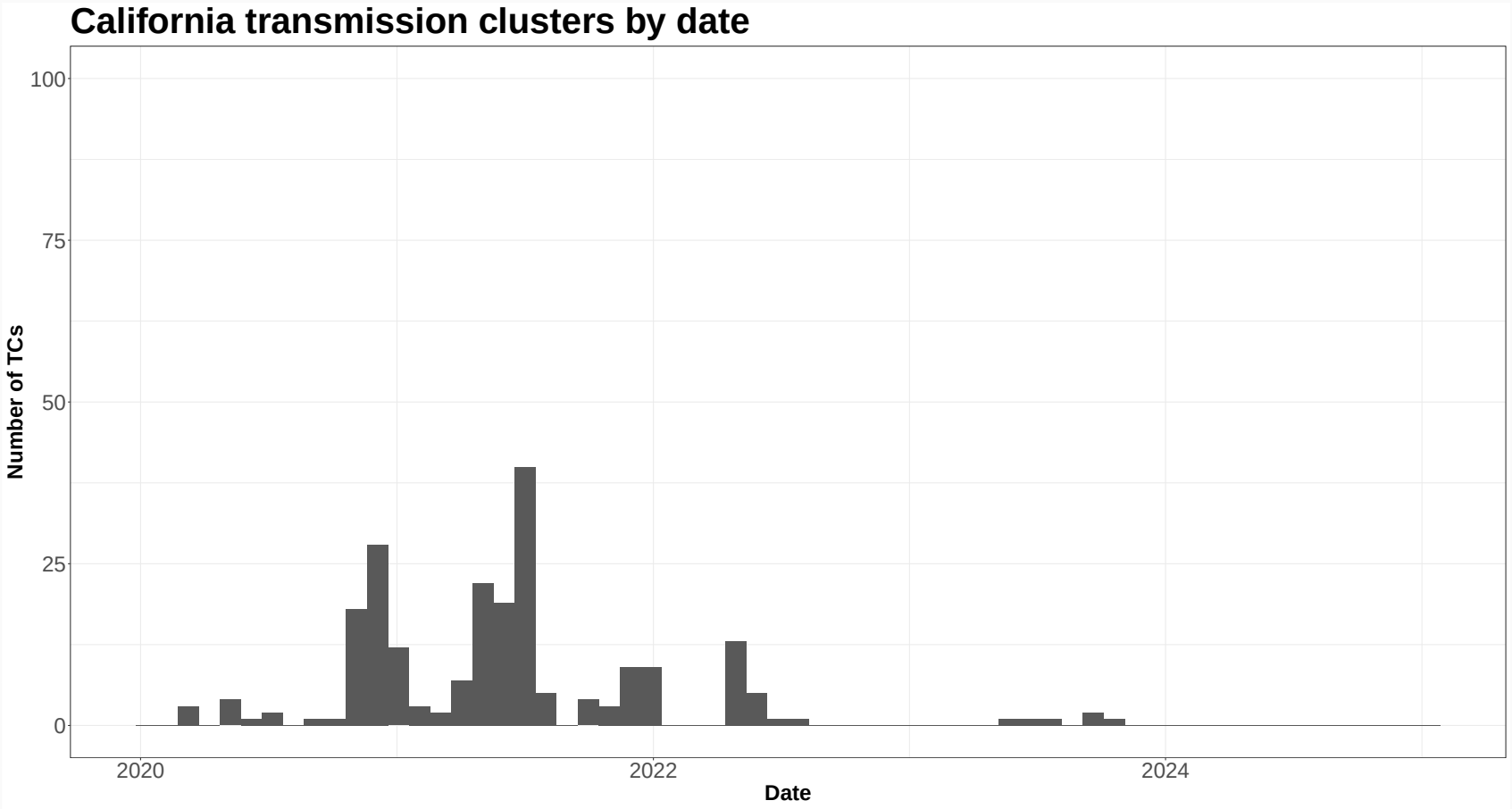
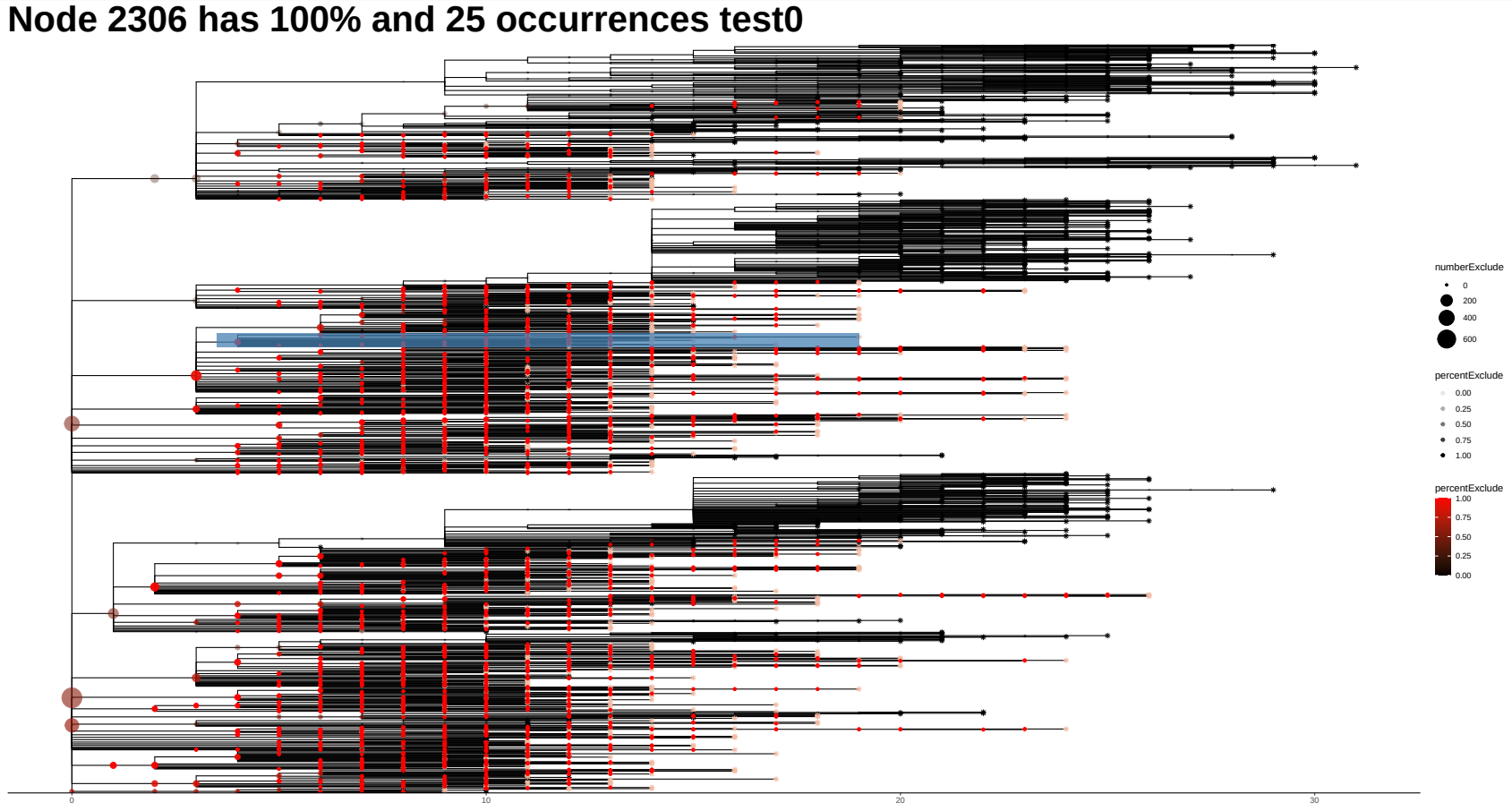


