Università degli Studi di Udine
Dipartimento di Scienze Matematiche, Informatiche e Fisiche
Corso di Laurea in Informatica

Bachelor Thesis

CIMICE : markov Chain Inference Method to Identify Cancer Evolution

Candidate
Nicolo' Rossi

Supervisor
Prof. Carla Piazza

Co-Supervisors
Prof. Alberto Policriti
Prof. Bud Mishra

Academic Year 2017-2018

Institute Contacts
Dipartimento di Scienze Matematiche, Informatiche e Fisiche
Università degli Studi di Udine
Via delle Scienze, 206
33100 Udine — Italia
+39 0432 558400
http://www.dimi.uniud.it/

Abstract

Cancer is one of the first causes of death in the world, especially in high-income countries. In fact, the World Health Organization has estimated that more than 60 people over 100000 die of cancer involving the respiratory system, making it the third most lethal disease in developed countries [3]. The strength of cancer is that of not being just a single disease but a class of them, where each tumor in each person and even in each tissue is different and evolves accordingly to rules that are still unclear. To cope with this behavior researchers in medicine and biology are gradually developing methods capable of providing personalized treatments to the patients. The sequencing techniques are currently in fast development and data about cancer is growing faster every day as the technologies are becoming more precise, reliable and cheaper.

Data analysis has acquired a crucial role in this field and one of the most challenging topic in this area is the reconstruction of cancer progression from cancer data. This is not an easy task as genomic data is noisy, incomplete and, in almost every case, acquired at a single time in the tumor existence, forcing the reconstruction of temporal information from static data. After many years from the first seminal work of Desper et al. [4], there is still not an accepted method that is actually used in medicine, but the efforts on the topic are growing and this research field got a name of its own: tumor phylogenetics.

The number of tools available to this end is so high that many reviews are now focusing on understanding and comparing their many complex approaches [1, 9]. Our goal is to prepare another tool for cancer progression inference but focusing on keeping it simple, fast and modern by making very few and reasonable assumptions, managing the analysis even on the largest datasets available, and using single cell sequencing data. Then we will proceed by comparing it with some more complex state-of-art tools, in particular the ones from the TRONCO pipeline (CAPRI, CAPRESE, TRaIT) [2], using both synthetic and real data.

CIMICE is implemented in Java and the sources are located here: `https://github.com/redsnic/tumorEvolutionWithMarkovChains`

Contents

Introduction

The goal of this work is to describe a novel method for cancer evolution inference, CIMICE, and to show how it performs on synthetic and real data.

Tumor phylogenetics is a quite new research field and cancer evolution inference is one of its most appealing topics; it is defined as the reconstruction of the sequence of states that are involved in the progression of a tumor from data. Until last years this process was considered to be linear and pretty regular among cancers of the same kind. It was thought in fact, that there was a precise sequence of genomic alterations that gradually lead the tumor through all of its stages, from the loss of duplication control to the metastatic diffusion trough the organs of the patients. This model became obsolete when it was discovered that cancer is composed of many different subpopulations [9]. Being quite improbable of having multiple sources for tumor initialization, this behavior hinted that cancer was more complex and that its progression may somehow be similar to that of species. This led to the development of tree cancer models, in which multiple evolutionary trajectories are possible. Cancer evolution is still to be fully understood and, due to the extreme complexity of the processes involved in it, there is still not a precise understanding of how to model tumors.

Globally there is a huge interest in tumor progression. Being able to reconstruct and possibly predict cancer evolution would be a great advantage in the preparation of personalized treatments to patients, making it possible to understand how the chosen drugs will behave in relation to the present and future cancer statuses. The challenge in this context is not only the definition of cancer models, but also their reconstruction from available biological data. Most methods use classical phylogenetic approaches often combined with standard learning algorithms that are based on likelihood maximization; valid approaches, but sometimes flawed by not very fitting assumptions or by extremely high theoretical complexities that force the use of heuristic search. [9]

To contribute actively to this field we prepare a novel model thought specifically for single cell sequencing, that approaches this open problem by using Markov Chains. We built CIMICE with simplicity in mind and on few focal points:

- to use a minimal set of assumptions
- to have low complexity reconstruction
- to have well defined limits

- to be a base on which to develop further improvements

This tool can be used in pipelines for data analysis and has a modular architecture itself. The input considered by CIMICE is a mutational matrix, a simple data type that associates genomic alterations to genes [9]. This data must be produced by a pre-processing step on experimental data. It is impossible to include this part directly in the tool because the origins of this kind of information can be very different and is necessary to always adapt the analysis specifically for it. The second step in the analysis is the structure reconstruction, in this process all the possible evolutionary trajectories are extracted and arranged in a directed and acyclic graph (DAG). This kind of representation is more expressive than the standard tree models because it permits the confluence of many different trajectories to the same cancer state, information that should be of great value in an hypothetic medical use. The way in which we prepare this structure is actually a parameter of CIMICE and can be modified for further improvements. In the third step we estimate the probability of each evolutionary trajectory by using a custom heuristic to disentangle the confluences. This phase returns a graph that shows a possible explanation for the input dataset, describing the transition probabilities from a stage to another when the disease acquires alterations. Finally, we use a standard format for output that permits easy data post-processing, like cancer stages annotation or filtering.

We provide a list of synthetic examples of model reconstruction that show how this tools performs when reconstructing simple and complex evolutions. We also prepared some preliminary tests on biological data that confirms that the tool is ready to operate in real data environments and we prove some formal properties of the reconstructed models.

The thesis is divided in four main different chapters. The first one gives a brief description of the background needed to understand the ideas behind CIMICE. The second one is the main core of this work and it describes in detail how our tool was made and why it is a valid option for cancer evolution inference. The third one contains a list of comparisons of CIMICE with the TRONCO pipeline tools in many different conditions. The fourth one shows two examples of the application of our tool to real biological data coming from two really different sources [5, 8]. Finally, in the summary we present some considerations and future directions.

1

Preliminaries

In this chapter our goal is to provide the reader with the minimum necessary information for the comprehension of the research field of this work.

1.1 CANCER AND MUTATIONS

The World Health Organization defines cancer as a class of diseases where a group of abnormal cells overgrow firstly invading the adjoining tissues and then extending to other organs of the body. The reasons why this can happen must be researched in the genetical and epigenetical code of the cancer cells. Biologist divided genes in two categories related to the effect of their disregulation in cancer cells: **proto-oncogenes** and **tumor soppressors**. The former are genes that, once mutated in oncogenes, actively support tumor growth, while the latter are often involved in error correction pathways or in cell cycle regulation and once suppressed their control function is lost. **TP53** is a typical example of a tumor suppressor gene and it is often called the genome keeper for its function of recognizing critical inconsistencies in the genetical code of the cell and organizing a response leading to DNA repair, if possible, to the block of cell cycle or to apoptosis (cell death). In particular the last two actions are quite important because they automatically stop a possible tumor growth, therefore it is not very surprising that this is one of the most mutated genes in human cancer [6]. For the proto-oncogenes an example is **PDGF**, a growth factor related to the formation of blood vessel that can be involved in many different tumors by inducing cell proliferation [7].

At the core of cancer are mutations, permanent modification of genome that are gradually acquired in the cell population. It may be helpful to remember that the DNA is basically a double helix consisting of two anti-parallel complementary strands. These strands are composed by long sequences of four nucleotides A,C,G and T, where A complements T, C complements G and vice versa. To produce a protein two different phases are necessary: transcription and translation. The first stage happens in the nucleus of the cell where DNA is converted to mRNA by proteins called RNA polymerase; the second happens in the cytoplasm and consist in the conversion of the mRNA in proteins. In the coding sections

of the genome, each three nucleotides, also called codons, are associated to the production of a single aminoacid of a protein or are signal for the translation stop. So it is quite clear that changing even a single nucleotide can create the modification of an aminoacid that can lead to misbehaviors of the produced protein. An emblematic case of this is the sickle cell anemia that is caused by a single point mutation in the HBB gene, coding for the β subunit of the hemoglobin.

There are many different kinds of mutation with very different scales which can involve cellular genomes even during tumor development, here are some examples:

- **SNVs**: Single Nucleotide Variants, also called point mutations, are the modification of a single nucleotide in the genome. They are distinguished in two classes: transitions, that replace a purine with another purine (A-G) or a pyrimidine with another pyrimidine (C-T), and transversion, that replaces a purine with a pyrimidine
- **Indels**: Event of insertion or deletion of a variable number of nucleotides at a certain position of the genome. Tandem mutation, like the ones associated to the Huntington's chorea, belong to this class.
- **CNVs**: Copy Number Variations, refers to the creation or the removal of copies of parts of the genome. Having multiple copies of genes is a common regulatory system in plants but it is not very diffused in animals and can lead to disgregulation of protein expression
- **CIN**: Chromosome INstability, increased risk of anomalous division of chromosomes, causing macroscopic variations of the genome
- **Katageis**: accumulation of a burst of SNVs in a restricted chromosomal area
- **Chromothripsis**: semi-random reassembly of a chromosome
- **Chromoplexy**: typical result of a Breakage Fusion Bridge (BFB) event, where two chromosomes link their ends together and this causes their breakage during cellular division

SNVs, Indel and CNVs are quite common in tumors when the error correction mechanism are compromised and their impact is variable, while the other lead to macroscopical changes which are better observed at chromosomal level.

1.2 TUMOR PHYLOGENETICS

Tumor phylogenetics is the research field that aims to reconstruct the sequence of mutations that lead to the formation of a tumor from experimental data. The name phylogenetics refers to the phylogeny of species, an older area of study that had already developed many methods to trace the evolutionary history of organisms.

There is a wide agreement on considering cancer progression evolutionarly when the tumor undergoes a

selective pressure, like that of a treatment, while in other circumstances this evolution is less precisely guided. It is in fact more probable that, without selection, the tumor starts acquiring mutations in a random fashion leading to the formation of many subclones with different genotypes and survival fitness to treatments [9]. From the cancer point of view this is quite an advantage: the survivability of its cells to external attacks is increased, cell specialization is supported, and functionality acquisition speeds is improved.

Knowing in advance what mutations a tumor will acquire would be an huge advantage to prepare better treatments to patients, maximizing success probabilities and minimizing treatment related risks.

1.3 DATA TYPES

The data used by the phylogenetic methods is distinguished in different classes based on how data is collected and on where it is collected from. Currently the majority of data comes from next generation sequencing techniques, like RNAseq and DNAseq but there is still data available based on microarrays or older hybridization techniques like FISH (Fluorescence In Situ Hybridization) that can look up for a single kind of alteration with very high precision.

As seen in figure 1.1, there are mainly three kind of experiments: **cross-sectional** where **patients** are samples, **regional** or **bulk** where **tissues** are samples and **single cell** where, as the name says, the **cells** are samples. The first ones are the most common, for example the majority of the **TCGA** data is of this kind, and they mix together data coming from a cohort of many different patients having the same disease. The second ones were used in the original work of Desper et al. [4], in it only a patient is taken into account and the many different extracted tumoral tissues, as metastasis, are analyzed separately. The last kind of data analyzes the single regions with a better resolution by extracting many samples of single cells from the various tissues, showing the inter-regional diversification of the tumor; data of this type is not very diffused yet but the technique is rapidly spreading and the results look really promising.

Figure 1.3 shows the main experiments for genetical end epigenetical analysis and 1.2 the typical models and algorithms used in the data analysis.

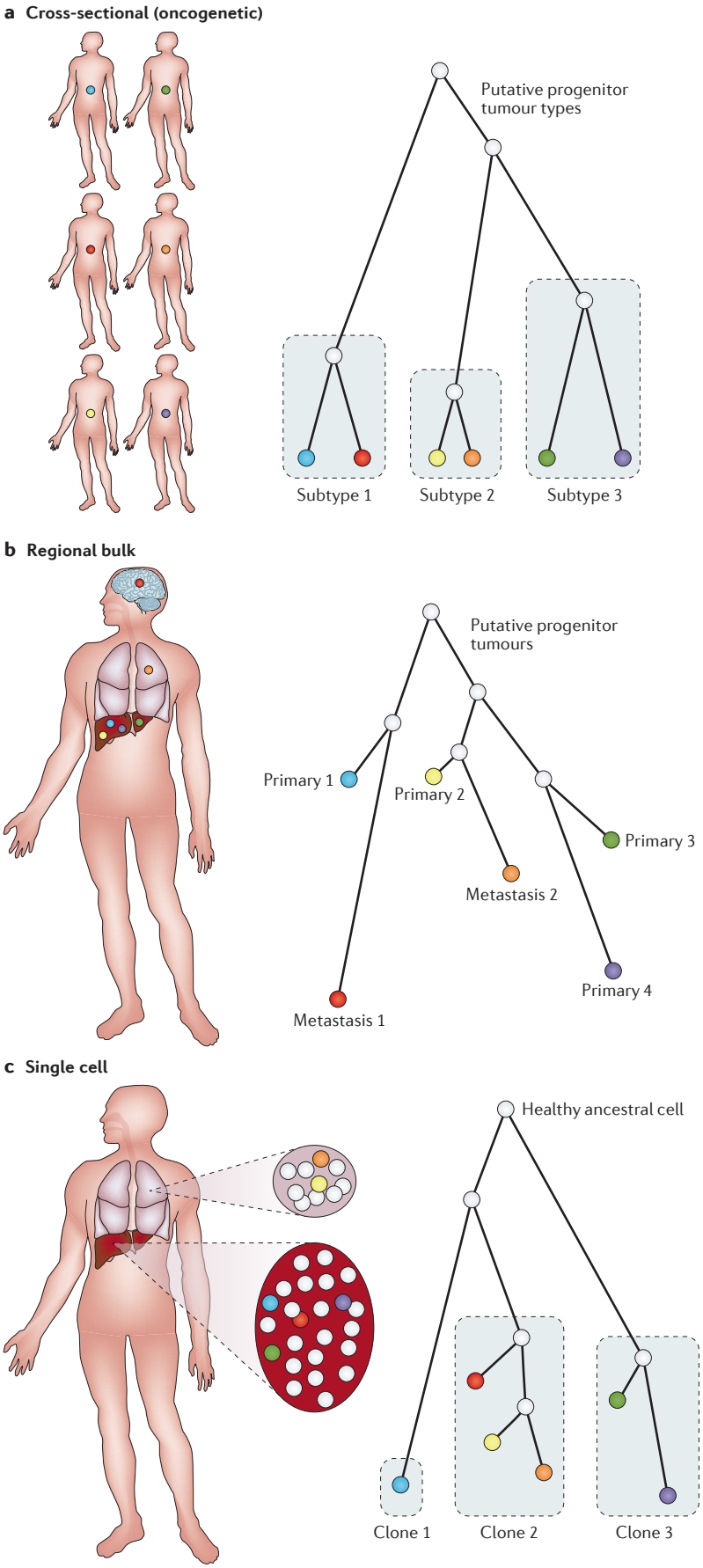

Figure 1.1: Visual representation of the three kind of experiments and a possible phylogenetic tree of the analyzed samples [9] (license number 4425280760601)

1.4 STATE-OF-ART OF CANCER PHYLOGENY TOOLS

In the last years the field of tumor phylogenetics assisted to the fast growth of the number of tools available for data analysis.

Even though none of them is equal, they can be classified in relation to the input, the algorithm and the output used. Input, models and algorithms mainly follows figures 1.3, 1.2 while outputs usually are in the form of clonal or mutational trees or, less frequently, DAGs. Clonal means that each node of the graph is a genotype and each edge displays a genotype change. Mutational means that each vertex is a single altered gene and edges means that the source gene can cause a mutation on the destination one.

CIMICE, the tool we show in this work, uses mutational matrices computed from single cell SNVs/Indels data as input, a custom model based on Markov chains that allows reconstruction from data in polynomial time and weighted clonal DAGs as output.

Model or algorithm name	Description	Refs
Evolutionary models and objective functions		
Maximum parsimony (MP)	Simplest phylogeny model; assumes that mutations are rare and so the tree with the fewest mutations is the most plausible	96
Minimum evolution (ME)	Distance-based analogue of maximum parsimony; assumes that the tree with the least amount of evolution is the most plausible	85
Probabilistic	Broad class of models well suited to complicated evolutionary scenarios, noisy data, and sampling over unknown evolutionary parameters; generally divided into maximum likelihood (ML), used to find one best-fitting tree for the data and model, and Bayesian, used to identify the space of plausible trees and parameters consistent with the model and data	86, 110, 111, 135, 137
Weighted least-squares (WLS)	Distance-based model defining the most plausible tree as that most closely approximating an input set of distances between taxa by a mean-square measure	121
Phylogeny algorithms and algorithmic techniques		
Combinatorial	Broad class of methods frequently used for character-based phylogenies to optimize over a discrete set of possible topologies; generally the most efficient methods, but suitable only for simpler models; examples include B&B, in which one exhaustively searches a space of all possible solutions while avoiding provably unproductive subspaces, and integer linear programming (ILP) or quadratic programming (QP), in which one converts the problem to a special class of mathematical optimization for which efficient solver programs are available	89, 131, 144
Heuristic search	Broad class of algorithms designed to approximately search a space of trees that are based on empirical effectiveness but are not proven to find the best possible trees; also used when solving for phylogeny models for which efficient, exact methods are unknown; a common generic heuristic is a genetic algorithm (for example, REF. 143), in which one generates a pool of possible solutions and 'evolves' them under a model of mutation and mating; many phylogeny-specific heuristics have also been developed (for example, REF. 82)	121, 143, 176
Neighbour joining (NJ)	Fast method for phylogenetics by successively refining subtrees, approximating a minimum-evolution tree while allowing a possibility of temporally impossible scenarios	76
Unweighted pair group with arithmetic mean (UPGMA)	Method for hierarchically constructing a tree by successively joining subtrees, yielding fast tree reconstruction but being dependent on the molecular clock hypothesis (that variation accumulates at equal rates in all tree branches)	21
Markov chain Monte Carlo (MCMC)	Class of algorithms that is suitable to many forms of probabilistic model and allows one to explore parameter ranges and uncertainty in assignments, but is generally too computationally costly to use on trees of more than a small number of nodes	87

Figure 1.2: The most common models and algorithms for tumor phylogenetics [9] (license number 4425280760601)

Technology and data type	Comments	Refs
Pre-NGS technologies		
Large-scale cytogenetic abnormalities	Convenient before sequencing became ubiquitous, but superseded by more comprehensive genomic studies	66
Microsatellite markers	Rapidly evolving, usually neutral markers	78
FISH	Useful for probing small numbers of CNVs in single cells; largely displaced by scSeq, but still important owing to its practicality for much larger numbers of single cells	69
aCGH	Early high-throughput method for bulk CNV profiling; still in use, although being displaced by DNAseq	145
Expression microarrays	Convenient for high-throughput before RNAseq became widely available; not commonly used for phylogenetics, as it provides only a noisy and indirect measure of genetic evolution	123
SNP chips	Designed initially for genotyping and association studies, but also used in many cancer studies to infer copy number profiles along the genome and to infer CNVs	
Bulk sequence technologies		
DNAseq SNVs	Perhaps the most commonly used marker type, it provides whole-exome or whole-genome profiles of evolution by point mutations	117
DNAseq CNVs	CNVs can be inferred by local changes in sequence coverage, instead of using aCGH or SNP arrays	117
RNAseq expression	More precise and accurate replacement for expression microarrays; nonetheless remains a niche technology for phylogeny studies	124
DNA methylation	Measured by bisulfite sequencing, provides unique information on the evolution of the cell state that is not apparent from conventional DNAseq methods; some methylation markers are neutral, others evolve to select for gene expression	79
scSeq technologies		
DNAseq SNVs	Uniquely powerful method for identifying large numbers of phylogenetic markers at the single-cell level; only recently making inroads as the technology has matured and data quality has improved	87,154
DNAseq CNVs	Perhaps the dominant technology for single-cell tumour phylogenetics, offering coarse-grained profiles of evolution by copy number change; robust to data quality issues in emerging scSeq technologies	71
Single-cell microsatellites	Not a widely used technique but one important to early tumour phylogeny studies; offers important advantages in profiling a putatively selectively neutral marker type	76

Figure 1.3: The most common experimental procedures for data acquisition [9] (license number 4425280760601)

1.5 DISCRETE TIME MARKOV CHAINS

A **Discrete Time Markov Chain** (or DTMC) is a Markov process, so a random process with the Markov Property, with finite and discrete states and discrete transitions among them. The **Markov Property** says that the transitions to other states are related only on the current state, so that there is no memory of the previous events that led to that particular state. Typically this is formalized as follows:

$$P(X(t_n) = s_n | X_{n-1}(t_{n-1}) = s_{n-1}, \ldots, X_1(t_1) = s_1) = P(X(t_n) = s_n | X_{n-1}(t_{n-1}) = s_{n-1})$$

Where $s_1, \ldots, s_n$ are states, $X(t_1), \ldots, X(t_n)$ are the states reached after the $t_1, \ldots, t_n$ discrete transitions respectively.

A DTMC can be fully defined by a discrete and finite set of states Q, a transition matrix M with size $|Q| \times |Q|$ where each cell $\langle a, b \rangle$ describes $P(X(t_n) = b | X(t_{n-1} = a))$ and q_I a vector of $|Q|$ elements indicating for each state its probability of being the initial state. Markov Chains are also the simplest Markov models and they are used with the assumption of the system being fully observable and autonomous.

2

Cancer models and tumor evolution reconstruction

In this chapter we will formally describe our method and its properties, giving also some examples on toy models.

2.1 BASE DEFINITIONS

2.1.1 Genotype

In this context a genotype is considered as a set of genes. Let *Genes* be the set of all genes, a genotype (gnt) is a set:

$$gnt = \{g_1, g_2, \ldots, g_n\} \text{ with } g_i \in Genes, i \in \{1 \ldots n\}$$

There can be at most $2^{|Genes|}$ different genotypes. Intuitively, genotypes identify the mutated genes in a sample.

When there are multiple types of mutations on a single gene of interest they are often not distinguished. This tool also supports a better resolution data, where each kind of mutation is considered as if it was a separate gene.

2.1.2 Dataset

A dataset D is a collection, so a multiset, of many genotypes extracted from different samples using biological procedures.

$$D = \{gnt_s | s \in Samples\}$$

where *Samples* is the set of the analyzed samples. This is also called **mutational matrix**.