Figure 20: California Transmission Clusters, (30% cutoff with 5 sequences necessary)

Benefits (or cons) Compared to Real-World Data

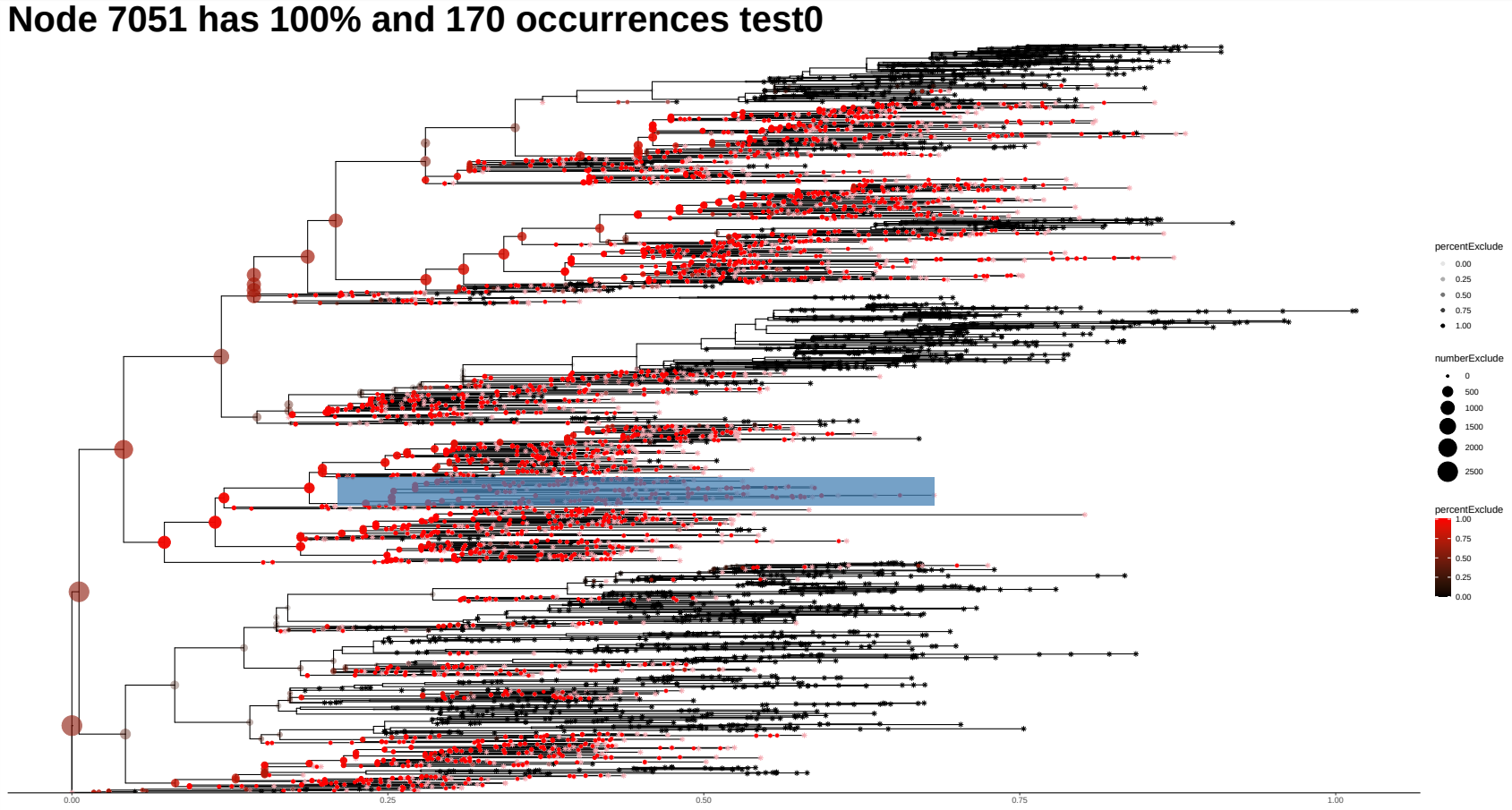
1. Proportional Sequences
2. No Bias

Tree (True)



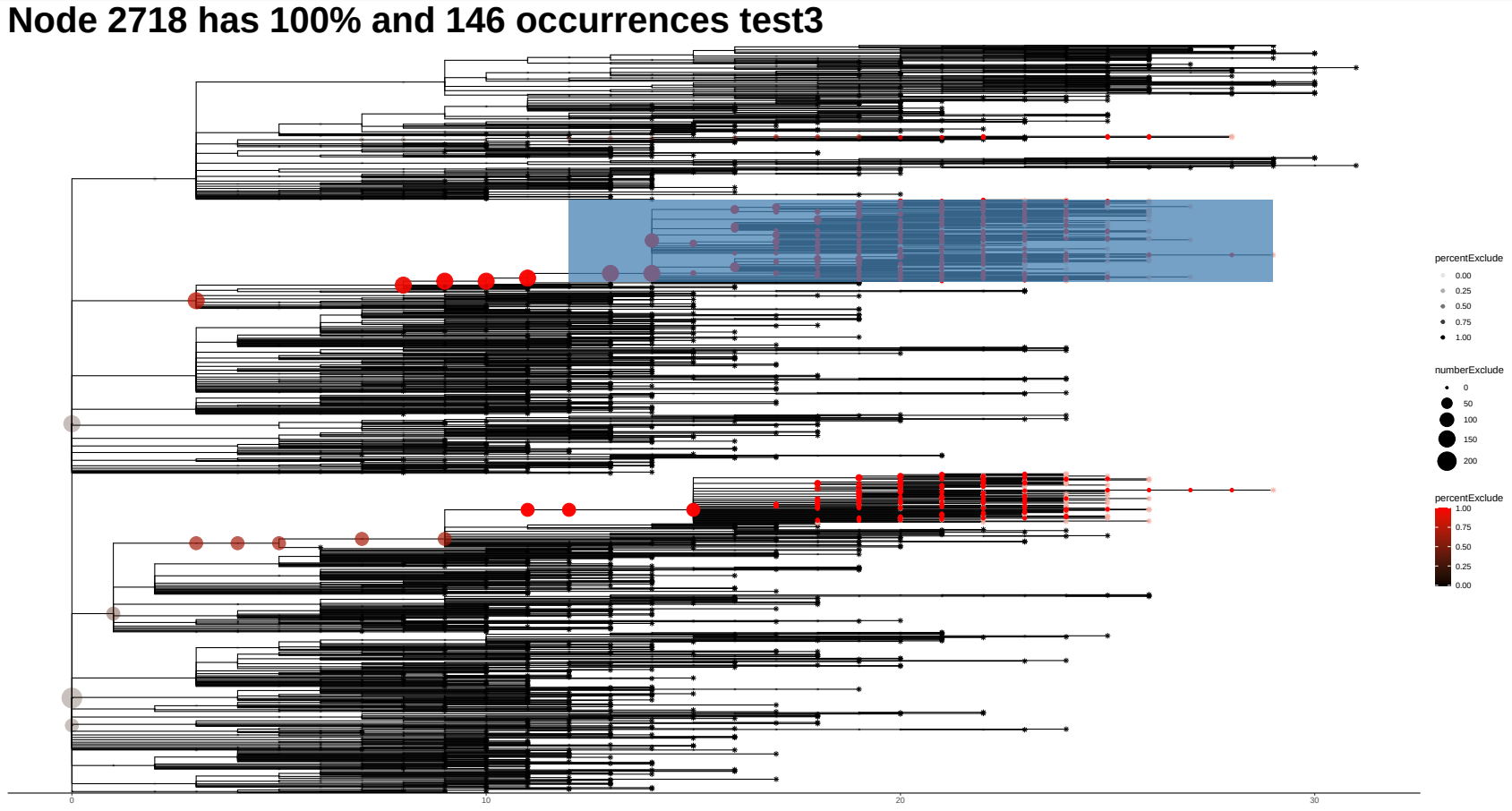
Simulation exported tree annotated for location ’test0’

Tree (ML Estimated)