2.1.3 Reduced genotype

In cancer data analysis it is often required to focus on a subset of all the possible alterations. When we consider a subset of genes (or alterations) we use the term *reduced genotype* to refer to the intersection of a genotype with the subset of the genes of interest. We use the term *reduced dataset* to refer to the dataset obtained by replacing each genotype it contains with its reduction. Formally, given a dataset D and a subset of genes p, the reduced dataset D_p is a multiset of genotypes:

$$D_p = \{x \cap p | x \in D\}$$

Usually, the most common option is to reduce the dataset D to D_{Genes_n}, where $Genes_n$ is the set of the n most mutated genes in D. In case of alterations occurring the same number of times in the dataset, the set $Genes_n$ is extended to contain more than n elements to avoid the forceful exclusion of some genes.

2.1.4 Observed probability of a genotype

It is possible to define an observed probability function for the genotypes of a dataset D:

$$P_D(gnt) = \frac{\#\text{count } gnt \text{ in } D}{|D|}$$

Where $gnt \in D$. So the observed probability of a genotype is the number of times it appears in its dataset divided by the total number of samples.

2.1.5 Splitting function

A **splitting function** h is a function that, for each node g of a DAG G associated with a probability $P(g)$, returns a list of $|indegree[g]|$ elements with sum equal to 1 indicating for each entering edge the percentage of $P(g)$ associated by h to it.

2.2 THE CANCER MODEL

We consider cancer as an evolutionary process that involves multiple entities, the cells. In our model each cell has its own evolution that is represented by the temporal sequence of all the different genotypes it had during its entire existence, starting from the clonal, or healthy, genotype. We consider the evolutionary process continuous and irreversible for each cell and cancer is seen as the collection of all the evolutionary trajectories of the cells it is made of. Considering the evolution of a single cell, according to our model, if a cell has a certain genotype at a certain moment in its life, after a certain time interval the cell can be either in the same stage as before or in a further one, if it evolved to another genotype acquiring new mutations. If we consider a time interval (Δt) small enough to make negligible the probability of having more than one mutational event occurring in it, the number of possible stages after that time is reduced to two: the current state and the next stage of this cell evolution. These stages are mutually exclusive and have a specific probability of occurrence.

A representation of cell evolution within this model can be made with a very simple graph where all the genotypes involved are nodes, each one having self loops labeled with the probability p of remaining in that same stage after Δt time and one other edge to the next stage, labeled with probability $1 - p$.

We can now create a more generic graph showing the probabilities for a cell of a specific tumor to evolve from a certain stage to another after Δt time. That is achieved by merging the representations we made for each cell in a single graph. In particular, if we call $A_i = (V_i, E_i, W_i)$ the graph corresponding to the i-th cell of the tumor and $A = (V, E, W)$ the graph we are willing to create, representing a tumor containing n cells in its final stage:

$$V = \bigcup_{i \in \{1..n\}} V_i$$

$$E = \bigcup_{i \in \{1..n\}} E_i$$

$$W(\langle a, b \rangle) = \frac{\sum_{i \in \{1..n\}} W_i(\langle a, b \rangle)}{\#\text{count}\{A_i | i \in \{1, \ldots, n\} \wedge a \in V_i\}}$$

In practice, W is obtained by mediating all the weights of all the edges of every graph representing a cell. Please take a look at the example in the next subsection.

2.2.1 Model example

Here we consider a very simplified example of a tumor involving very few cells:

- Figure 2.1 represents the complete structure of the tumor, the leaves at level 3 (1 to 6 from left to right) are the sampled cells.

- Figures 2.2 to 2.7 show the sequence of genotypes that lead to each sampled cell.
- Figures 2.8 to 2.13 show some examples of possible Markov Chains underlying the samples evolution.
- Finally, figure 2.14 presents the final Markov Chain representing the evolutionary probability of each sampled tumor cell.

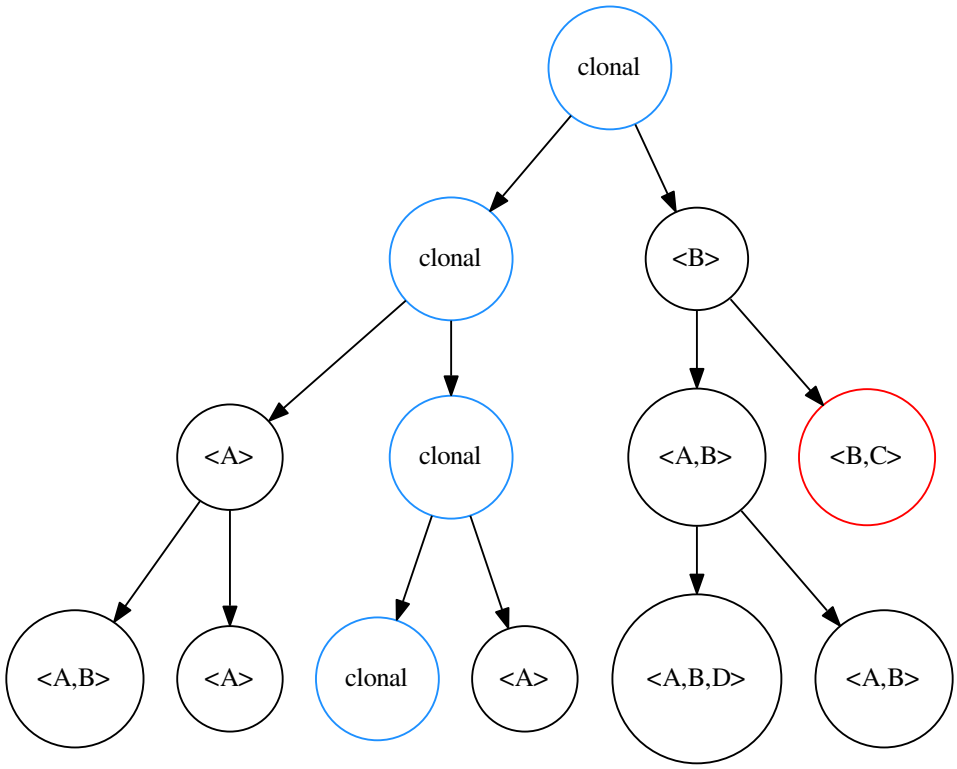

Figure 2.1: This model shows a cell duplicating and gaining mutations in the process. Clonal indicates the healthy genotype, every other node is labeled with the mutated genes in that cell. The genotype $\langle B, C \rangle$ is considered lethal

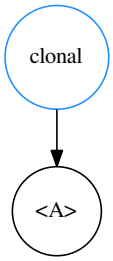

Figure 2.2: history of cell 2

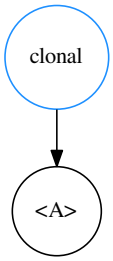

Figure 2.3: history of cell 4

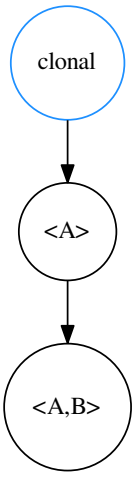

Figure 2.4: history of cell 1

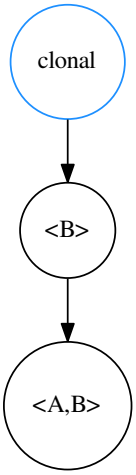

Figure 2.5: history of cell 6

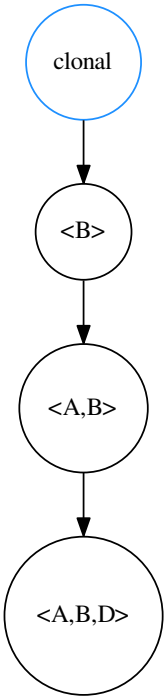

Figure 2.6: history of cell 5

Figure 2.7: history of cell 3

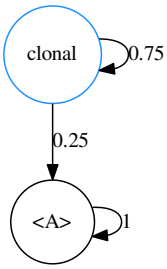

Figure 2.8: DTMC of sample 2

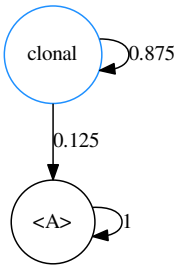

Figure 2.9: DTMC of sample 4

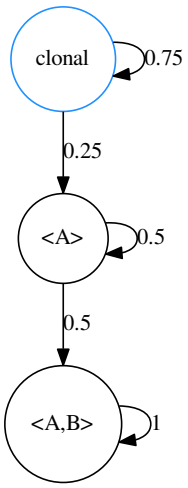

Figure 2.10: DTMC of sample 1

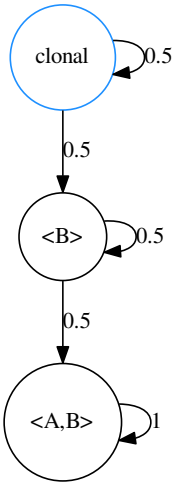

Figure 2.11: DTMC of sample 6

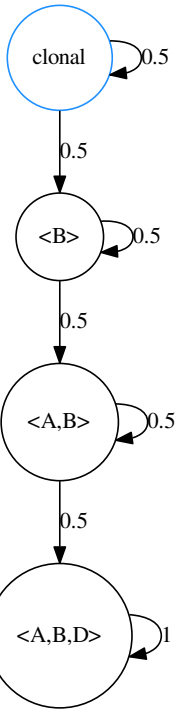

Figure 2.12: DTMC of sample 5

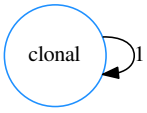

Figure 2.13: DTMC of sample 3

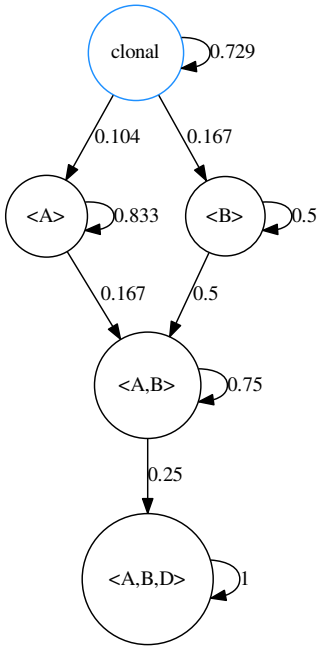

Figure 2.14: Merge of the transition graphs of each cell

Note that the sum of the weights of each exiting edge in figure 2.14 is 1. This is a Discrete Time Markov Chain and it can be used to generate datasets, as described in subsection "Artificial dataset generation".

2.2.2 Assumptions

To use this model with the cancer evolution reconstruction method, that will be later described, some assumptions are necessary:

1. Mutations can be acquired one at a time or in groups but the evolutionary history is always the one involving less mutations
2. When a cell gets a mutation it can not be removed
3. The most probable history of a genotype is the one with most observation
4. The dataset has enough information to be a statistically significant representation of the analyzed cancer at the time of sampling
5. A mutational event in a cell is dependent only by its current genotype
6. A cell has a probability to undergo a mutational event that is independent from time

Note that because of point one this method is fit for SNV data, insertions and deletions (even involving multiple genes). Point two might seem similar to the Infinite Set Assumption (ISA) typically made in

this context, but this version is actually weaker because we are not assuming that the tumor can only have a single way to evolve, so there are multiple possible trajectories to a given genotype.

From these assumptions $A = (V, E, W)$ can not have cycles (excluding self loops) and each node has the sum of the weights of all exiting edges equal to one. This last property, combined with assumptions 5 and 6, makes our model a Discrete Time Markov Chain. From now on we will call A the **generator graph**.

2.2.3 Artificial dataset generation

Our goal is to reconstruct a cancer model from real data, the datasets we use for this analysis are collections of genotypes extracted from cancer cells at a given time, a snapshot of the tumor. We expect to find cells that are in almost every stage of their tumoral evolution, in different quantities and following different evolutionary trajectories. Accordingly to these expectations, it is possible to use our model to generate datasets respecting all our assumptions for validation purposes. To generate an entry of the dataset D_A it is sufficient to set a time interval $t = k\Delta t$ with $k \in \mathbb{N}^+$ and to perform a random walk of length k on the generator graph A adding the genotype of the last node reached to D_A. Repeating this procedure $n = |D_A|$ times we get a complete dataset. When $n \to \infty$ and $k >= \max\_path(A)$ each possible genotype will be represented in the graph as we would expect from our model. For example, by using the generator graph (like the one in figure 2.14) and simulating a random walk of length five ten times we obtain this dataset:

```
s\g      A B C D
Sample_0 0 1 0 0
Sample_1 1 0 0 0
Sample_2 1 0 0 0
Sample_3 1 0 0 0
Sample_4 1 1 0 0
Sample_5 1 1 0 0
Sample_6 1 1 0 1
Sample_7 1 1 0 1
```

note that the two cases in which there were no mutations were removed.

2.3 TUMOR PROGRESSION RECONSTRUCTION

With these notions it is now possible to define, given a dataset D, a graph $A_D(V, E_1)$ where

$$V = \{\langle gnt, P_D(gnt)\rangle | gnt \in D \wedge P_D(gnt) > 0\} \cup \{\langle \emptyset, P_D(\emptyset)\rangle\}$$

$$E_1 = \{\langle v, v' \rangle | v, v' \in V \wedge v[0] \subset v'[0] \wedge |v[0]| + 1 = |v'[0]|\}$$

In other words, a node $v \in V$ represents a unique genotype, $v[0]$ is the set of mutated genes, $|v[0]|$ is the total number of alterations in that genotype and $v[1]$ is the probability to find that genotype in the dataset D. Each genotype is connected to all the other genotypes that have a single additional mutation. $\emptyset$ is the clonal (normal, without any mutation on the considered genes) genotype and it is always present.

Theorem 1 (**The observed genotypes graph is a DAG**).

Given a graph $G = (V, E)$ defined as before, it has no cycles

Proof. By the definition of E_1: $\langle a, b \rangle \in E$ $a, b \in V$ if $|a[0]| + 1 = |b[0]|$, thus from a node it is only possible to reach other nodes with an higher number of mutations. If there is a cycle and $v \in V$ belongs to the cycle, v can be reached by v but this is impossible because v has exactly the same number of mutations of v.

□

We now define some simple notations to make the subsequent considerations more understandable and concise:

$$P(v) = v[1] \text{ where } v \in V$$

$$\Pi_b = \{a | \langle a, b \rangle \in E_1\}$$

$$\Lambda_a = \{b | \langle a, b \rangle \in E_1\}$$

so Π_b is the set of the parents of b and Λ_a is the set of the children of a.

2.3.1 Adding weights to the edges

In this subsection our goal is to define an easy to compute empirical formula that will set transition probabilities on all the edges of the graph. In its creation we considered many different problems related to the input data that will be later discussed. The computation of this formula is divided in four different steps:

1. *UpWeights* extraction: defining the probabilities of a genotype to have a certain history (Figure 2.15).

2. *UpWeights* normalization: normalizing the *UpWeights* of the incoming edges for each node (Figure

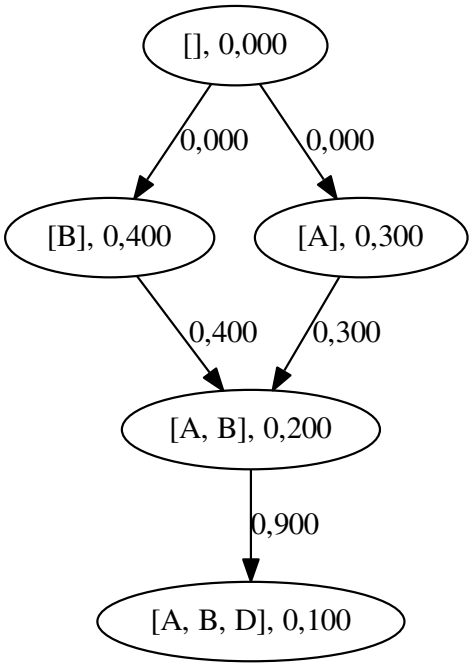

Figure 2.15: First step: upWeights computation

2.16).

3. *DownWeights* extraction: defining the probabilities of a genotype to evolve following a certain path by using *UpWeights* information (Figure 2.17).

4. *DownWeights* normalization: normalizing the *DownWeights* of the outcoming edges for each node. These are the final weights (Figure 2.18).

For the first step we use this formula:

$$W_{up}(\langle a, b\rangle \in E) = \frac{1}{|\Lambda_a|}(P(a) + \sum_{x \in \Pi_a} W_{up}(\langle x, a\rangle))$$

its goal is to estimate how many of the observations of a genotype are part of trajectories traversing a given edge. The empirical assumption here is that the most probable origin of a genotype is the most observed one, this is why $P(a)$, the observed probability of the source of the edge, has a role here. The recursion is executed to also take into account the observations of all the ancestors of a. The division by $|\Lambda_a|$ is needed to avoid over-considering a node with many children and has the property to give the same *UpWeight* to histories with equal observed probability.

The second step is just a normalization to set the sum of the incoming edges of each node to 1:

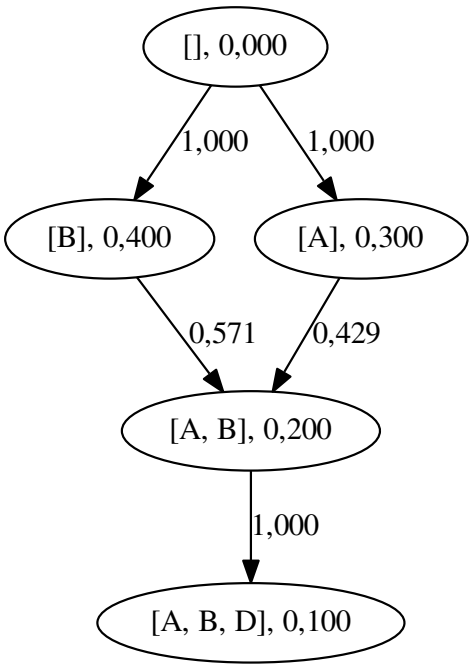

Figure 2.16: Second step: upWeights normalization

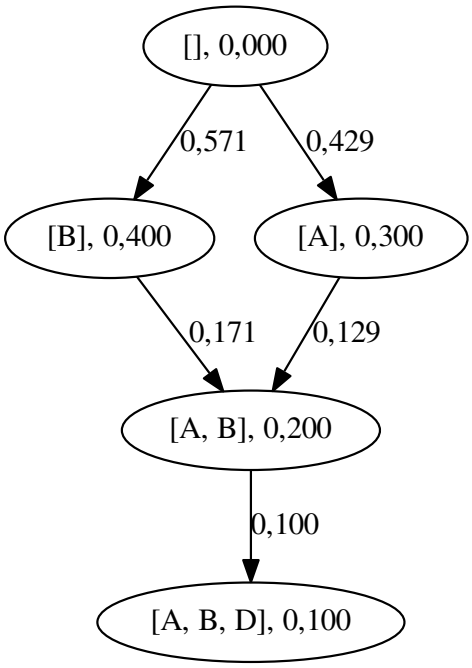

Figure 2.17: Third step: downWeights computation

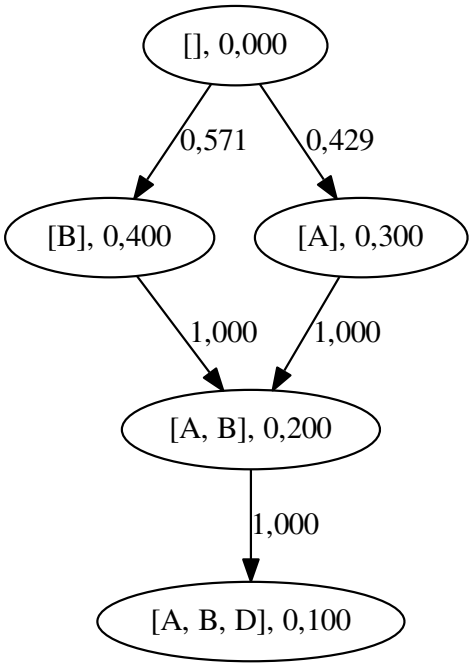

Figure 2.18: Fourth step: downWeights normalization

$$\overline{W}_{up}(\langle a,b\rangle \in E_1) = \begin{cases} 1 & \text{if } a[0] = \emptyset \\ \frac{W_{up}(\langle a,b\rangle \in E)}{\sum_{x \in \Pi_b} W_{up}(\langle x,b\rangle)} & \text{else} \end{cases}$$

so now $\overline{W}_{up}(\langle a,b\rangle \in E_1)$ is related to the probability of b to have originated from a.

The third phase is similar to the first, but now we consider the observations of the successors of a node reachable through an given edge:

$$W_{down}(\langle a,b\rangle) = \overline{W}_{up}(\langle a,b\rangle)(P(b) + \sum_{x \in \Lambda_b} W_{down}(\langle b,x\rangle))$$

that is then normalized to get the final weights:

$$P(\langle a,b\rangle) = \overline{W}_{down}(\langle a,b\rangle) = \frac{W_{down}(\langle a,b\rangle)}{\sum_{x \in \Lambda_a} W_{down}(\langle a,x\rangle)}$$

With this last formula we estimate how much it is probable to evolve from genotype a to b, assuming that an evolution towards groups of genotypes is proportional to their observed probabilities. The role of the normalization steps is to transform absolute probabilities referring to the whole graph to relative ones, as seen from a specific node.

2.3.2 The genotype graph is a discrete time a Markov chain

Because of the normalization step described before, the sum of the weights of all the edges exiting from a node is 1, with the only exception of nodes without any out-coming edge for which a self loop probability of 1 is considered implicit. The time in this Markov Chain is discrete and defined by the acquisition of a mutation. The self loops with probability 1 are to be considered more as a lack of further information on late mutations than as a fixed state of the tumor, which however could also be the case when considering small subsets of mutations.