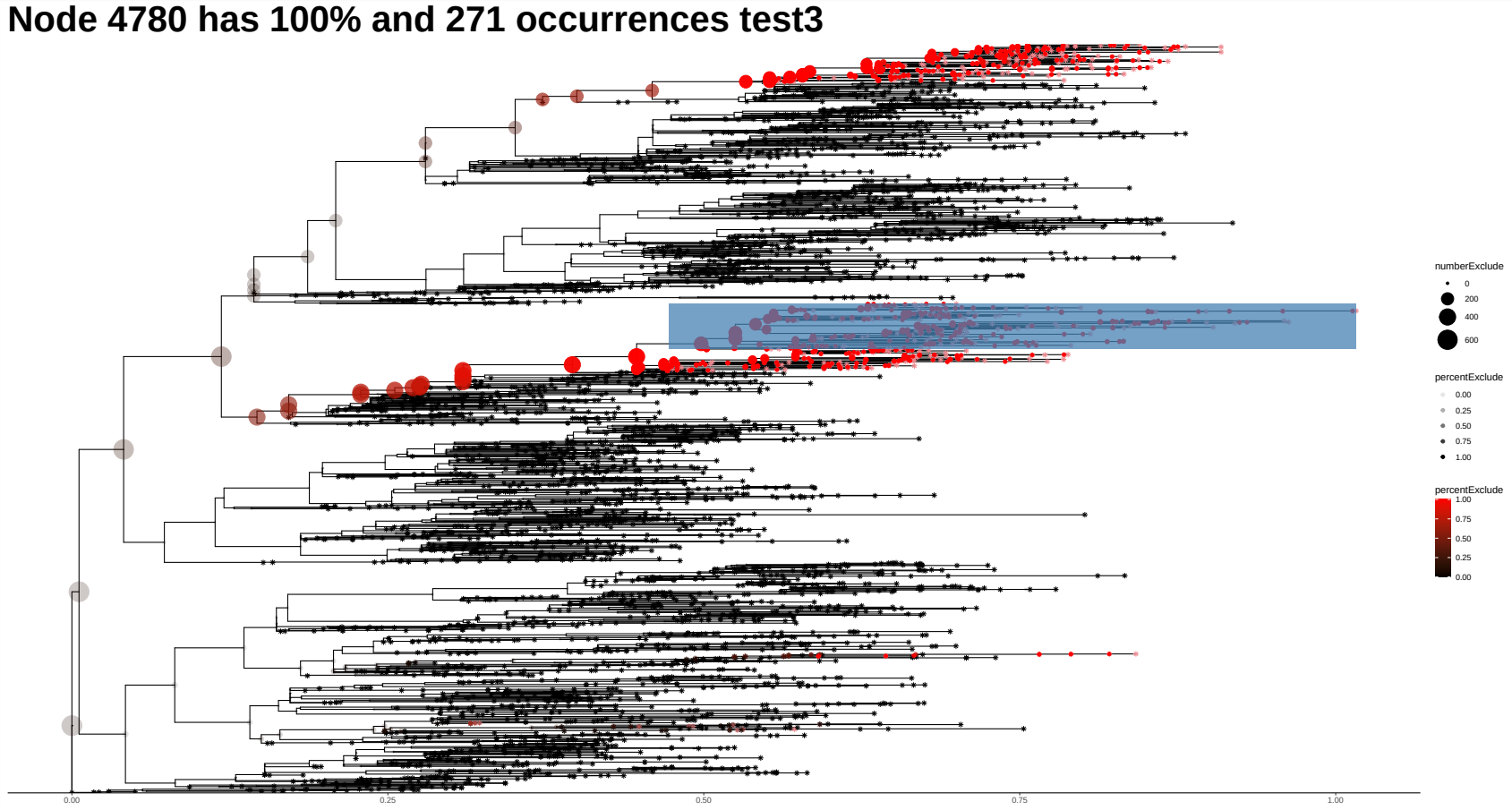
Tree calculated with maximum likelihood annotated for location ’test0’

Tree (True)