2.3.3 Adding a simple support for multiple mutations at a time

From the previous definition of edges it follows that, if in the dataset there is a genotype a with n mutations and there is not any other genotype b having a subset of the alterations seen in a with cardinality $n-1$, the node relative to a will have no incoming edges. In the hypothesis that any tumor cell evolves from the clonal genotype, this situation would not be acceptable because a path from the clonal genotype to any other node is needed. The first option we adopted was to directly connect orphan genotypes to the clonal one, meaning that we were not able to reconstruct their history even though they were present in the dataset. Another idea is to avoid the formation of these orphan nodes by extending the rules for the definition of the edges. To do so the definition of E was changed from:

$$E_1 = \{\langle v, v' \rangle | v, v' \in V \wedge v[0] \subset v'[0] \wedge |v[0]| + 1 = |v'[0]|\}$$

to

$$E_\infty = \{\langle v, v' \rangle | v[0] \subset v'[0] \wedge (\nexists v'' | v''[0] \subset v'[0] \wedge dist(v''[0], v'[0]) < dist(v[0], v'[0]))\}$$

where

$$v, v' \in V \wedge v'' \in V / \{v'\}$$

$$dist(A, B) = ||A| - |B||$$

Theorem 2 (E_∞ **is an extension of** E_1)**.**

$E_1 \subseteq E_\infty$

Proof. by the definition of E_1 if $\langle a, b \rangle \in E_1$ then $|a[0]| + 1 = |b[0]|$ and $a[0] \subset b[0]$ so $dist(a[0], b[0]) =$

1. If $\langle a, b\rangle \notin E_\infty$, then, by the definition of E_∞, there must exists c so that $\langle a, c\rangle \in E_\infty$ and $dist(a[0], c[0]) < 1$. Being $\forall x, y(dist(x, y) \geq 0)$, then $dist(a[0], c[0]) = 0$, hence $a[0] = c[0]$, concluding that, again for the definition of E_∞, $\langle a, c\rangle \notin E_\infty$ and this leads to an absurd.

□

The consideration about the absence of loops still holds with E_∞.

Theorem 3 (**Each node can be reached from the clonal genotype node**)**.**

---

if $a \in V$, then there is a path to a from the node relative to the clonal genotype, r

Proof. this proof is done by induction on the length of the path from r to a.

If $(\nexists c \in V \setminus \{\mathrm{r}\} | c[0] \subset a[0])$, then $r[0] \subset a[0]$ and, by the definition of E_∞, $\langle r, a\rangle \in E_\infty$ or $a = r$ so the hypothesis holds.
If $(\exists c \in V \setminus \{\mathrm{r}\} | c[0] \subset a[0])$, then there exist at least a node $d \in V \setminus \{\mathrm{r}\}$ so that $\langle d, a\rangle \in E_\infty$. If there is a path of length l from a node n to d and $\langle d, a\rangle \in E_\infty$, then there also is a path of length $l + 1$ from n to a. For the inductive hypothesis there is a path from r to d and also from r to a.

□

From Theorem 3, each genotype has at least one evolutionary trajectory coming from the clonal genotype. Note that the computation of weights remains the same even with these changes.

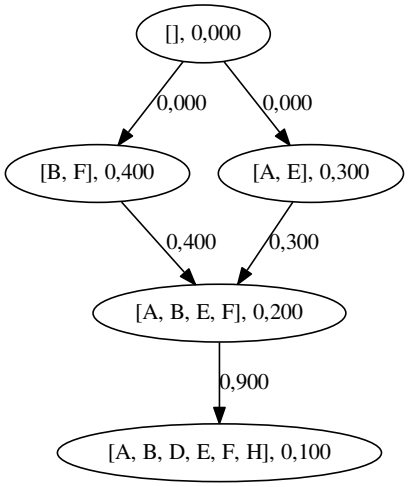

Figure 2.19: It is possible to acquire multiple mutations in one step

2.3.4 Limits of this method

Our reconstruction method extracts transition probabilities using an empirical approach based on genotype observations. This method can be biased when assumption 3 does not hold. This can lead to a miscalculation of edges weights when a genotype is reconstructed as preceded by more than one genotype. For example if we have some genotypes with only A gene mutated, others with only B mutated and some with A and B mutated, our method will always consider possible both the evolutionary trajectories passing by A and B respectively which then reach the AB genotype. We cannot understand from data if, for example, genotype A cannot lead to AB in these circumstances.

2.4 A SIMPLE EXAMPLE

To show how the reconstruction works we give here a simple example using a toy dataset:

```
s\g      A B C D
Sample1  1 0 0 0
Sample2  1 0 0 0
Sample3  0 1 0 0
Sample4  0 1 0 0
Sample5  0 1 0 0
Sample6  1 1 0 0
Sample7  1 0 1 0
Sample8  1 0 0 1
Sample9  1 1 0 1
Sample10 0 1 0 1
```

There are ten samples, the considered genes are four, A, B, C and D, and each number 0 or 1 is associated with the absence or the presence respectively of a mutation in a given sample on a given gene. To prepare the structure, firstly it is extracted the observed probability of each genotype, in this case $\langle A\rangle(0.2)$, $\langle B\rangle(0.3)$, $\langle A, B\rangle(0.1)$, $\langle A, C\rangle(0.1)$, $\langle A, D\rangle(0.1)$, $\langle A, B, D\rangle(0.1)$, $\langle B, D$ and $\rangle(0.1)$, and then the nodes are created. Each node is subsequently connected with an incoming edge to all the other nodes labeled with genotypes that are subset of the one of the current node and having maximum cardinality. Finally the weights are computed in four steps as seen in the subsequent figures (from 2.20 to 2.23).

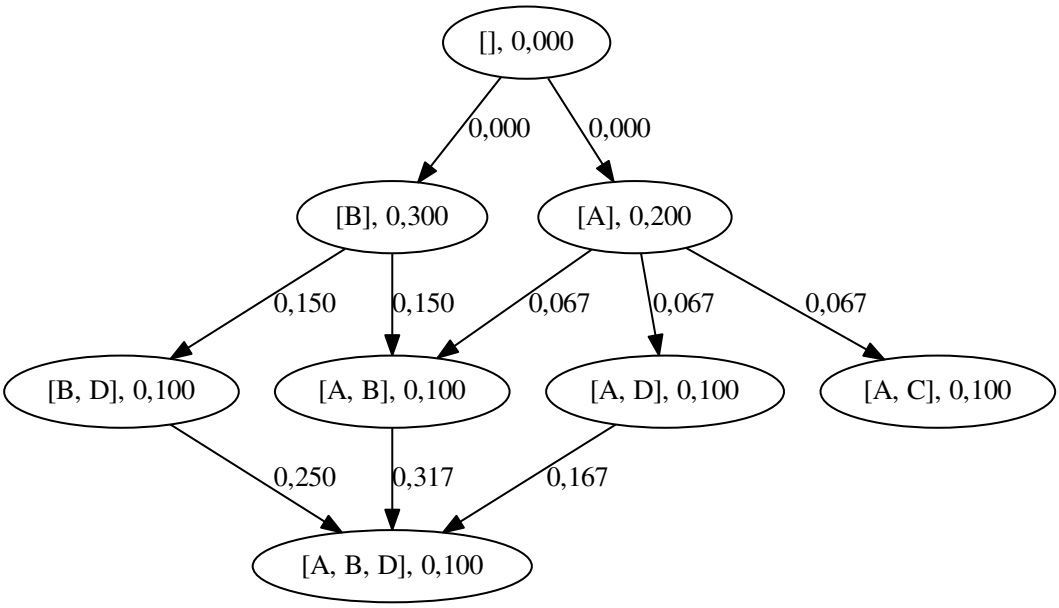

Figure 2.20: First step: upWeights computation

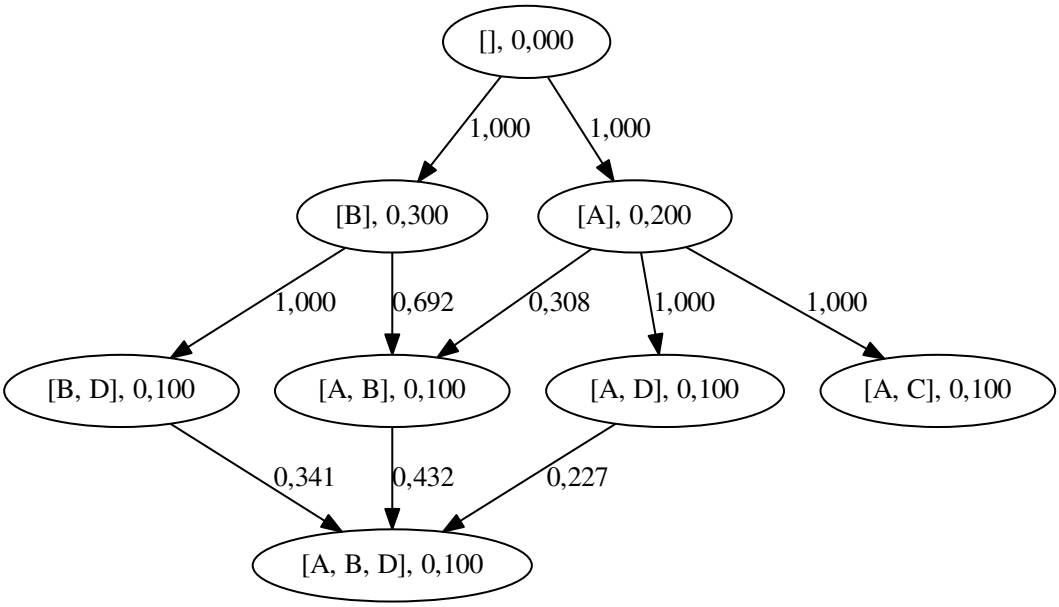

Figure 2.21: Second step: upWeights normalization

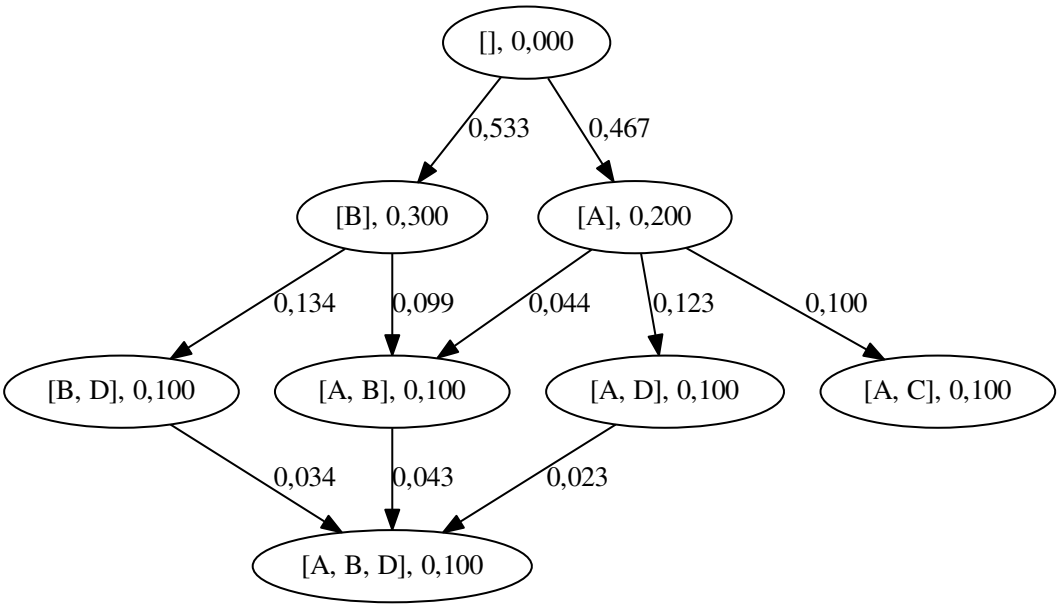

Figure 2.22: Third step: downWeights extraction

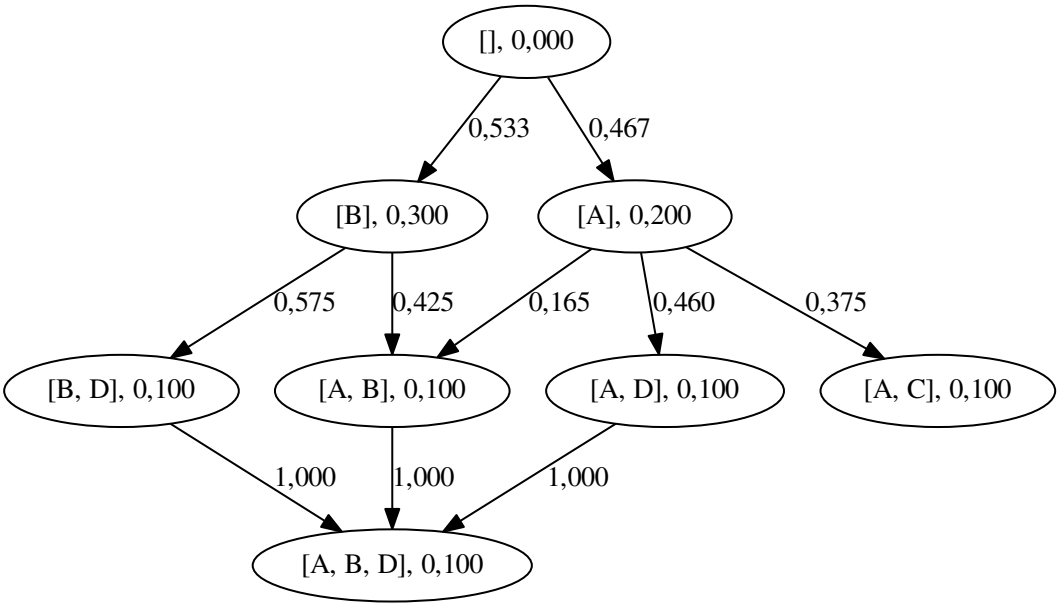

Figure 2.23: Fourth and final step: downWeights normalization

2.5 CONSIDERATIONS ON METHOD ACCURACY

In this section we show how our method can infer a model that is representative of a generator that could have produced the observed dataset both in simplified and general cases.

2.5.1 Additional definitions

We start with some useful definitions that will formalize the elements we will consider during our proofs.

Definition 1 (**Generator**).

We define G as a generator if G is a Markov Chain, has a DAG topology with self loops, a single genotype associated to each of its nodes and respects the assumptions of our method.

It is important that the generator respects our assumptions because it means that our tool is capable of reconstructing the topology of the generator correctly, for example a generator that can produce genotype $\langle A \rangle$ after $\langle A, B \rangle$ would be against assumption 2 and, by the rule of E_∞, CIMICE would never put an edge from $\langle A, B \rangle$ to $\langle A \rangle$.

Definition 2 (**k-admissible generator**).

We define G as a k-admissible generator for a dataset D if for each genotype $g \in D$ it holds that $P_D(g) = P_G^k(g)$, where $P_G^k(g)$ is the probability of generating g in a simulation of G with paths of length k.

Intuitively, it is a generator that, when simulated infinity times with paths of length grater than its height, would produce a certain genotype with exactly the probability of extracting it choosing randomly an entry from the dataset.

Definition 3 (**Structure**).

A structure for a dataset D is a connected DAG $S = (V, E)$ with a single root where each vertex v is associated with an element $label[v] = g \in D$ and with a probability $P(v)$ and for all $g \in D$

$$P_D(g) = \sum_{x \in \{v \in V | label[v] = g\}} P(x)$$

So, a structure is basically determining the temporal order of the genotypes in all the possible evolutionary trajectories.

The two following procedures decide how a DAG with nodes labeled with probabilities can be transformed in a tree with nodes labeled with probabilities determined by a given splitting function.

Definition 4 (**Splitting function application**).

Given a structure S and a splitting function f we define $f(S)$ as the result of this procedure:

For all non-root node v in topological order do:

- *apply f over the current node (v)*
- *split v creating $|indegree[v]|$ different nodes $v_1 \dots v_{|indegree[v]|}$ with $P(v_i) = P(v)f(i)$*
- *link each v_i with a single parent that is $(parents[v])[i]$ and with the same children of v*
- *remove v from T*

This second case is more complex due to the self loops management.

Definition 5 (**Splitting function application on a generator**).

Given a generator G and a splitting function f we define $f(G)$ as the result of this procedure:

For each non-root node v in topological order do:

- *apply f on the current node (v)*
- *split v creating $|indegree[v] - 1|$ different nodes $v_1 \dots v_{|indegree[v]|-1}$ with $P(v_i) = P(v)f(i)$ where $P(v)$ is the probability of reaching the node v afrer k transitions.*
- *link each node v_i with a single parent that is $(parents[v])[i]$ and is not v, and with the same children of v*
- *weight each edge $e \in f(G)$ to respect $P(v_i)$ (there can be at most a single way to assign the weights, see Theorem 4)*
- *v is removed from T*

Definition 6 (**Non redundant k-admissible generator**).

A k-admissible generator G is said to be non redundant if:

$$(\forall g \in G) \nexists g' \in G \setminus \{g\} | g = g'$$

Definition 7 (**Non redundant structure**).

A structure S is said to be non redundant if:

$$(\forall g \in S) \nexists g' \in S \setminus \{g\} | g = g'$$

Intuitively, we use the term *non reduntant* for either a generator or a structure to identify the ones that associate a different genotype to each node.

Definition 8 (**Indistinguishable generators**).

G and G' are said to be indistinguishable if:

$$paths[G] = paths[G']$$

We mean that, considering nodes labeled with the same genotype equal, two generator are *indistinguishable* when their sets of possible paths are equal.

Definition 9 (**Untimed generator**).

An untimed generator untimed(G) of a generator G is a graph so that:

- *untimed(G) has the same nodes of G*
- *untimed(G) has no self loops*
- *the weight function is changed to:*

$$W_G(\langle a, b \rangle) = \frac{W_{untimed(G)}(\langle a, b \rangle)}{\sum_{x \in adj[a] \setminus a} W_{untimed(G)}(\langle x, b \rangle)}$$

So, an *untimed generator* for a graph G, is a DAG where each edge is labeled with the probability of transitioning from a node to another when a mutation is acquired in the original generator G.

Definition 10 (Unravelling tree for a DAG).

An unravelling tree T for a DAG A from a node $a \in A$ is a tree so that:

- $a \in T$
- *for each node $v \in subDAG(a)$, $indegree[v]$ nodes $w_1, \ldots, w_{indegree[v]}$ copies of v are in T*
- $w_1, \ldots, w_{indegree[v]}$ *have a single parent*
- *the parent $w_1, \ldots, w_{indegree[v]}$ is one of the parents of v*
- *none of $w_1, \ldots, w_{indegree[v]}$ has the same parent*

We will show later that the application of the splitting function is actually a particular case of DAG unravelling from the root.

2.5.2 Method correctness

In this subsection we aim to prove that our method is capable of extracting an untimed version of a *k-admissible* generator for the analyzed dataset that is compatible with a chosen splitting function.

We start by proving that, in the simplified case of trees, there can be at most one possible solution to our problem.

Theorem 4.

Given a dataset D, a structure T for D with tree topology and $k \geq height[T]$, there exists at most one G k-admissible generator for D with the same topology of T.

Proof. Since G has to have the same topology (self loop excluded) of T, we use the notation v' to denote the node of G corresponding to the node v of T.

Let l be a list of real numbers. Let $C'(l, n)$ be the set of combinations with repetitions of n elements from the list l. For each combination $x = (e_1, e_2, \ldots, e_n) \in C'(l, n)$ we can consider the product $e_1 * e_2 * \cdots * e_n$ of all the numbers occurring in x. We consider the sum $\tilde{C}'(l, n)$ of all such products. Formally:

$$\tilde{C}'(l, n) = \sum_{x \in C'(l,n)} \left(\prod_{e \in x} e \right)$$

and let $P(v \in T)$ be the probability function associated with T. For G to produce observation accordingly to $P(v)$ by using paths of k length it must be that:

$$P(v') = \sum_{p \in Paths_k(r,v')} \left[\tilde{C}'(p.self, k - |p.edges|) \prod_{e \in p.edges} W(e) \right]$$

where $Paths_k(r, v')$ are the elementary paths (paths not passing more than one time by each node) from the root r to the node v. Each path is seen as a pair $\langle edges, self \rangle$ where *edges* is the list of edges that are part of the considered path and *self* is the list of the weights of the self loops of the traversed nodes. Note that by saying "*G produces observations accordingly to* $P(v)$" we mean that the sum of the probabilities of each k length path of G that ends on a node v' is exactly $P(v)$. By the properties of trees, we get that $|Paths_k(r, v)| = 1$.

We will now demonstrate the unique existence of a generator G with the same topology of T by induction on the depth at which all the parameters of the generator, so the weights of self loops s_i and of other edges a_i, are known:

Base Case: depth is 0 (Figure 2.24)

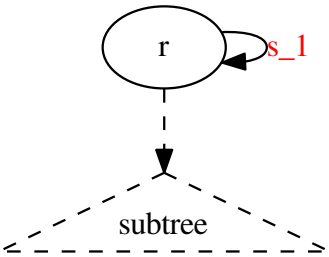

Figure 2.24: Representation on the situation at the base case, s_1 is the unknown

if depth is zero, then the only parameter we want to know is the self loop on the root s_1. We can extract it by using the formula of $P(v)$:

$$P(r) = \tilde{C}'(s_1, k - 1) = s_1^k$$

the only path of length k that can reach r is that one passing by the self loop of r k times. Thus all the parameters of depth 0 are known.

Inductive Step: depth is n (Figure 2.25)

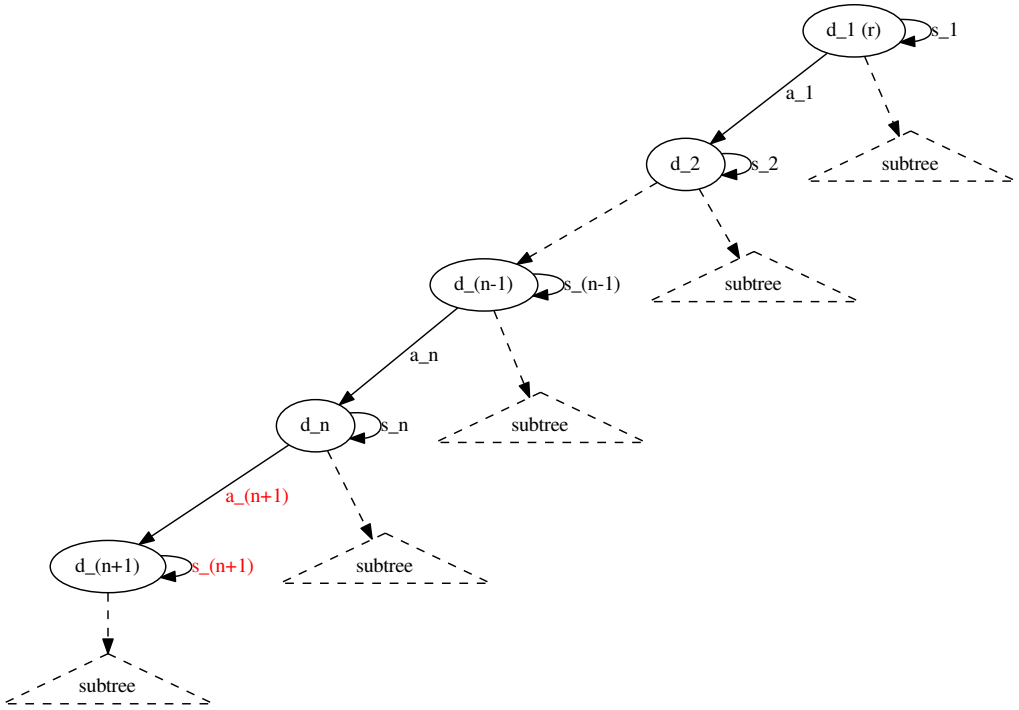

Figure 2.25: Situation at the inductive step, the unknowns are s_(n+1) and a_(n+1)

By inductive hypothesis all the parameters up to depth n are known, without loss of generality we will consider a single branch of the tree.