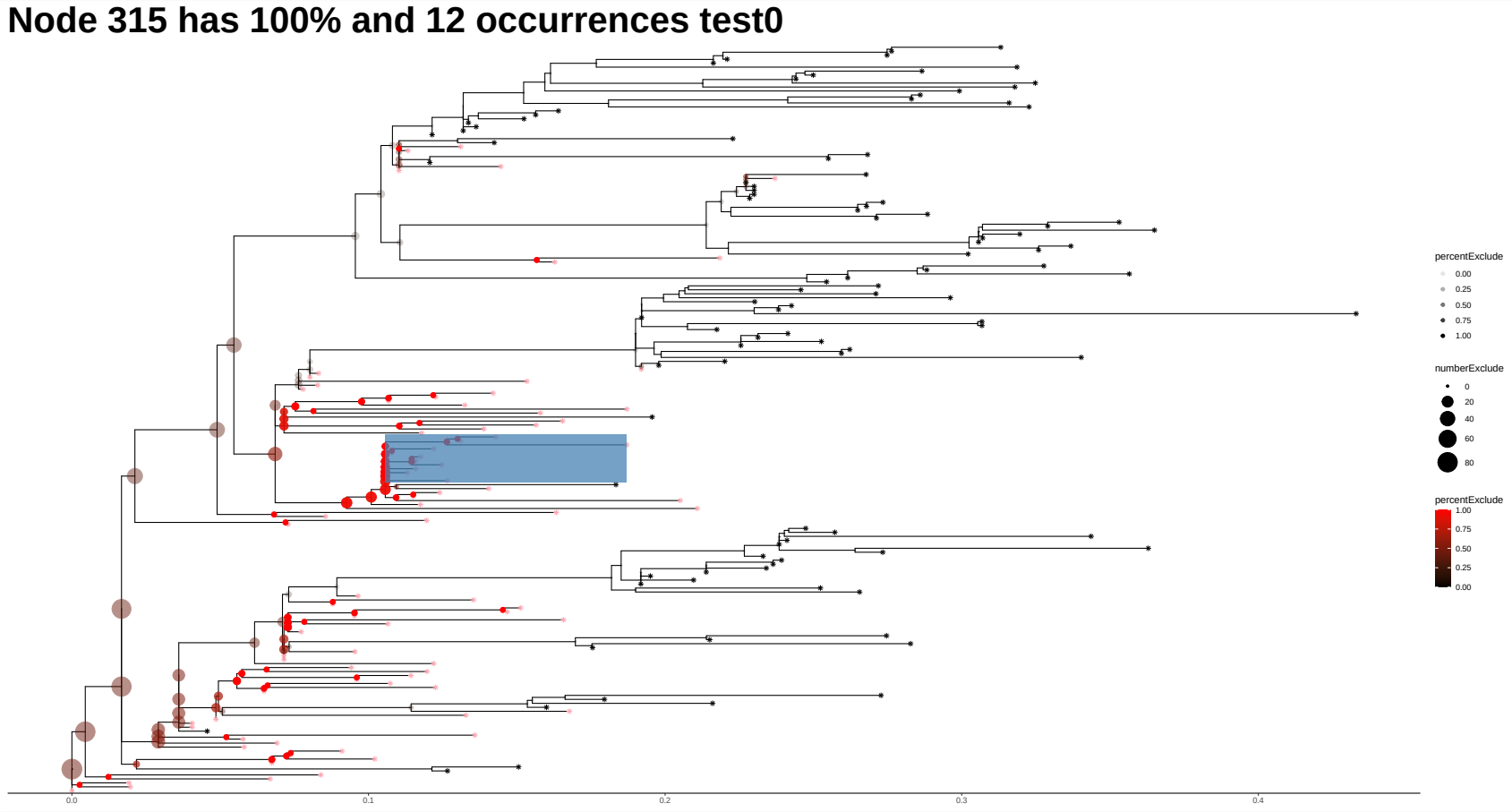
Simulation exported tree annotated for location ’test3’

Tree (ML Estimated)



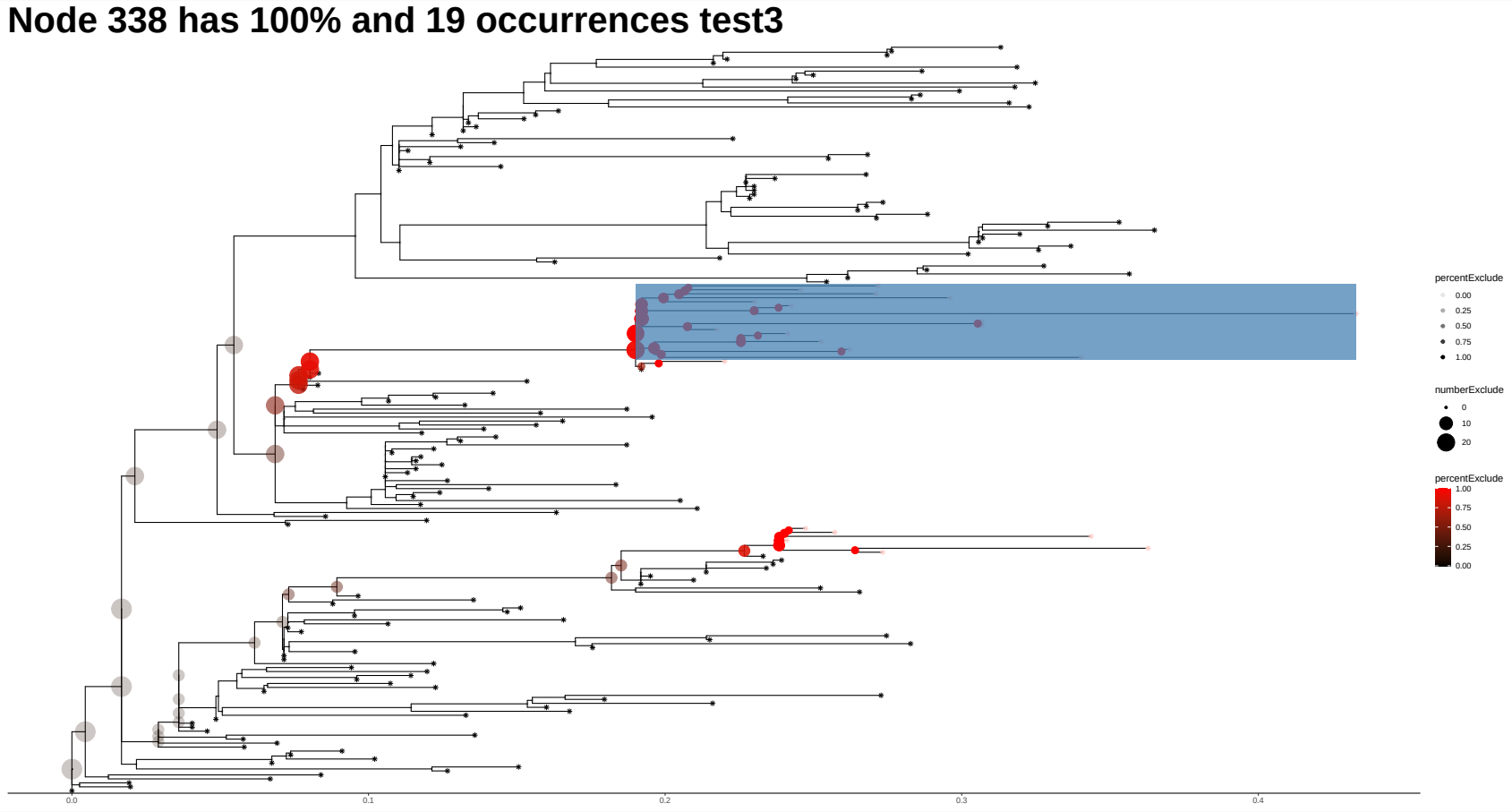
Tree calculated with maximum likelihood annotated for location ’test3’