$a_1 \dots a_n$ and $s_1 \dots s_n$ are the known edges and self loops of the branch up to depth n and we are willing to extract a_{n+1} and s_{n+1}. To extract a_{n+1} we use this formula:

$$\sum_{v \in subtree(v_{n+1})} P(v) = a_{n+1} \Big(\prod_{i=1 \dots n} a_i \Big) \sum_{j=0}^{k-n} ([s_1 \dots s_n], j)$$

This means that the probability of reaching the subtree of v_{n+1} (with v_{n+1} included) in G is equal to that in T. This is due to the graph being a tree with self loops so when a path enters a subtree it cannot leave it.

Finally to compute s_{n+1} we can use $P(v)$ formula:

$$P(v_{n+1}) = \Big(\prod_{i=1 \dots n+1} a_i \Big) \tilde{C}'([s_1 \dots s_{n+1}], k-n-1)$$

we can manipulate this to get an equation of the kind:

$$c = f(s_{n+1})$$

where $f(s_{n+1})$ is a polynomial of grade $k-n-1$ with all positive coefficients, coefficient of the zero grade term equal to 0 and coefficient of the $k-n-1$ grade term 1. In the domain of probabilities $dom = [0,1]$, being f monotone increasing and passing by $(0,0)$ and $(1,m)$ points with $m \geq 1$, there can only be one solution to this equation.

Thus, knowing all the parameters of depth n in a branch of a tree structured generator permits the extraction of all the unknown parameters of depth $n+1$.

□

In the following lemmas we will better describe the effect of the application of a splitting function on the structure.

Lemma 1.

Let S be a structure and f be a splitting function, $f(S)$ is a tree and height[S] $\geq$ height[$f(S)$]

Proof. By the definition of $f(S)$, the resulting graph is connected, acyclic and every node has a single parent with the exception of the root. Hence $f(S)$ is a tree.

In the application of $f(G)$ no height is increased: (by induction of the height of v)

if *height[v]*$= 0$, then v is the root so no operation is executed

if *height[v]*$= n$, then there is at least one already processed ancestor of v, v', with height $n-1$, the ancestor of v with maximum height. Following the application of f, *indegree[v]* nodes $(v_1, \ldots, v_{indegree[v]})$ are produced, each one connected directly with a predecessor of v. Hence

$$\max(height[v_1], \ldots, height[v_{indegree[v]}]) + 1 = n$$

□

Lemma 2.

Let D be a dataset, S be a structure for D and f be a splitting function, $f(S)$ is a structure for D.

Proof. Being S a structure, by Lemma 1, $f(S)$ is a Tree and has a single root. By the properties of the splitting functions, the sum of the probabilities associated with each node obtained from the

splitting of a single node in S is exactly $P(S)$, hence the sum of the probability of all nodes of $f(S)$ is exactly that of S so the property that $(\forall g \in D)P_D(g) = \sum_{x \in \{v \in V_{f(S)} | label[v]=g\}} P(x)$ holds. Finally each node of $f(S)$ is labeled with a genotype.

$f(S)$ satisfies all the properties of a structure for D.

□

Lemma 3.

Let S be a structure and f be a splitting function, then $f(S)$ has the topology of an unravelling tree of S from the root node.

Proof. By the definition of structure, S has a single root and the graph is connected.

We will show how the splitting function application generates a structure with the topology of the unravelling tree of the topology of S from the root node. The proof will proceed by induction on the height at which a node is split:

height 0: the root has no incoming edges, no operation is executed and $f(S) = S$. By the definition of unravelling tree the root is the only element that belongs to the produced topology.

heigth n: without loss of generality we consider a single node v of the DAG at height n. This node has $indegree[v]$ parents, by the inductive hypothesis the topology of the unravelling tree containing all its parents is that of $f(ancestors(v))$. Note that the root always belongs to $ancestors(w)$ for any w except the root itself. Following the procedure to apply the splitting function, $indegree[v]$ copies of v are created, each with a single and different parent from the parents of v. This corresponds to the definition of the unravelling tree for S from the root.

□

Now we use these properties to show that, when a splitting function is applied, there is at most a single solution to our problem.

Corollary 1.

Let D be a dataset, S a non redundant structure for D and f be a splitting function, for all $k \geq heigth[S]$ there exists at most one k-admissible generator G for D with the same topology of $f(S)$.

Proof. By Lemmas 1 and 2 $f(S)$ is a structure with tree topology and $height[S] \geq height[f(S)]$, so by Theorem 4 this corollary is proved.

□

With the following theorem we want to show that, when applying a splitting function, there is at most one possible k-admissible generator capable of generating a given dataset when simulated infinity times with paths of length greater than its height.

Lemma 4.

Let D be a dataset, S be a non redundant structure for D and f be a splitting function. If G is the k-admissible generator for D with the same topology of $f(S)$, then there exists at most one G' non redundant k-admissible generator for D so that $f(G')$ is isomorphic ($\cong$) to G.

Proof. Firstly we define $R_Q(v)$ with $v \in Q$ as the probability of reaching the node v in a simulation of Q, a *k-admissible generator* for D, with paths of length k. We also define $R_Q(g)$ as the probability of reaching any node labelled with genotype $g \in D$ in a simulation of Q with paths of length k.

Being G a *k-admissible generator* for D we have that $(\forall g \in D) R_Q(g) = P_D(g)$ and that $(\forall v \in G) R_Q(v) = \sum_{p \in paths(G,root,v,k)} P_G(p)$, where $paths(G, root, v, k)$ is the set of the paths from $root$ to v in G of length k.

Now we give a procedure *merge* to modify G and produce G' by merging all the nodes of G with the same genotype:

For each set of nodes $v_1, \ldots, v_n \in G$ with the same label add a node v' in G' so that:

- $R_{G'}(v') = \sum_{i=1}^{n} R_G(v_i)$
- $parents[v'] = \bigcup_{i=1}^{n} parents[v_i]$
- $children[v'] = \bigcup_{i=1}^{n} children[v_i]$
- $W(\langle v', a \rangle) = \sum_{i=1}^{n} W(\langle v_i, a \rangle) \frac{R_G(v_i)}{R_{G'}(v')}$ for all nodes $a \in G'$

After this procedure:

- In G' there is a single node for each genotype (G' is non redundant)
- $(\forall g \in D) R_{G'}(g) = R_G(g)$ by the definition of R for genotypes (G' is *k-admissible generator* for D)
- Having G and $f(S)$ the same topology, G has the topology of unravelling of a non redundant structure and being the procedures described here the reverse of that of the unravelling the topology of G' is that of S and the set of paths of G and G' is equal.

Finally, we show that G and G' are indistinguishable by induction on the depth reached from a

path of length k starting from the root:

If depth is 0: the root or its self loop are not modified by this procedure, hence the paths reaching the root are the same in G and G'

If depth is n: Up to depth $n-1$ the paths in G and G' are equal. Having each node v' of depth $n-1$ in G' the union of the adjacency list of each node $v_1, \ldots, v_m$ equal to v' in G, from v' it is possible to reach every node at depth n in G by traversing a single edge.

So G and G' are indistinguishable.

Finally we show that $f(G') \cong G$:

By the definition of $f(S)$ and that of $f(G')$, $f(S)$ has a node for each incoming edge and the same happens with $f(G')$ (excluding the self loops), having S and G' the same nodes and being both non redundant, $f(S)$ and $f(G')$ have the same topology (again with the exception of the self loops).

Being f the same splitting function applied to S and G' the probability of reaching a node of $f(G')$ after k transitions is the same as the observation probability of a node of S in the associated dataset D. By Theorem 4 and Lemmas 1, 2 and 3 there exist only a single generator with tree topology associated to a given tree topology structure of $f(S)$ that is *k-admissible*, but being $f(G')$ and G *k-admissible*:

$$f(G') \cong G$$

□

Definition 11 (**Merge procedure**)**.**

Let G be a generator, we define as merge(G) the homonym procedure defined in the proof of Lemma 4. This procedure returns a non redundant generator that has the same paths and the same probability of reaching a genotype after k transitions in G.

This procedure is basically the opposite of the splitting application.

With this theorem we close our demonstration in the simplified case of trees, note that by CIMICE(D) we refer to the gaph returned in output by CIMICE when using any splitting function and D as dataset.

Theorem 5.

Let G be a non redundant generator of height h and D be a dataset generated from G using paths of

length $k \geq h$ in which each genotype is represented with the probability of reaching it in the generator in k transitions. If G is a k-admissible generator for D and has a tree structure, CIMICE(D) returns a weighted structure T so that $T \cong$ untimed(G).

Proof. By the definition of G and by the hypothesis on the dataset D, CIMICE(D) has the same topology of G but without any self loop; this is also the topology of *untimed(G)*.

Since *untimed(G)* has a tree topology, the weight of any outcoming edge e of a node is the proportion of the sum of the $P(v)$ of any node v reachable passing through e. Being each genotype represented with the probability of reaching it in the generator in k transitions and having G a tree structure the weights of T are exactly the same of the weights of *untimed(G)* being $P(g)$ equal in both G and D.

Being the set of nodes, edges and weights equal for T and *untimed(G)*, then $T \cong$ *untimed(G)*.

□

Here we extend the last result to the general case, note that with CIMICE(f, D) we refer to the gaph returned in output by CIMICE when using f as splitting function and D as dataset.

Theorem 6.

Let G be a generator, $k \geq height[G]$, f a splitting function and D a dataset where $P_D(g)$ is equal to the probability of reaching g after k transitions. CIMICE(f, D) using f as splitting function returns untimed($merge(f(G))$)

Proof. Being dataset D a representation of $P_G(g)$ the structure obtained by CIMICE, T, has the same topology of G (with the exception of self loops). The weight set on each edge $\langle a, b\rangle$ of T by our method is, by definition:

$$W_{CIMICE}(\langle a, b\rangle) = \frac{\sum_{v \in subDAG(b)} P_D(v) \prod_{p \in path(a,v)} f(p)}{\sum_{v \in subDAG(a)} P_D(v) \prod_{p \in path(a,v)} f(p)}$$

note that, by hypothesis, this is also equal to:

$$W_{CIMICE}(\langle a, b\rangle) = \frac{\sum_{v \in subDAG(b)} P_G(v) \prod_{p \in path(a,v)} f(p)}{\sum_{v \in subDAG(a)} P_G(v) \prod_{p \in path(a,v)} f(p)}$$

Generator G is split with function f and then $merge(f(G))$ is computed. This leads to a generator that is $k - admissible$ for D with the same topology of G and $P_G(g) = P_{merge(f(G))}(g)$ for any g. Also note that $merge(f(G))$ is unique for any G. By the definition of the *untime* procedure each self loop is removed and:

$$W_{untimed}(\langle a,b\rangle) = \frac{W(\langle a,b\rangle)}{1 - W(\langle a,a\rangle)}$$

Changing the point of view to that of probabilities, the probability of traversing an edge $\langle a,b\rangle$ is the probability of reaching b when taking a transition from a, excluding $\langle a,a\rangle$, in a simulation of $merge(f(G))$. Being $merge(f(G))$ *k-admissible* for G this probability is exactly:

$$W_{untimed}(\langle a,b\rangle) = \frac{\sum_{v \in subDAG(b)} P_G(v) \prod_{p \in path(a,v)} f(p)}{\sum_{v \in subDAG(a)} P_G(v) \prod_{p \in path(a,v)} f(p)}$$

so

$$W_{untimed}(\langle a,b\rangle) = W_{CIMICE}(\langle a,b\rangle)$$

and, having $untimed(merge(f(G)))$ the same topology of CIMICE(f, D):

$$untimed(merge(f(G))) \cong \text{CIMICE}(f,\ D)$$

□

Finally, at the end of this section, we proved that our method returns the untimed vesrsion of a generator that could actually have produced the observed dataset.

2.5.3 Example

To clarify Theorem 6 we give a simple example showing all the steps needed to obtain $untimed(merge(f(G)))$ and CIMICE(f, D). We will use a simplified splitting function that divides a node equally on its incoming edges.

The input dataset for CIMICE has these proportions:

```
clonal: 0,1%
A     : 4,2%
B     : 9,3
AB    : 43,5%
ABC   : 11,7%
AD    : 31,2%
```

that were computed from the generator G (figure 2.26). In the figures 2.27, 2.28 and 2.29 are shown

the results of the subsequent application of splitting function, merging and untiming procedures, while figures 2.30 and 2.31 show the two steps of computation made by CIMICE. This should also point out that the splitting function is just a parameter of our tool.

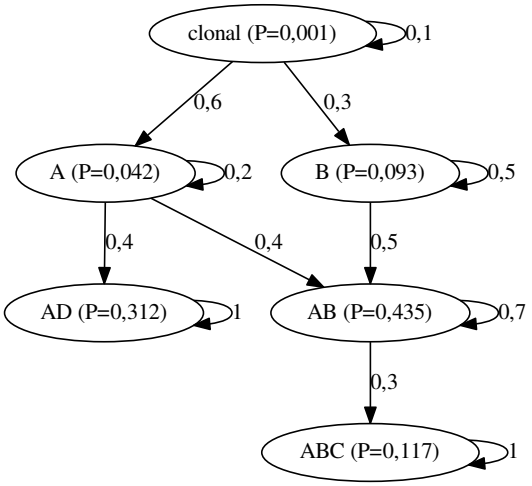

Figure 2.26: the starting generator G is known by hypothesis

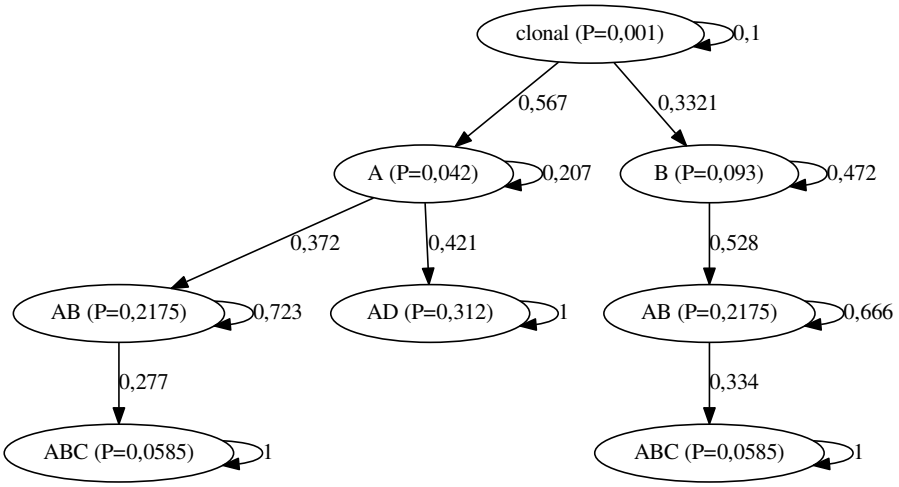

Figure 2.27: application of the splitting function to G, $f(G)$

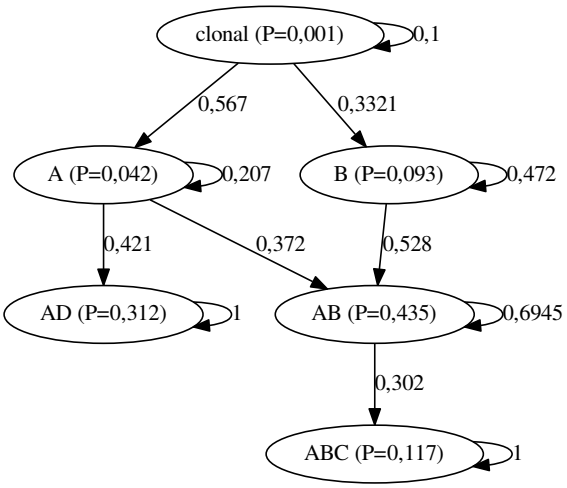

Figure 2.28: $f(G)$ is merged to obtain $merge(f(G))$

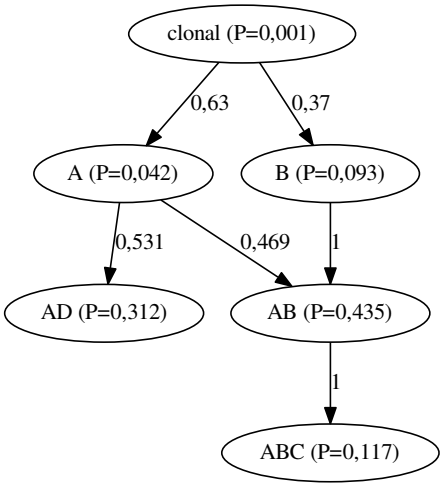

Figure 2.29: untiming operation application, $untimed(merge(f(G)))$

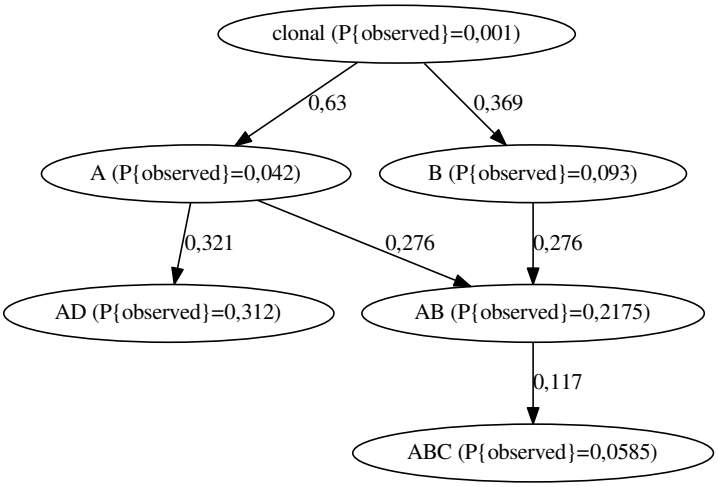

Figure 2.30: First step of CIMICE execution, computation of downWeights after splitting function application

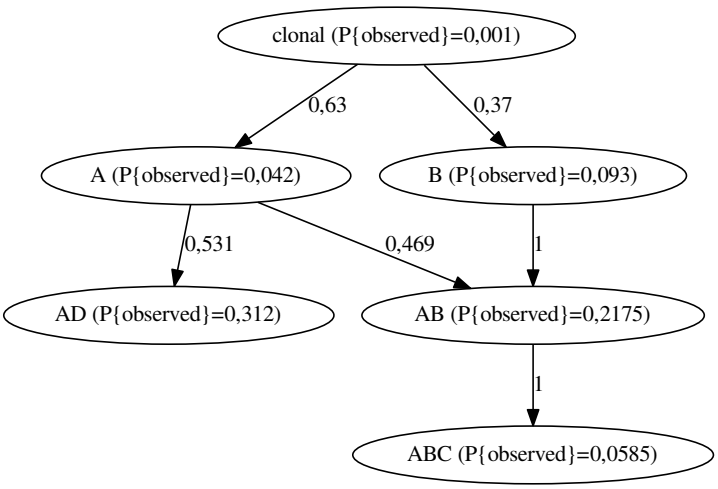

Figure 2.31: normalization step, this would be the final output of CIMICE

3

Comparison with TRONCO tools on synthetic data

Here we show the results obtained from many artificial datasets by using our reconstruction method and the tools of the TRONCO pipeline. The samples are generated by random walks on the generator graphs and the dataset are based on two parameters: n, the number of samples, and k, the length of the random walks. Note that in our model an edge means that it is possible to pass from a state (genotype) to another, in the TRONCO tools it means that a mutation on the source gene can be the cause of the appearance of another one on the destination gene. The weight of the edges in the models reconstructed by our tool follow an heuristic and so they are not expected to converge to the true ones that are indicated in blue in each generator. This is due to the fact that we don't check if assumption number 3 is respected by the generator, however, in example number 3 this assumption holds being the generator a tree and, in fact, the inferred transition probabilities converge to the generator's ones.

3.1 EXAMPLE 1

As a first example we take a generator with a rather simple structure and with self loops of equal probability, as seen in figure 3.1. It is visible that our method needs at least 100 samples and simulation paths of length 18 (figure 3.2) to fully reconstruct the structure of the generator and increasing sampling makes possible to reduce the necessary k (figure 3.3). The TRONCO tools result (figure 3.4) is quite consistent with our reconstruction, mutations on C can only come from A's and D's are most likely to come after B's.

3.1.1 Generator model

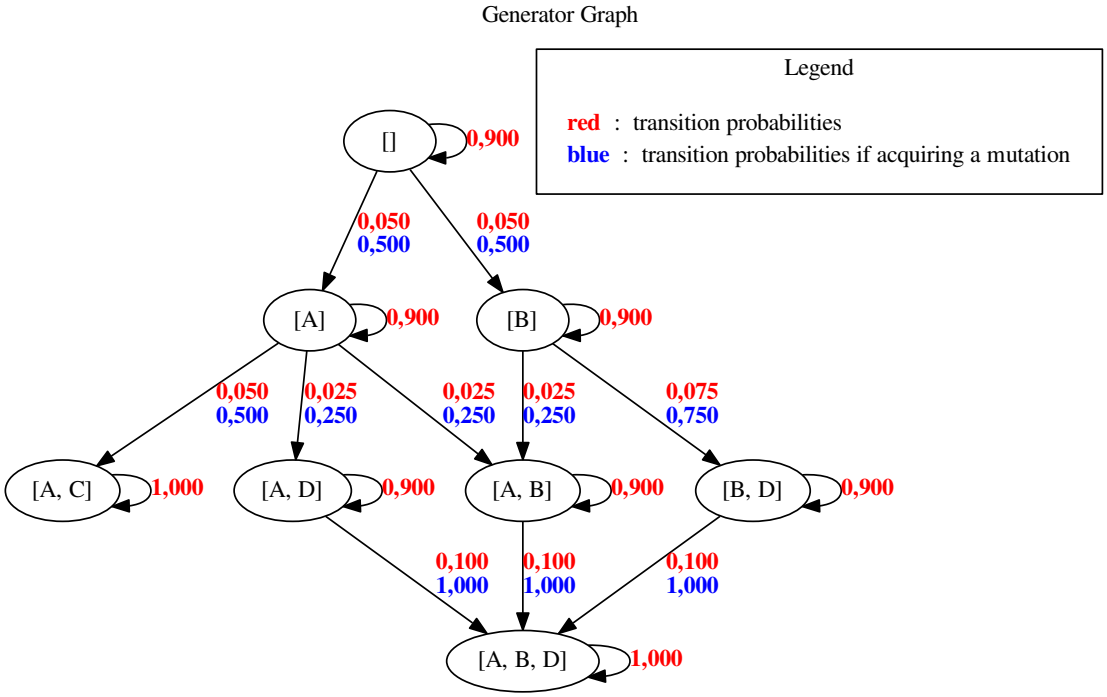

Figure 3.1: self loops have the same probability: 0.9

3.1.2 Our reconstruction

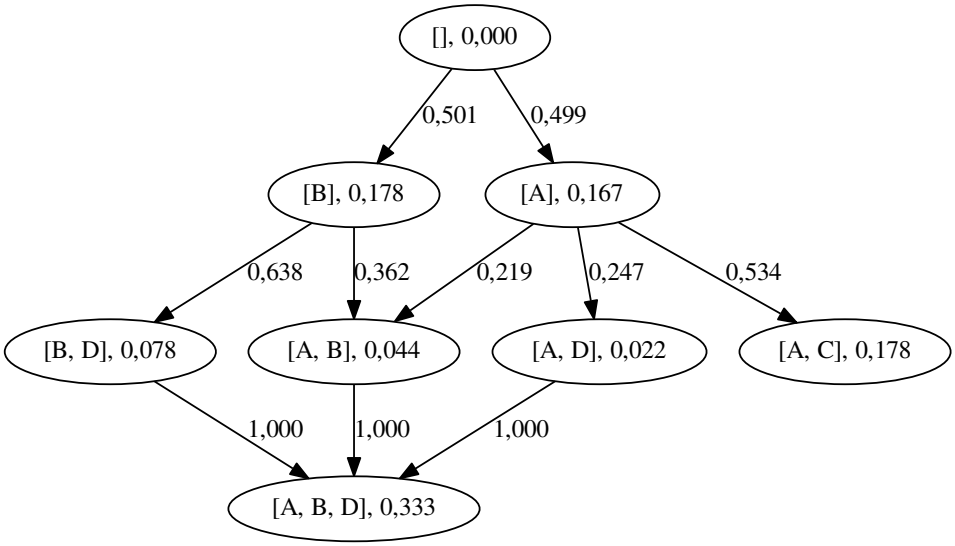

Figure 3.2: $n = 100$, $k = 18$

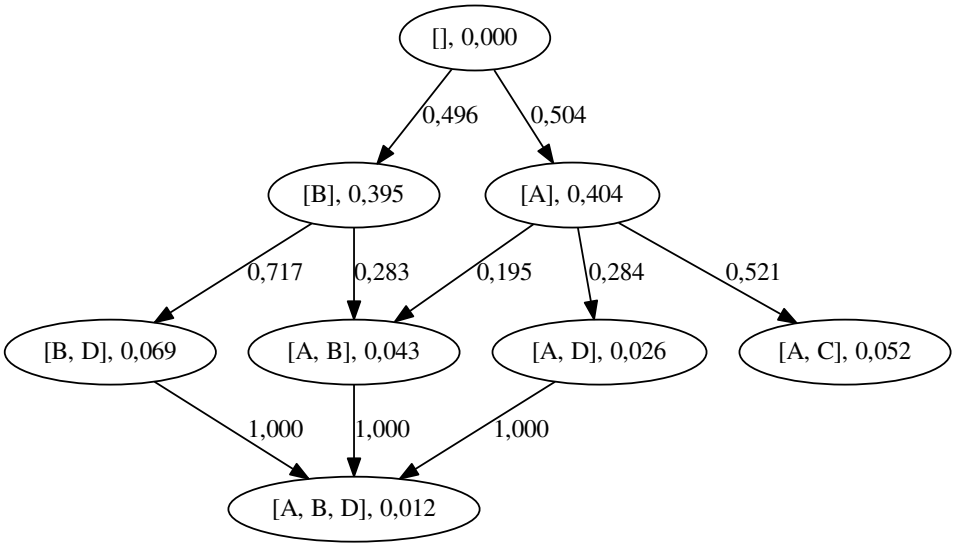

Figure 3.3: $n = 1000$, $k = 5$

3.1.3 TRONCO suite

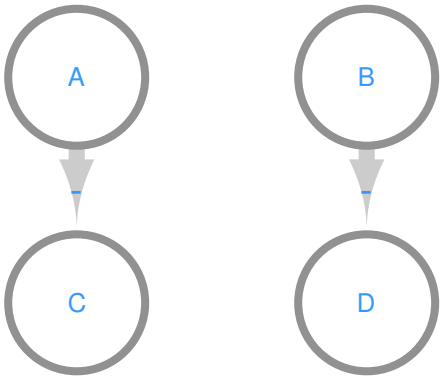

Figure 3.4: the result given by any TRONCO routine when given enough samples obtained with sufficiently long walks

3.2 EXAMPLE 2

In this example self loops have different probabilities (see 3.1), the results are consistent with that of example 1 (figures 3.6, 3.7 and 3.8).

3.2.1 Generator model

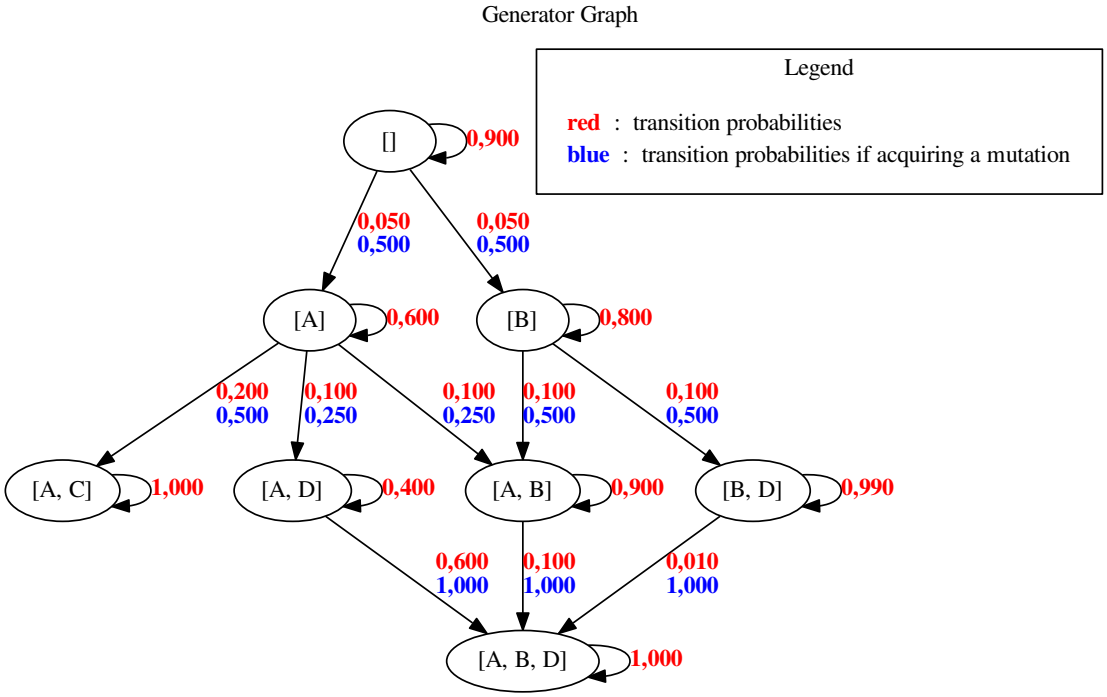

Figure 3.5: This generator is meant to have different self loop probabilities

3.2.2 Our reconstruction

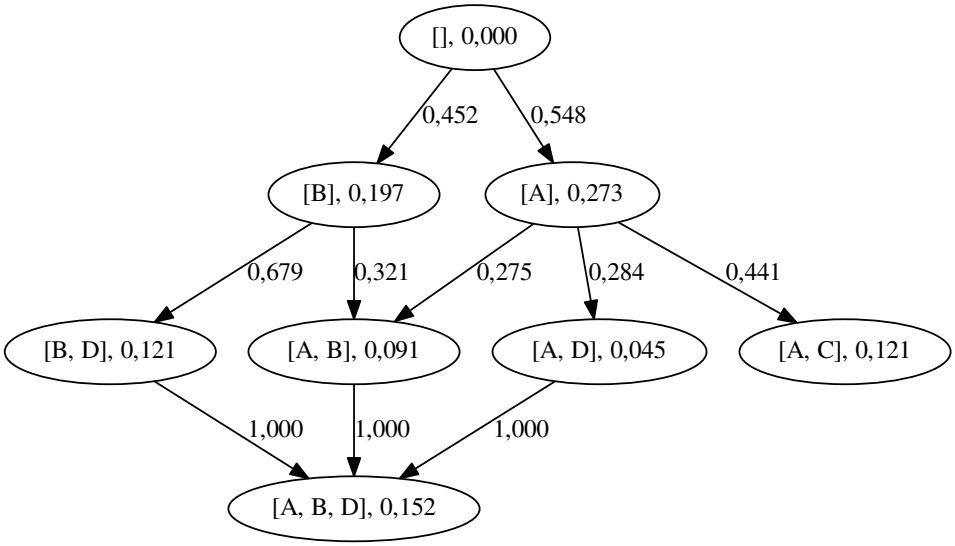

Figure 3.6: $n = 100$, $k = 11$

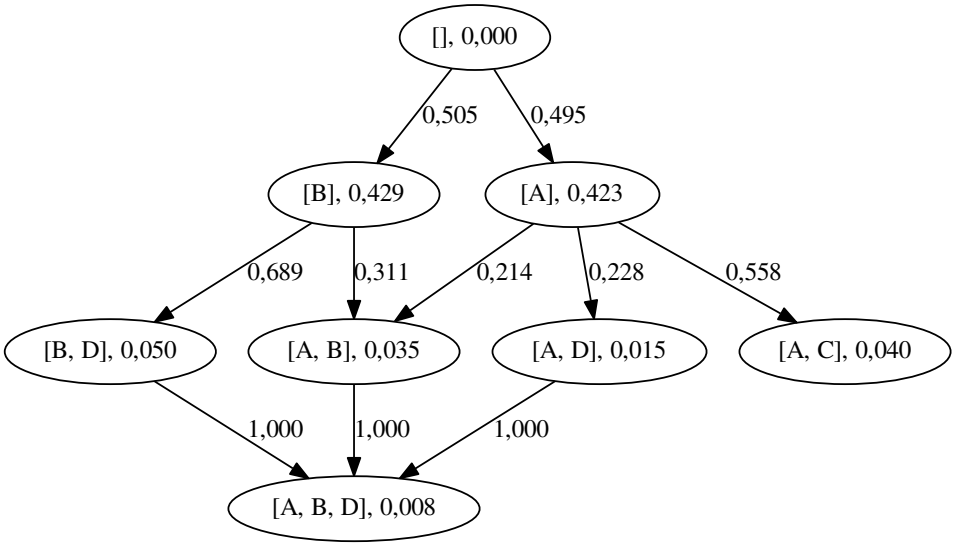

Figure 3.7: $n = 10000$, $k = 4$

3.2.3 TRONCO suite

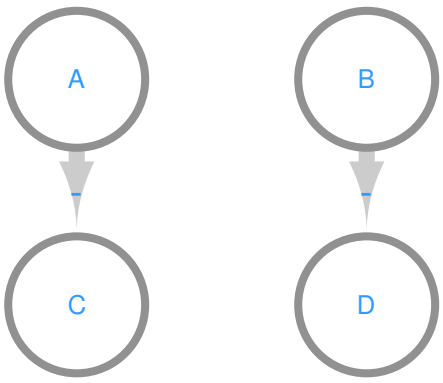

Figure 3.8: the result of the TRONCO tools does not change

3.3 EXAMPLE 3

In this example we prepared a tree generator to show how our method converges to the real probabilities of the generator (see figure 3.9, the blue probabilities). To this end we show how the approximation on the weights reduces increasing the number of samples (figures 3.10, 3.10, 3.11, 3.12, 3.13 and 3.14). It is interesting to see that A is not present in the reconstruction of the TRONCO tools (figure 3.15).

3.3.1 Generator model

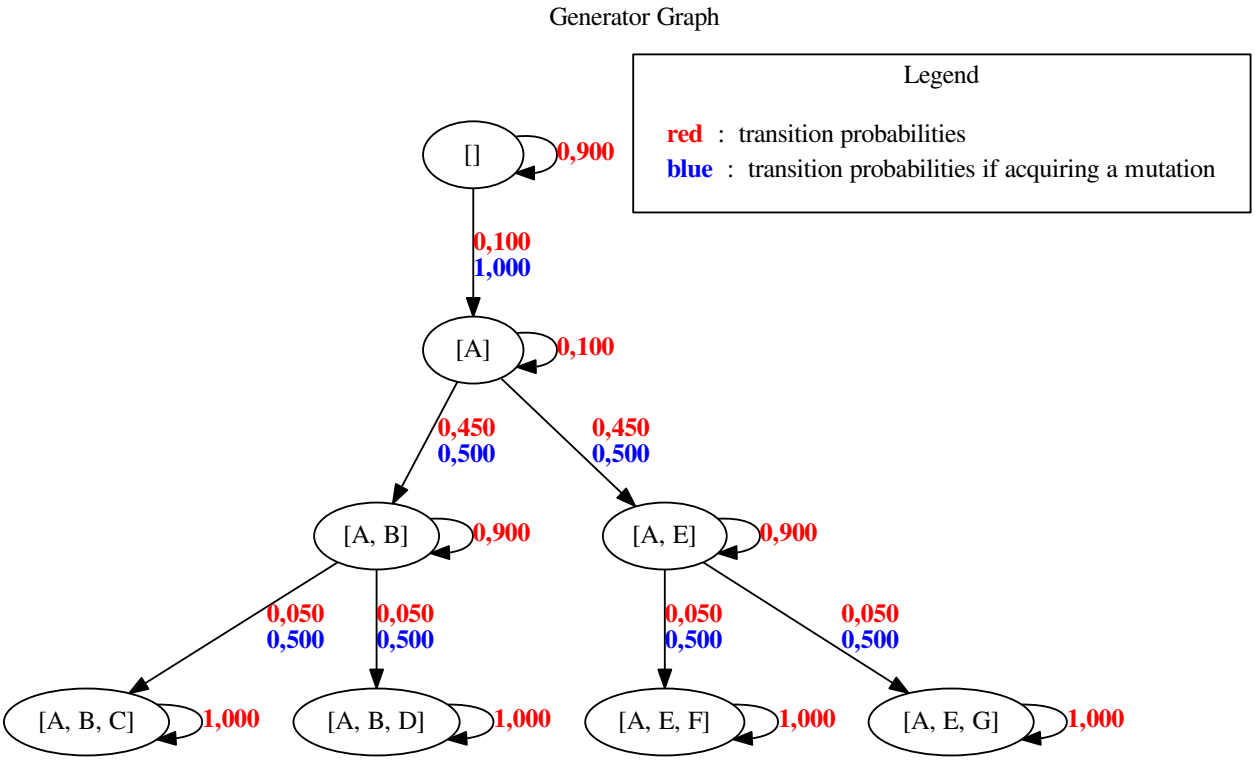

Figure 3.9: In this case our aim is to generate an easy case for the reconstruction algorithms

3.3.2 Our reconstruction

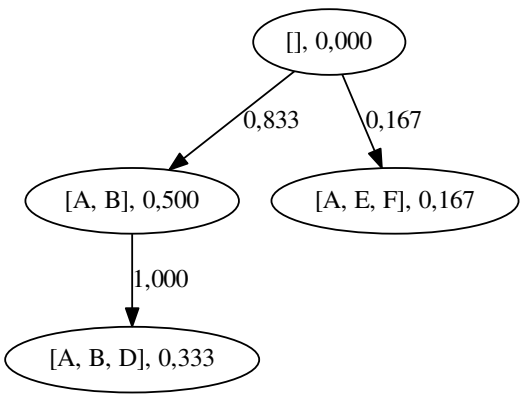

Figure 3.10: $n = 10$, $k = 8$, sampling is too low to represent each genotype

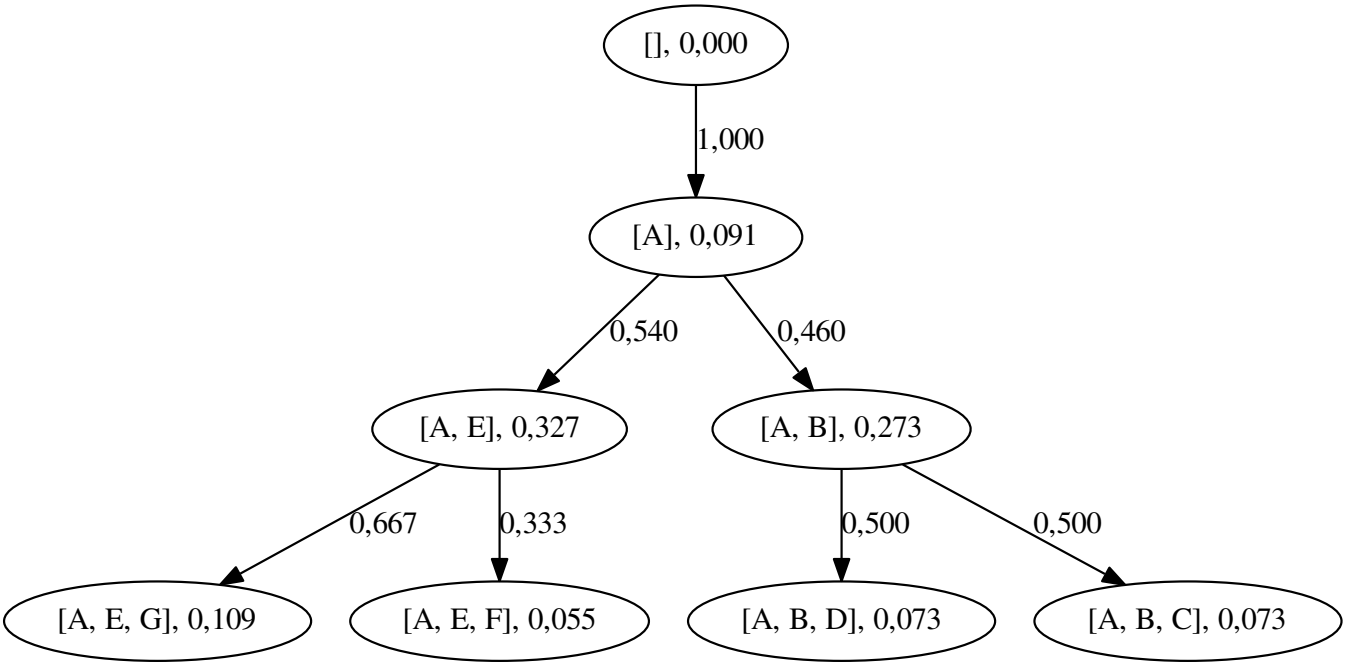

Figure 3.11: $n = 100$, $k = 8$

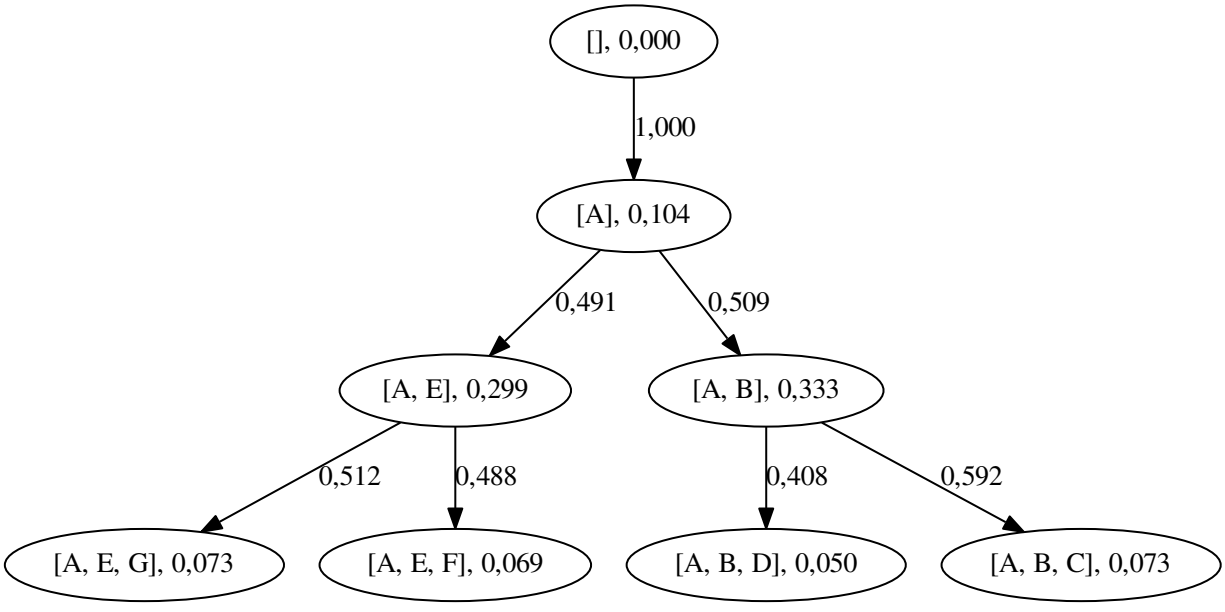

Figure 3.12: $n = 1000$, $k = 8$, it can be seen how the increased sampling improves the accuracy of the inferred transition probabilities