Tree (ML Estimated + Sampling)



Tree calculated with maximum likelihood annotated for location ’test0’ (0.2 sample)

Tree (ML Estimated + Sampling)



Tree calculated with maximum likelihood annotated for location ’test3’ (0.2 sample)

Identifying Fitness

1. Find viruses that spread most often
2. Fitness $= \sum_{k=0}^{n} \frac{1}{d}$ where n is the total number of children and d is the distance to that child
3. Graph against individual traits
4. Found from phylogeny

	EGPF+M1M1M1M1M1M1M1	EGPF+M1M1M1M2M1	E
EGPF+M1M1M1M1M1M1M1	0	8	
EGPF+M1M1M1M2M1	8	0	
EGPF+M1M1M2M1M1M1M1M1	14	10	
EGPF+ATN4+M1	20	16	
EGPF+1H7G+M1	20	16	
EGPF+SQW1+M1M1	21	17	
EGPF+M1M1M2M1M2M1M1M1M1	15	11	
EGPF+M1M1M2M1M2M1M1M1M2M1M1	18	14	
EGPF+M1M1M2M1M2M2M1M1	15	11	
EGPF+M1M1M2M1M3M1M1	13	9	
EGPF+M1M1M2M2M1M1M1M1	14	10	
EGPF+M1M1M2M3M1	11	7	
EGPF+LBSD+M1	22	18	
EGPF+M1M1M2M4M1M1M2M1M1M1M1	20	16	
EGPF+V1NF+M1	23	19	
EGPF+M1M1M3M1M1M1M1M1M1	14	10	
EGPF+M1M1M3M1M1M1M1M1M2M1M1	16	12	