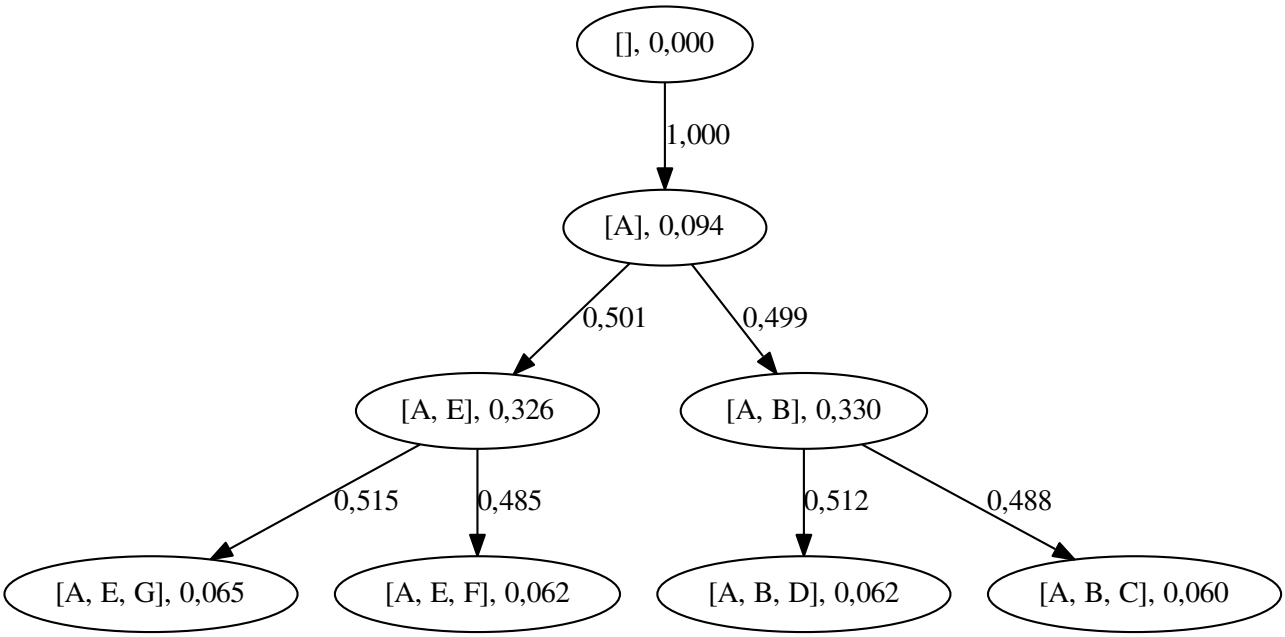

Figure 3.13: $n = 10000$, $k = 8$

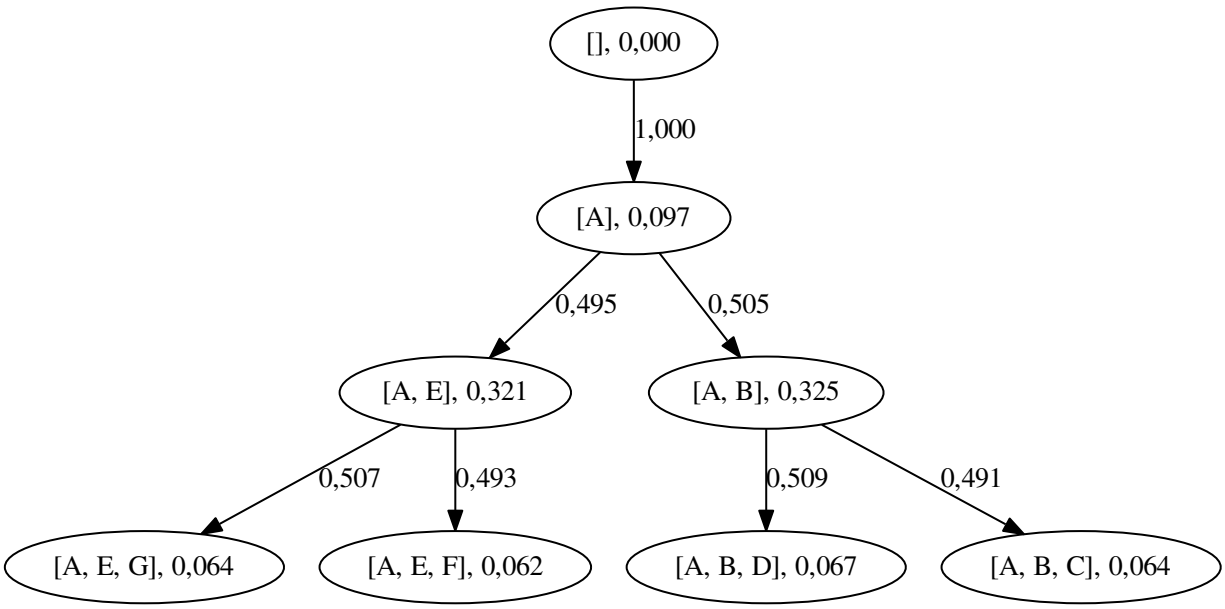

Figure 3.14: $n = 100000$, $k = 8$

3.3.3 TRONCO suite

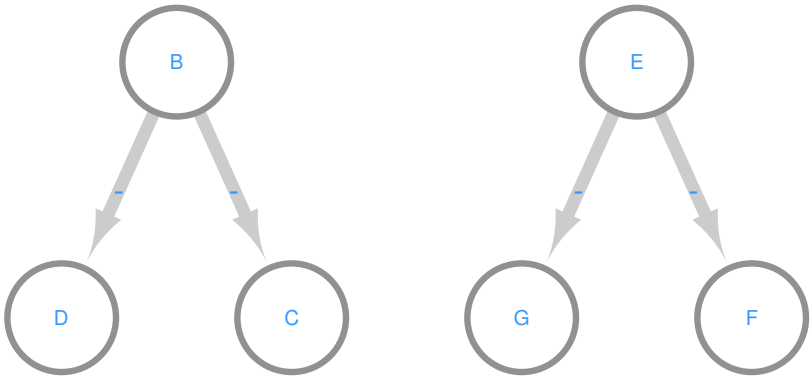

Figure 3.15: again the results from the TRONCO are similar, it is questionable why A is not represented being the driver mutation

3.4 EXAMPLE 4

In this case we show how the methods cope with larger graphs (see 3.16), like in case 1 and 2 we show an example with equal self loop probabilities and one with random ones. The considerations on our method remain the same, with an increased number of possible samples is reasonable that sampling must be increased (figures 3.17, 3.18 and 3.19).

3.4.1 Generator model

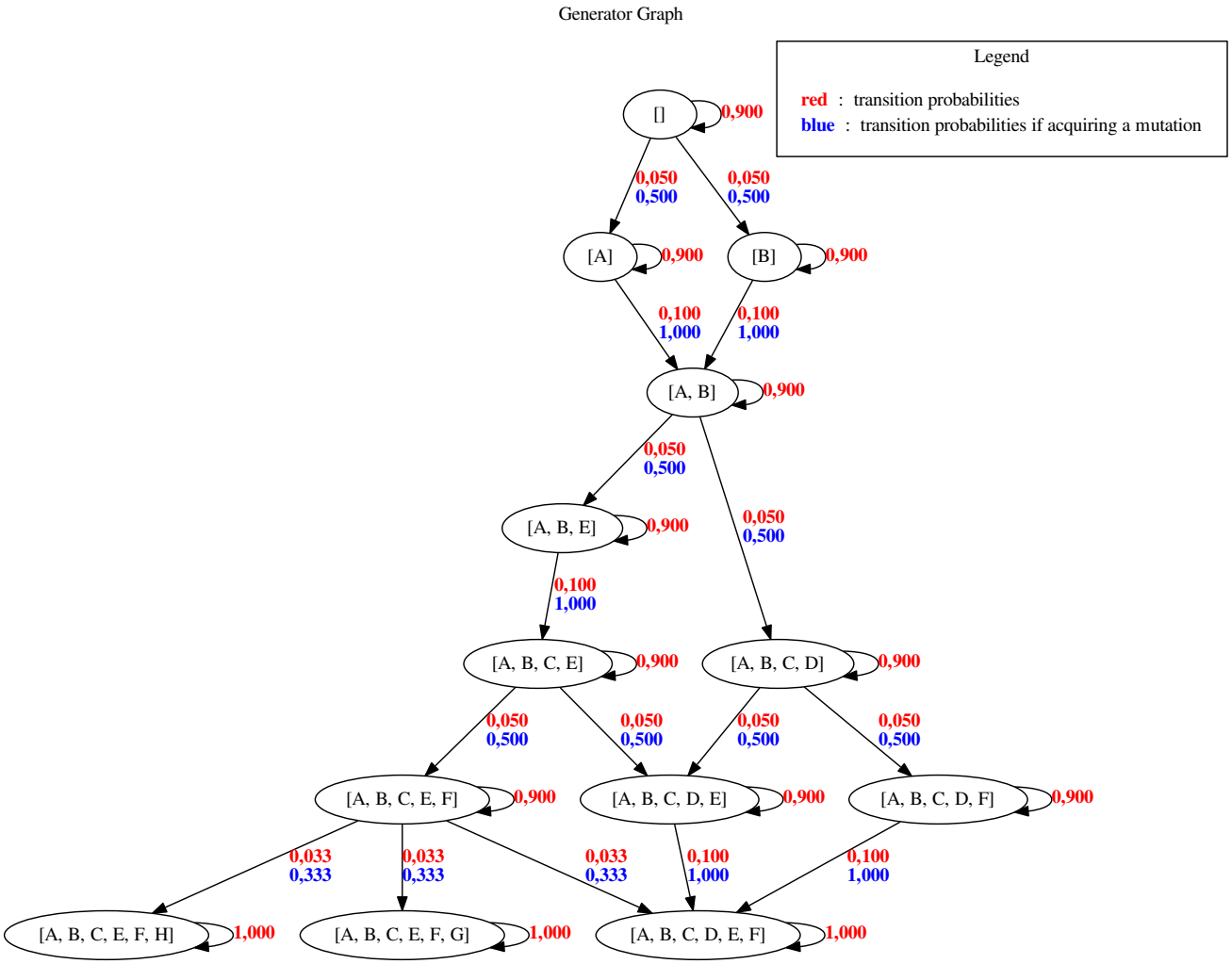

Figure 3.16: all self loops have 0.9 probability to be taken on each step of any random walk

3.4.2 Our reconstruction

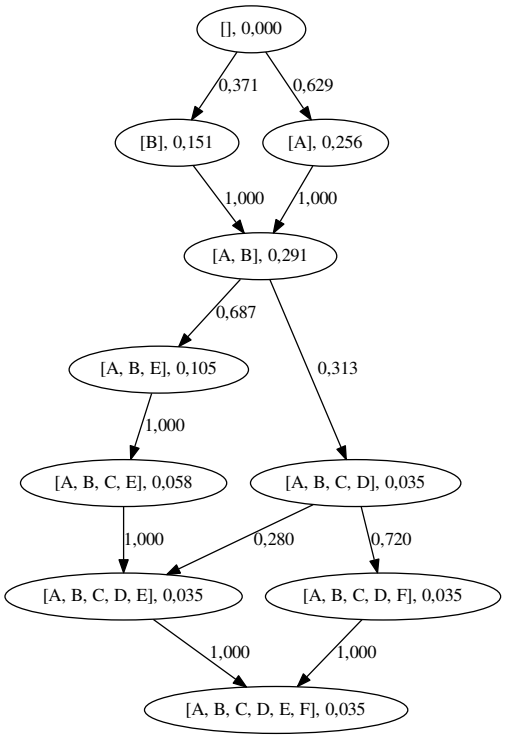

Figure 3.17: $n = 100$, $k = 21$, some late genotpes are missing

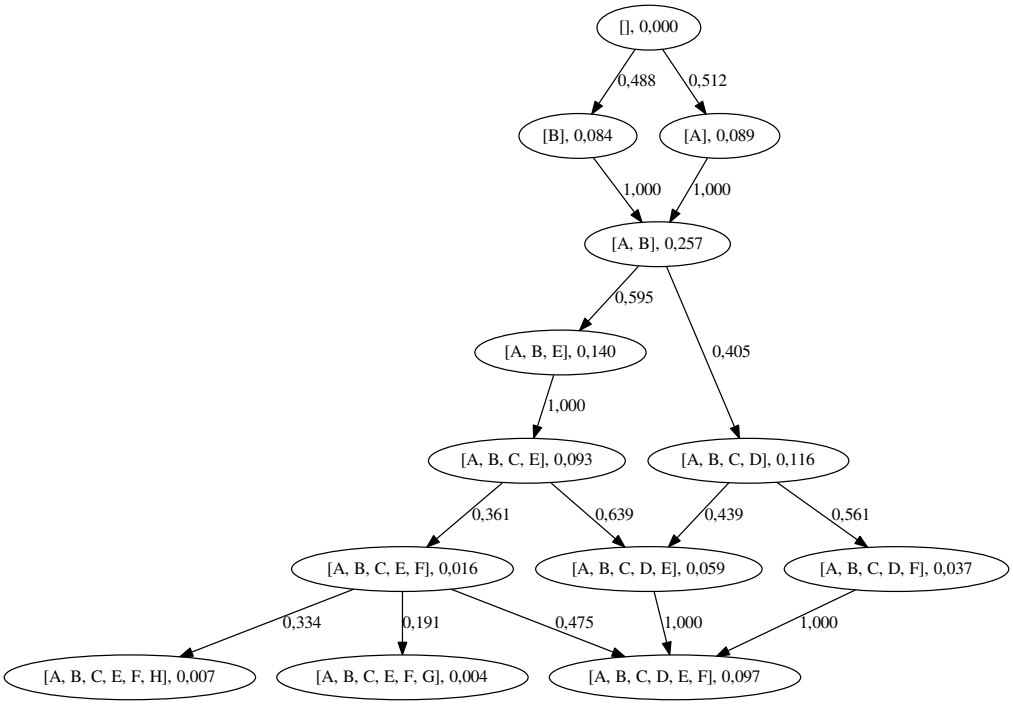

Figure 3.18: $n = 1000$, $k = 28$, increasing sampling and walk length all genotypes are reached

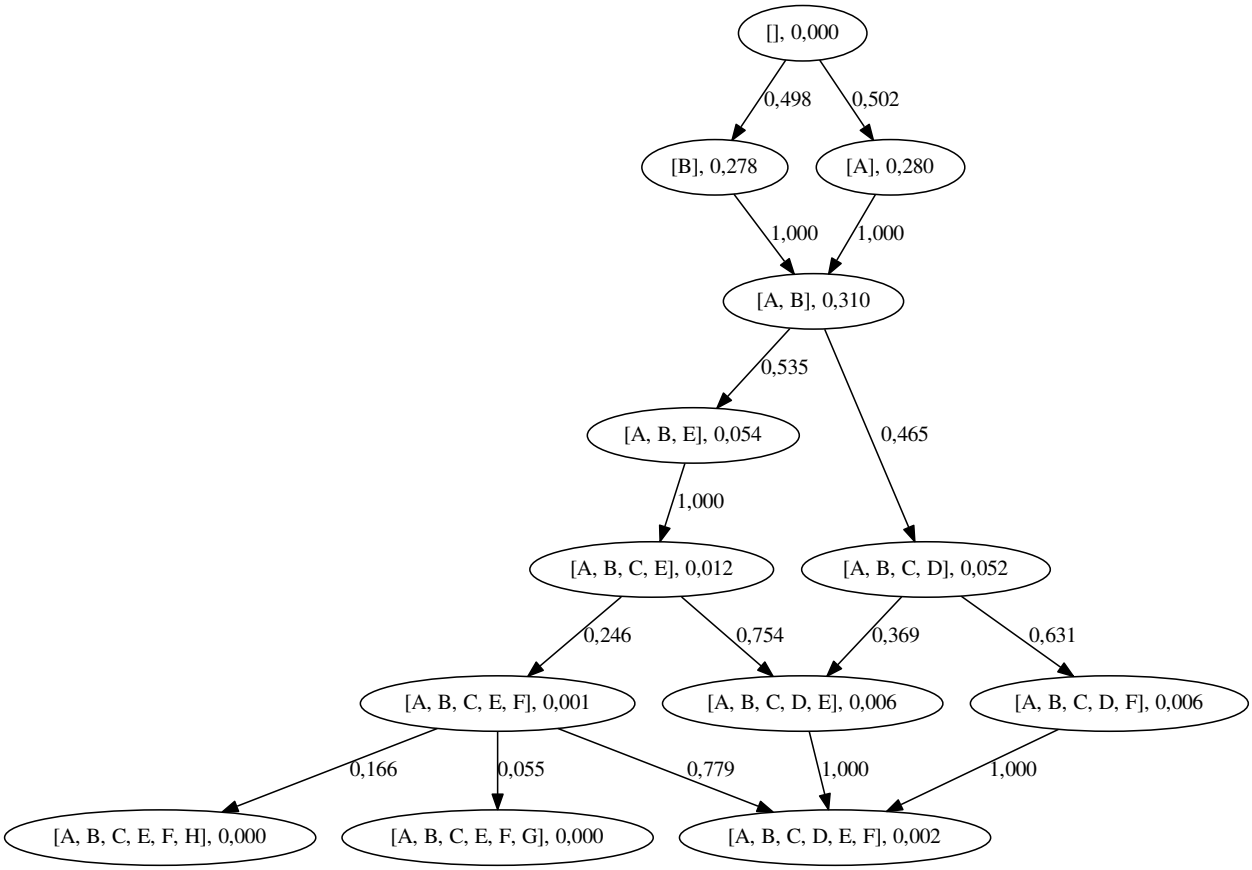

Figure 3.19: $n = 100000$, $k = 11$, increasing sampling allows to reduce the length of random walks

3.4.3 TRONCO suite

In this case the tools using trees (from figure 3.20 to 3.24) and DAGs (figures 3.25 and 3.25) start to show different behaviors.

Trees: CAPRESE, Prim, Gabow, Edmonds

Figure 3.20: $n = 100$, $k = 21$ (Gabow/Edmonds)

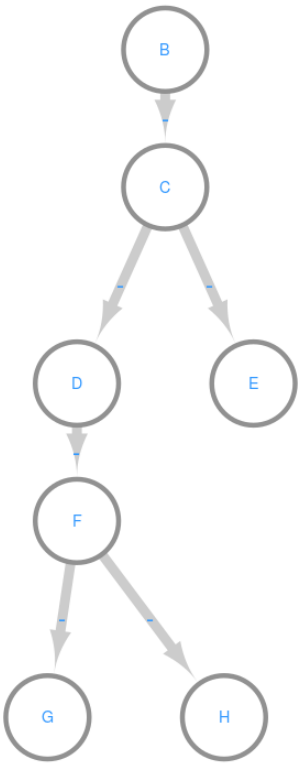

Figure 3.21: model with an high ($n = 100000$) number of samples (Gabow/Edmonds)

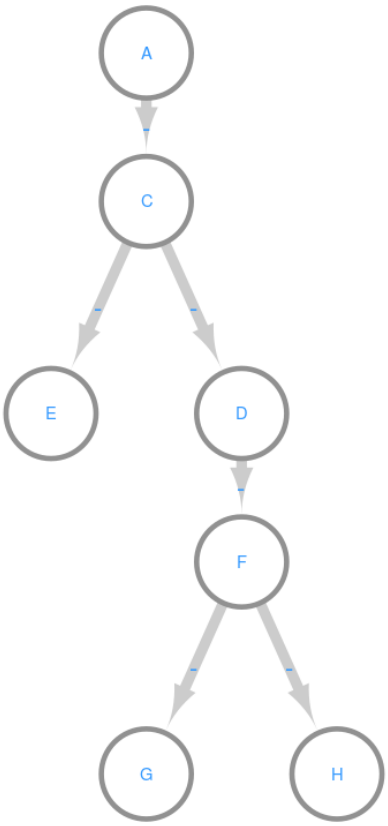

Figure 3.22: alternate tree that shows a different root (Gabow/Edmonds)

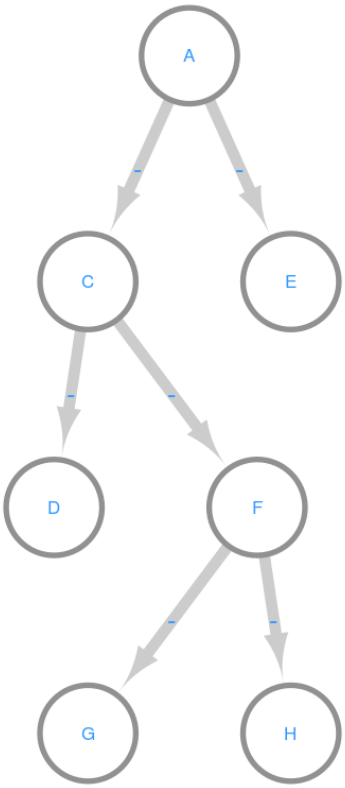

Figure 3.23: the tree from Caprese manages E differently (high number of samples)

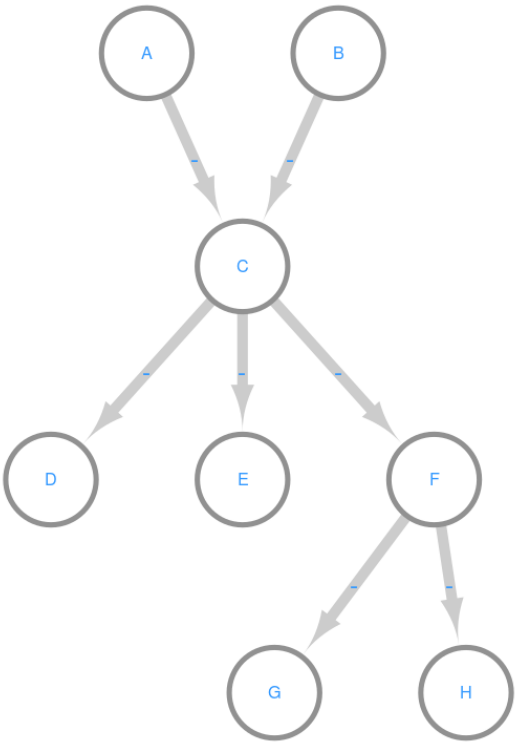

Figure 3.24: the tree from Prim finds out that A and B have the same role (high number of samples)

DAGs: CAPRI and ChowLiu

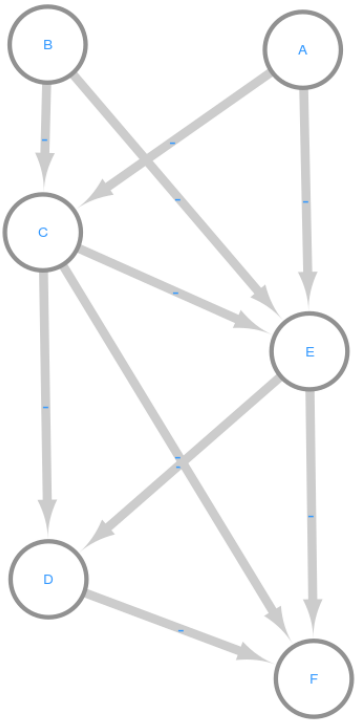

Figure 3.25: with a low ($n = 1000$) number of samples (ChowLiu/CAPRI)

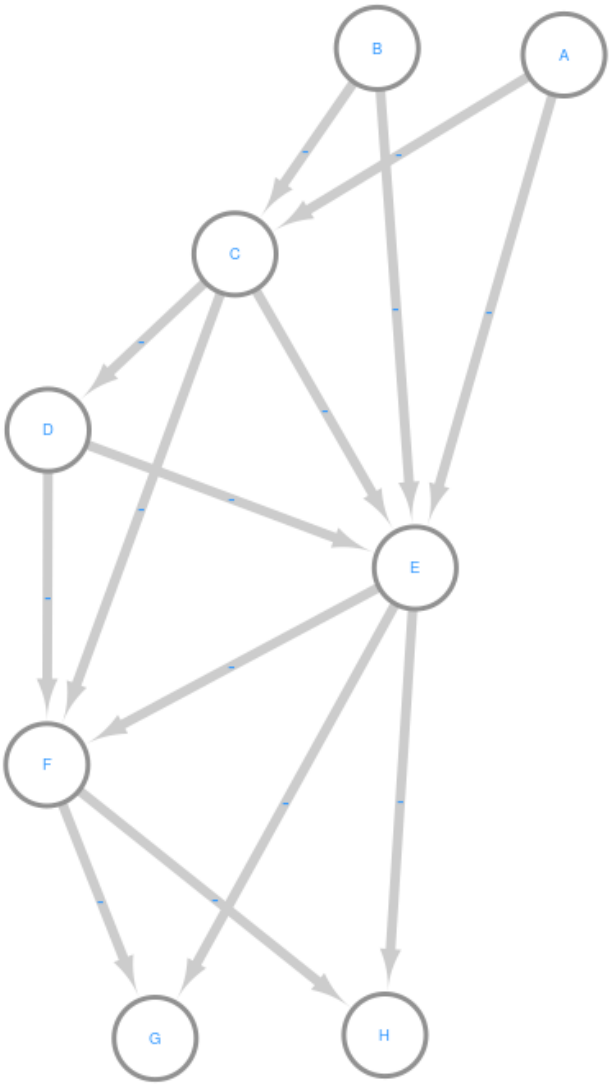

Figure 3.26: with an higer ($n = 100000$) number of samples (ChowLiu/CAPRI)

3.5 EXAMPLE 5

Same structure of example 5 but with random self loop probabilities (see generator 3.27).

3.5.1 Generator model

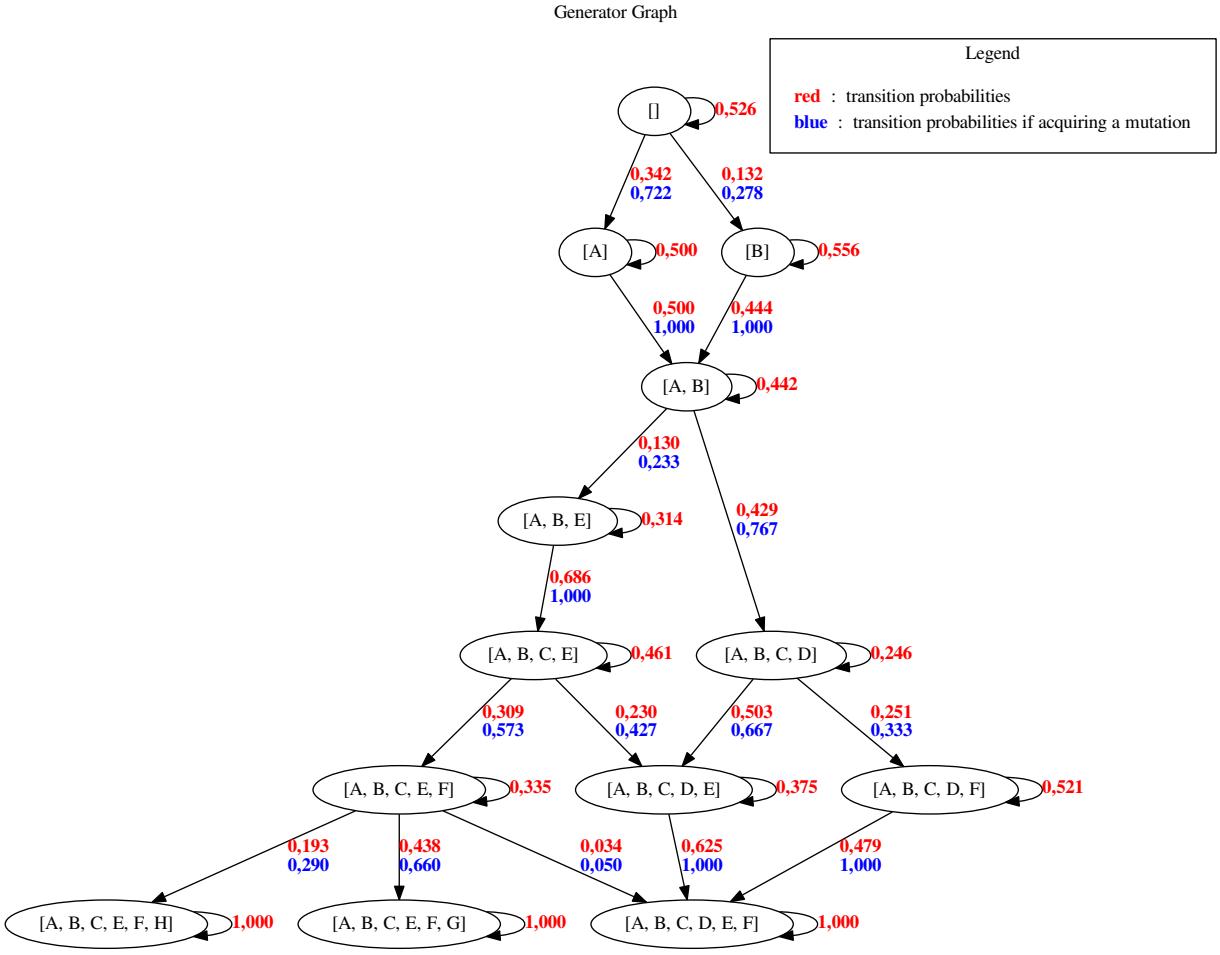

Figure 3.27: the structure is the same as in the section before

3.5.2 Our reconstruction

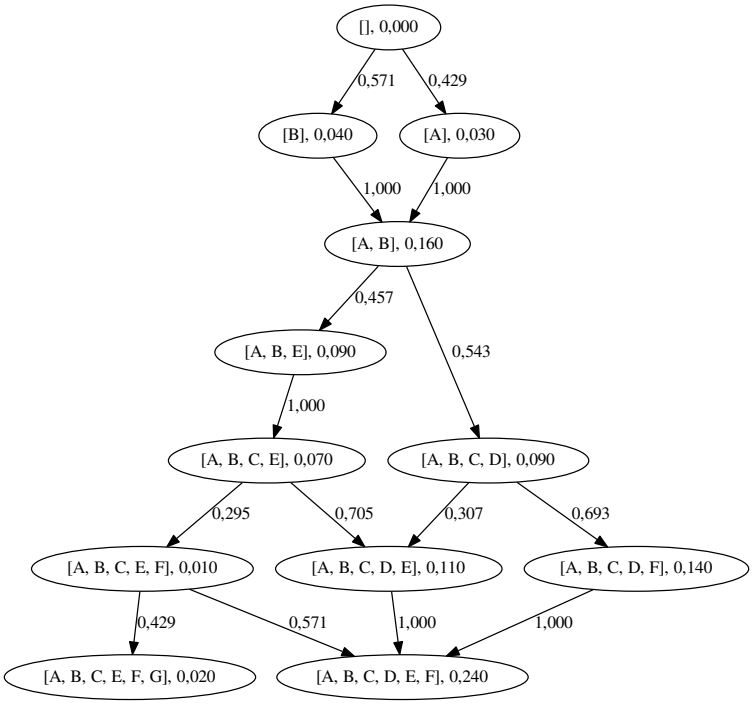

Figure 3.28: $n = 100$, $k = 7$

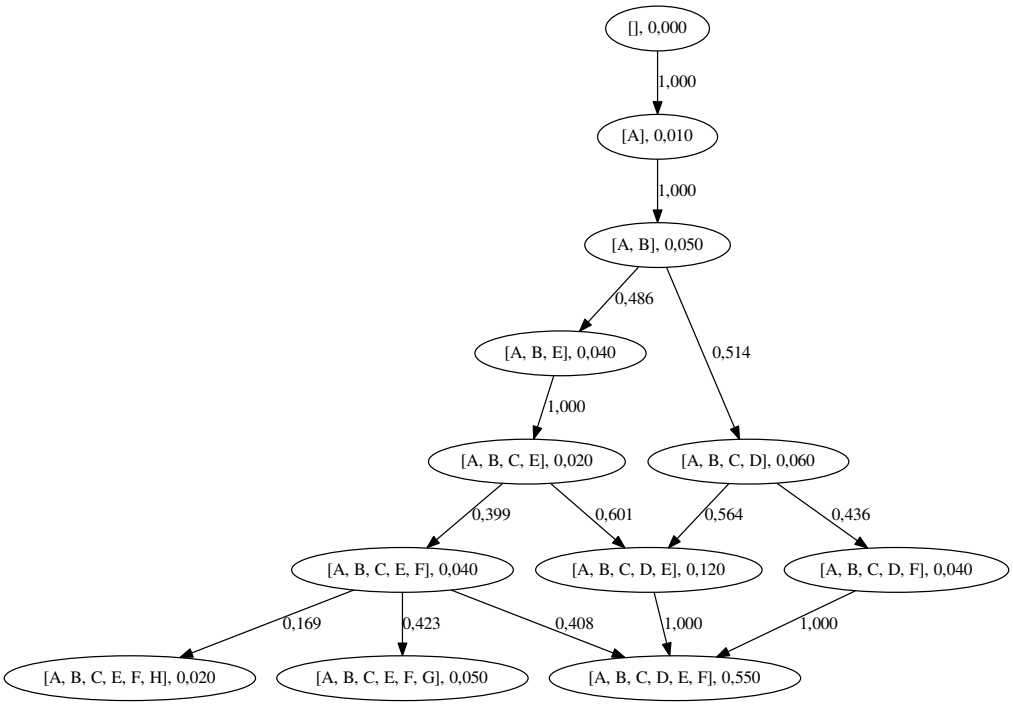

Figure 3.29: $n = 100$, $k = 9$, by extending the length of the random walks we gain information on the later nodes sacrificing that on the earliest ones

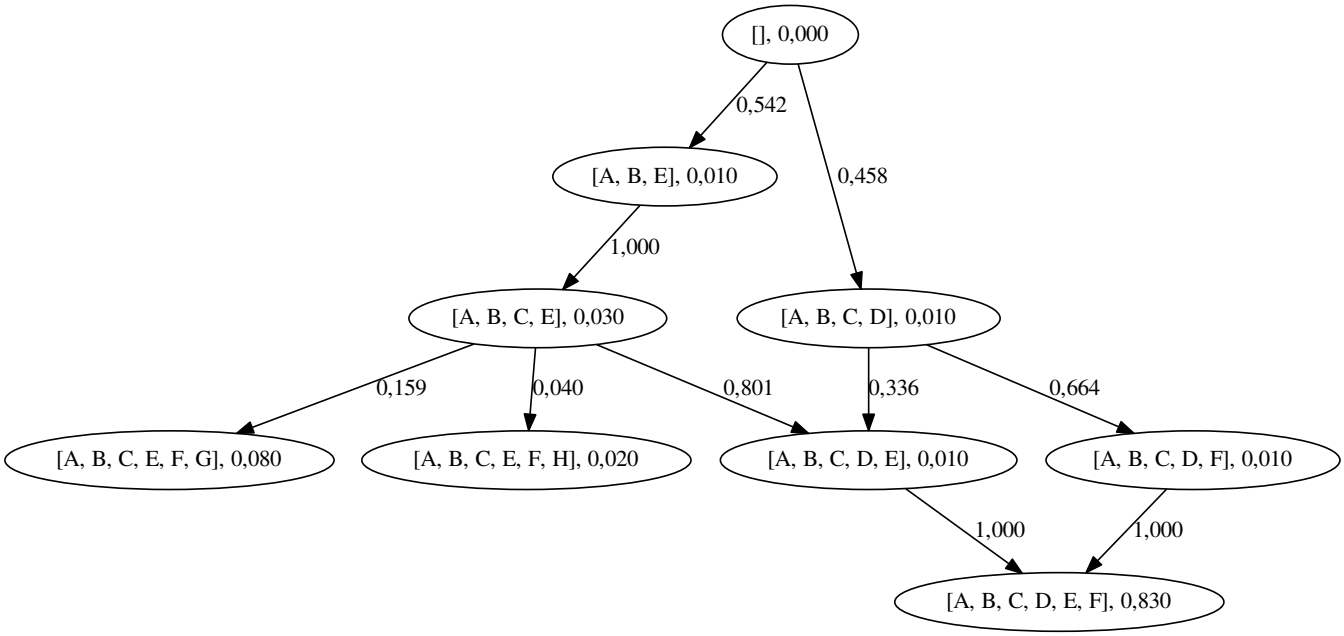

Figure 3.30: $n = 100$, $k = 13$, extending it again the early history is lost

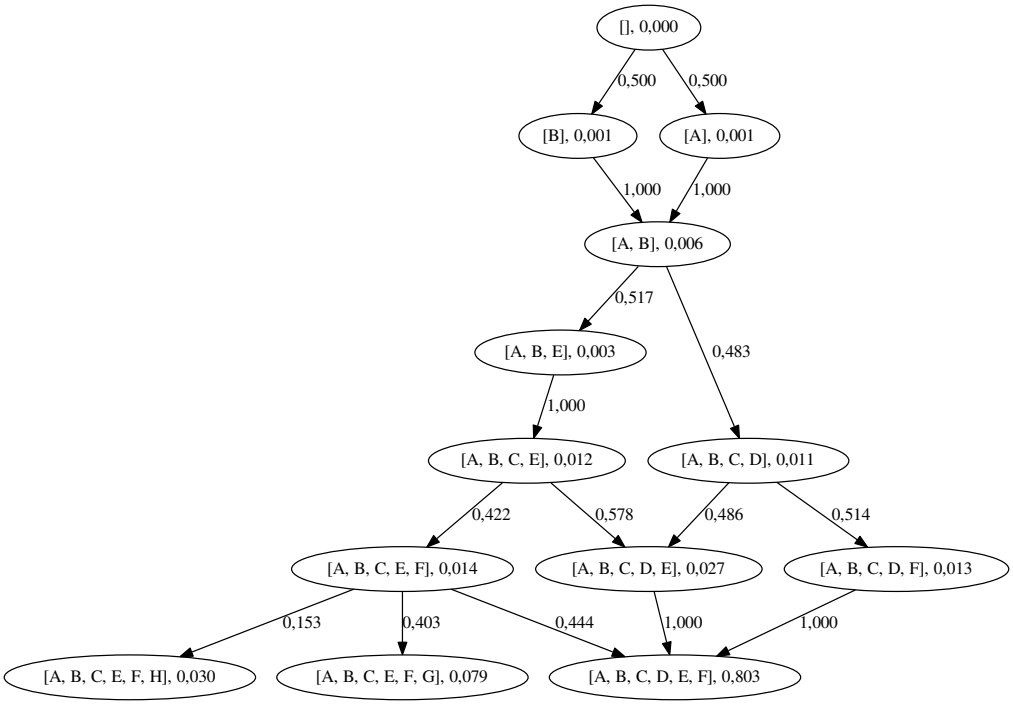

Figure 3.31: $n = 1000$, $k = 13$, adding samples permits to find every genotype even with long walks

3.5.3 TRONCO suite

Only Gabow, ChowLiu and CAPRI shown differences with these different transition probabilities.

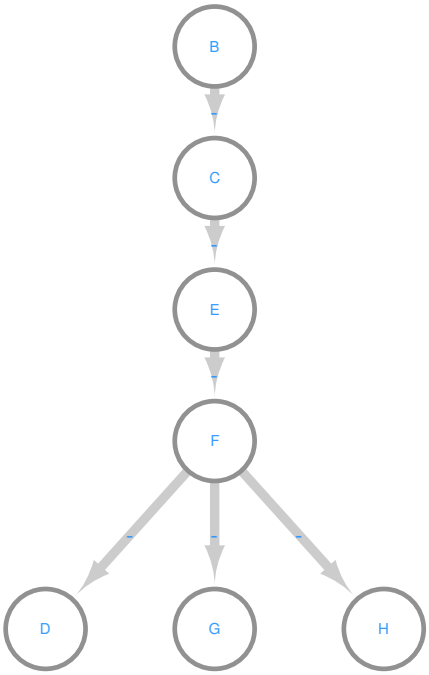

Figure 3.32: Gabow $n = 100000$, $k = 13$

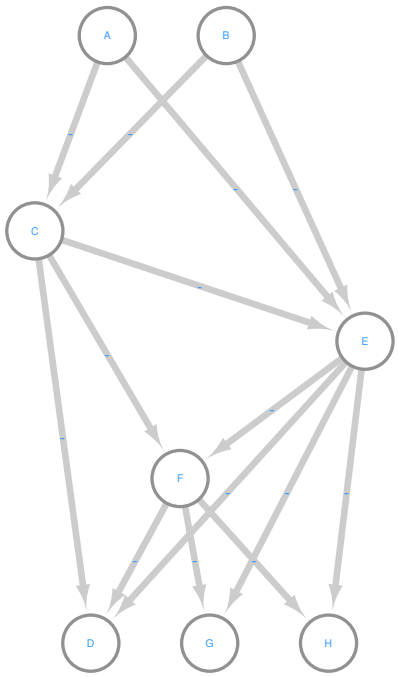

Figure 3.33: ChowLiu/CAPRI $n = 100000$, $k = 13$

3.6 EXAMPLE 6

In this case we decided to produce a very complex example by creating the graph structure and weights completely at random (figure 3.34).

3.6.1 Generator model

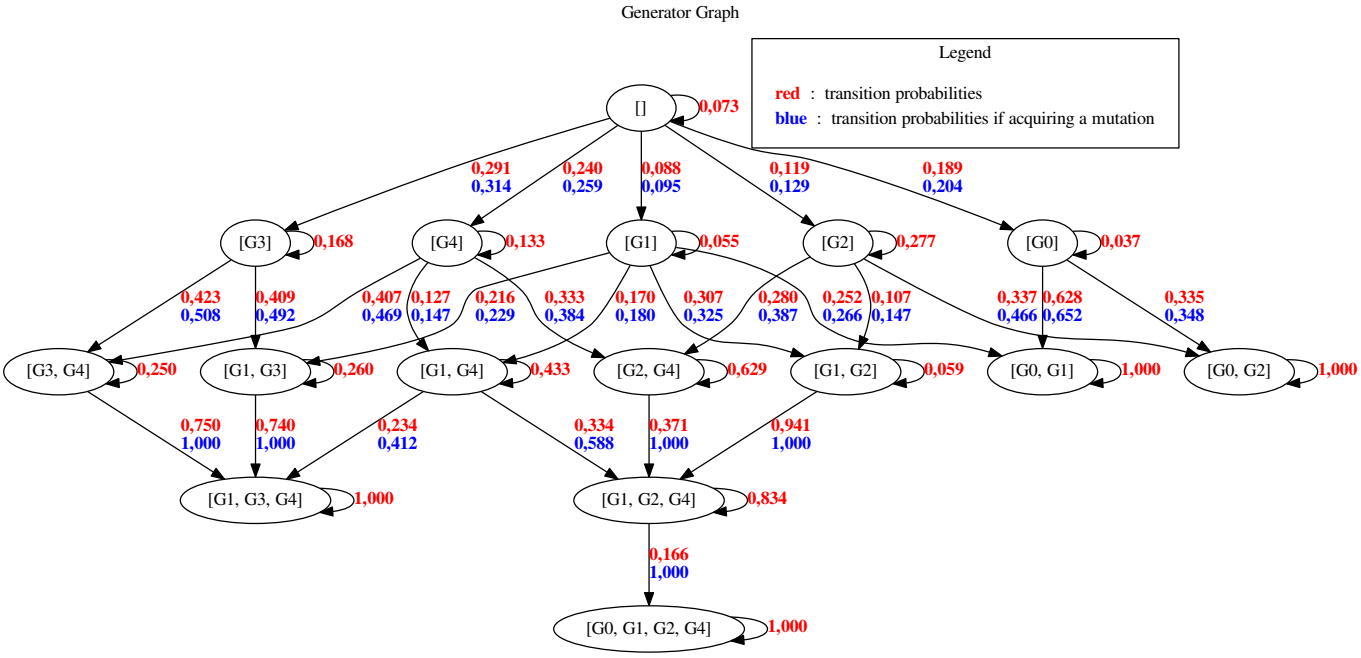

Figure 3.34: this generator is created randomly selecting nodes, it aims to represent a case where there is not a clear cancer progression

3.6.2 Our reconstruction

In this case we show how the reconstruction behaves when increasing the number of samples after fixing $k = 4$ for all graphs (figures from 3.35 to 3.38).