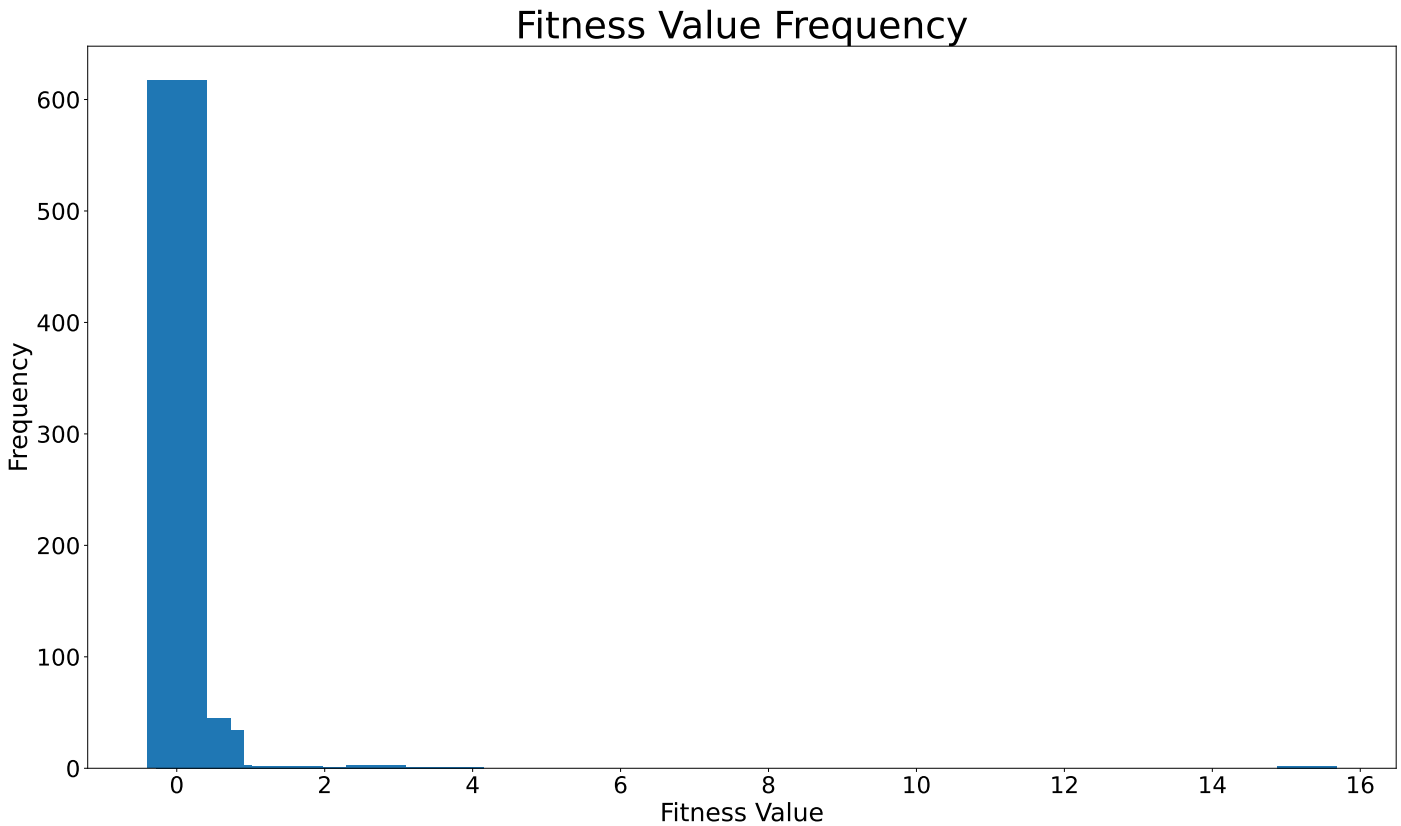
Figure 21: Distances CSV

```python
def compute_fitnesses(tree_path):
    tree = Tree(tree_path, format = 1)

    fitness_matrix = {}
    for seq1 in tree:
        if not seq1.is_leaf():
            continue
        sum = 0.0
        if (seq1.up):
            for seq2 in seq1.up.children:
                if seq1 != seq2:
                    sum += 1/(tree.get_distance(seq1, seq2)+1e-6)
            fitness_matrix[seq1.name] = sum
```

Figure 22: Fitness Calculation

Identifying Fitness



Next Steps for Fitness

1. Find explicit formula for fitness...
2. OR make model to predict fitness

Future Directions

1. Compare sampling effect
2. Compare tree reconstruction techniques (UShER, ML, MP, etc.)
3. Look at pandemic mitigation efforts
4. Observe transmission graph
5. Apply ideas like fitness to real-world data

Figures i


D. C. Adam, P. Wu, J. Y. Wong, E. H. Y. Lau, T. K. Tsang, S. Cauchemez, G. M. Leung, and B. J. Cowling.
Clustering and superspreading potential of sars-cov-2 infections in hong kong.
Nature Medicine, 26(11):1714–1719, Nov. 2020.


S. Mysid.
English: Cartesian grid, Nov. 2006.


Slowkow.
Needleman-wunsch pairwise sequence alignment, Apr. 2014.


V. Terzic.
Deutsch: Wilder baum in der natur., Jan. 2016.