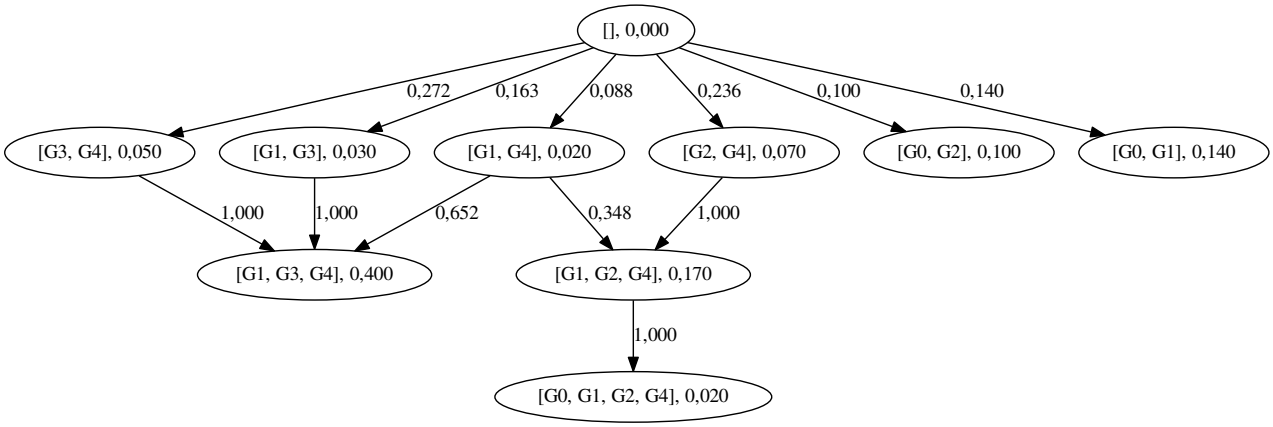

Figure 3.35: $n = 100$

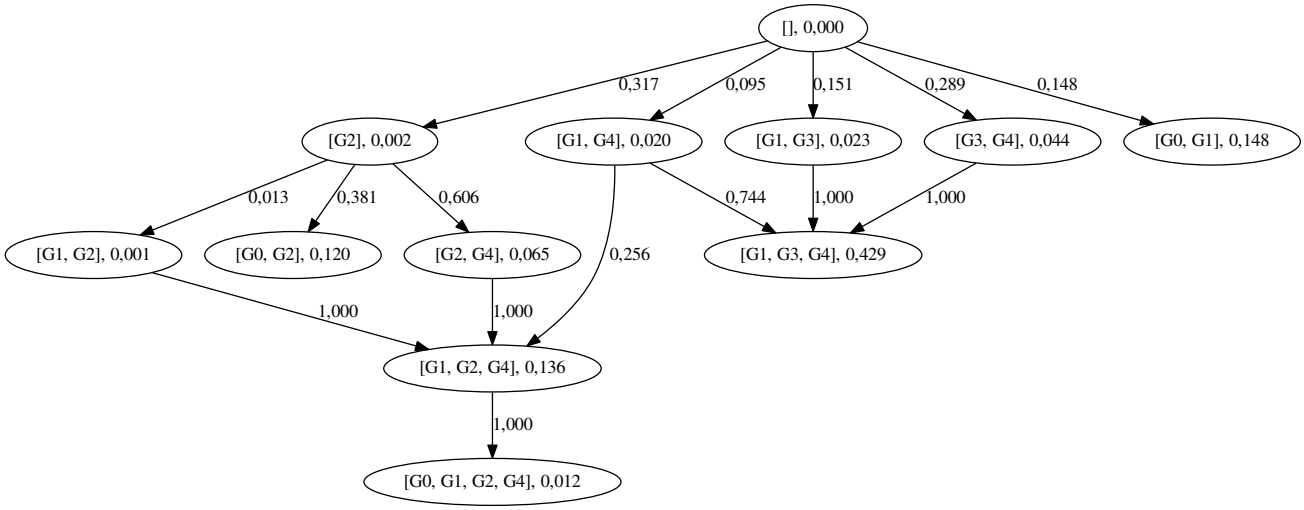

Figure 3.36: $n = 1000$

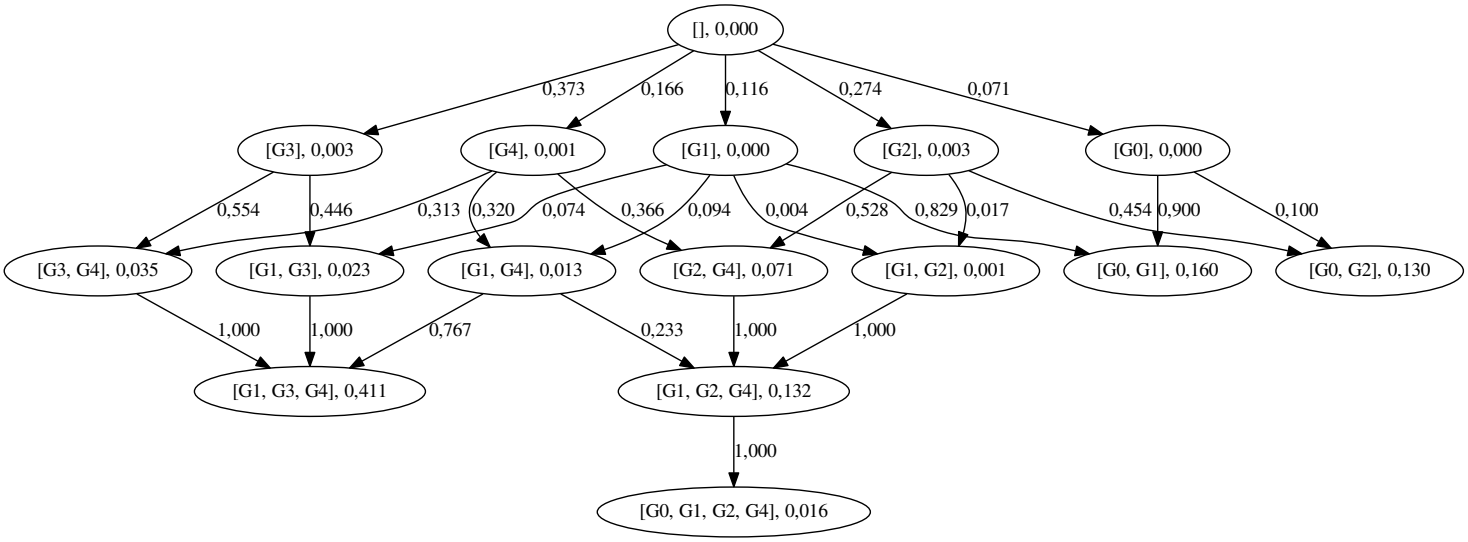

Figure 3.37: $n = 10000$

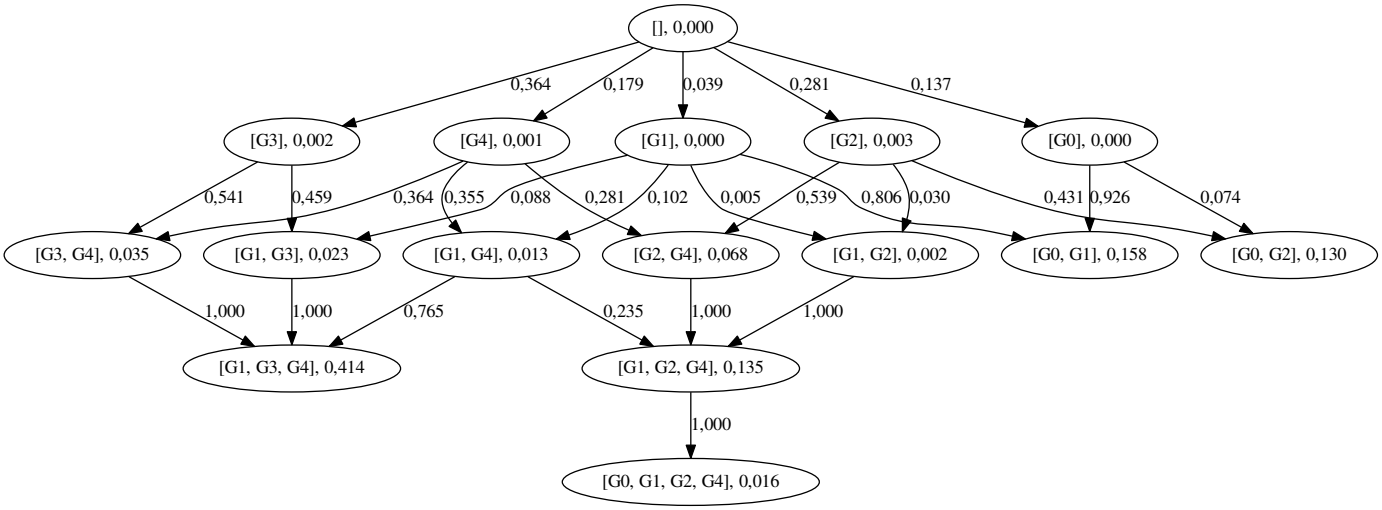

Figure 3.38: $n = 100000$

3.6.3 TRONCO suite

Each graph was obtained from a dataset generated with $n = 100000$ and $k = 4$ with the exception of the ChowLiu's one that had problems in the computation with this dataset (figure 3.41). Even through all the result from the TRONCO pipeline (figures from 3.39 to 3.43) are more or less consistent with each other but it seems that CAPRESE (figure 3.39) is the one that has more trouble inferring a solution in this context.

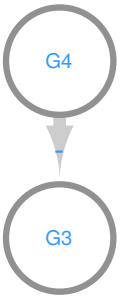

Figure 3.39: CAPRESE

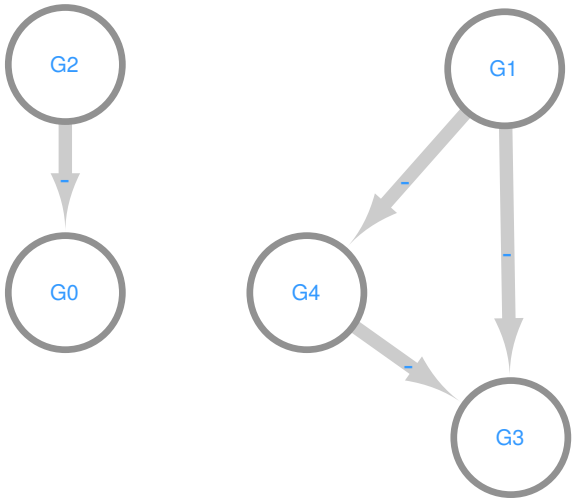

Figure 3.40: CAPRI

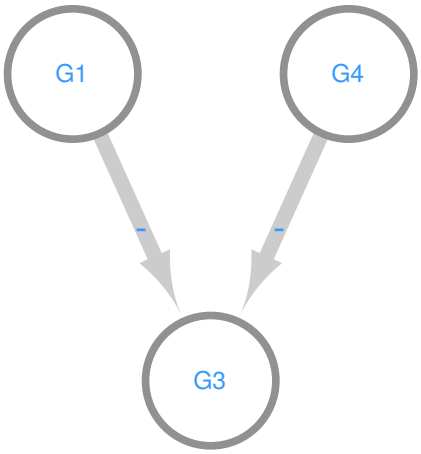

Figure 3.41: ChowLiu: this method failed almost every time with this generator, the dataset used has $n = 100$ and $k = 3$

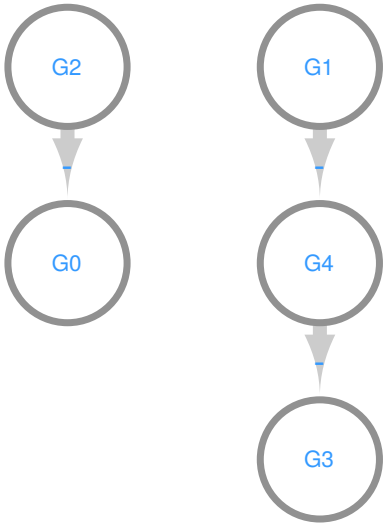

Figure 3.42: Gabow/Edmonds

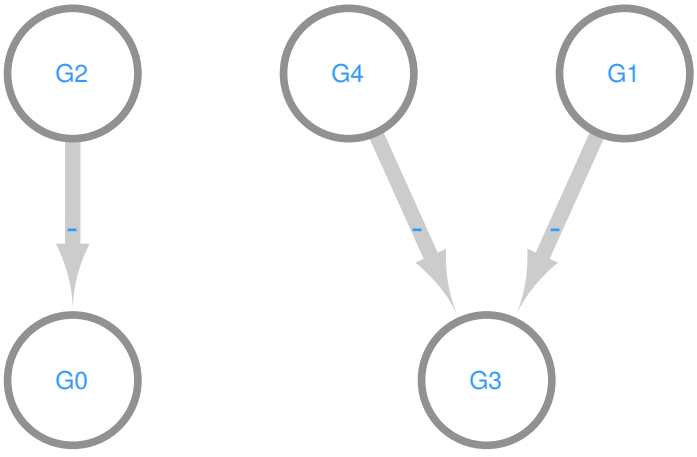

Figure 3.43: Prim

4

Test on Real Data

In this section we present a test of CIMICE on real data. We cannot extract any biological conclusion from these results because the pre-processing is not refined enough, but they should give an idea on how this tool can be used.

4.1 FROM CANCER SCSEQ DATA

Using the data of single cell sequencing (scSeq) of colorectal cancer coming from different tissues of three different patients [8], we ran our tool considering the most mutated genes obtaining the results seen in figure 4.1.

As it can be seen, there are very few cases in which the difference from a node and one of its successors is of a single gene and for 6 genotypes it is impossible to reconstruct an history at all. This is due to the really low number of samples we are analyzing, only 20 and also from different patients.

4.2 FROM BACTERICAL SCSEQ DATA

To cope with the lack of scSeq experiments on the same cancer tissue, we used data coming from different bacterial populations, as presented in this work [5]. This data comes from the analysis of STEC (Shiga Toxin producin Escherichia Coli) population from the patients of two hospitals in the Netherlands for a total of 120 samples. Hospitals are environments with an high selective pressure on bacteria for the high usage of drugs like antibiotics. By comparing the sequences of these bacteria with a reference for STEC we pointed out a coars number of mutated genes and we used this preliminary information to run our tool. Even with this uncleaned data our tool is capable of inferring some possible evolutionary trajectories among the samples, as seen in figure 4.2.

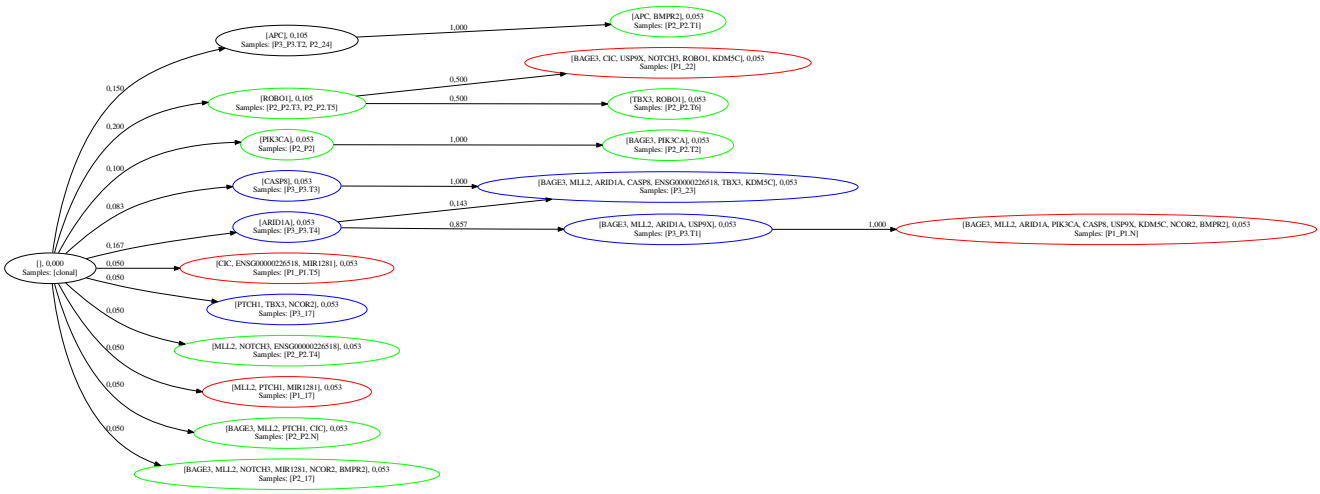

Figure 4.1: Result using [8] data, each color is associated to one patient, black nodes are composed genotypes seen in different patients

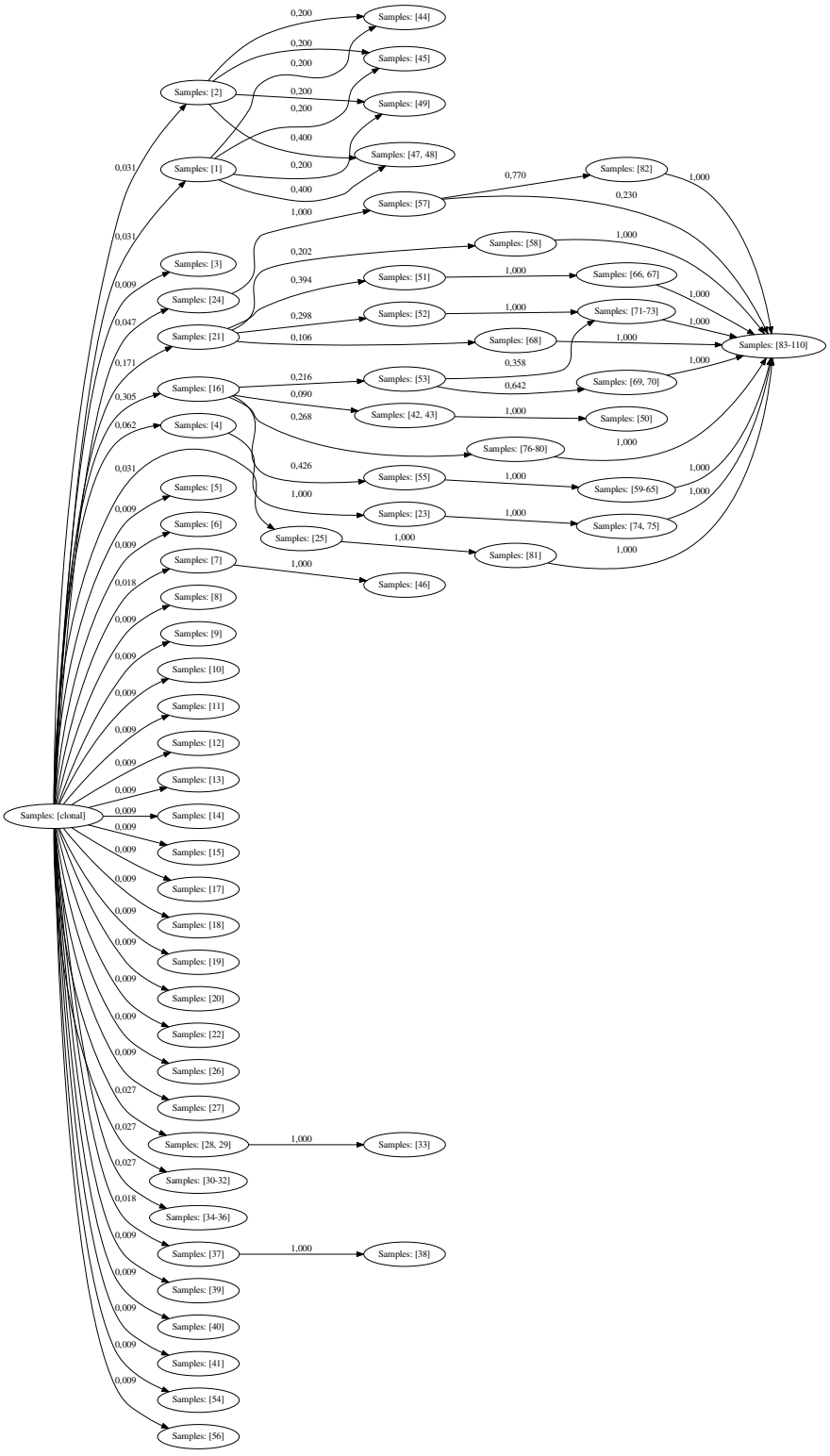

Figure 4.2: Result using [5] data, we omitted the complete genotypes for the samples because they were too long to be displayed.

Summary

In this work we presented CIMICE, a simple, efficient and novel method for cancer progression inference based on Markov chains. The method exploits few commonly used assumption. We proved its correctness with respect to our cancer model and we showed its performance on simulated and biological data through a Java implementation. We are planning to improve the analysis made on bacterial data as it seems a very promising approach to validate cancer inference methods biologically when the specific characteristics of the more complex features of the eukaryotic genomes are not taken into account. It would also be very interesting to enrich this method considering the peculiarities of different mutational types like CNVs and chromosome scale variations that could help in finding the temporal order of events. We could develop many different splitting functions for our tool and compare them to check their performances in order to understand how valid is our heuristic. It would also be quite important to find larger and higher resolution datasets that could help in improving our biological validation. There are many more ways to improve CIMICE, but this was exactly our goal, to create a simple core on which gradually grow a more advanced cancer inference method.

Bibliography

[1] Niko Beerenwinkel, Chris D Greenman, and Jens Lagergren. Computational cancer biology: an evolutionary perspective. *PLoS computational biology*, 12(2):e1004717, 2016.

[2] Luca De Sano, Giulio Caravagna, Daniele Ramazzotti, Alex Graudenzi, Giancarlo Mauri, Bud Mishra, and Marco Antoniotti. TRONCO: an R package for the inference of cancer progression models from heterogeneous genomic data. *Bioinformatics*, 32(12):1911–1913, 2016.

[3] WTO Department of Information, Evidenceand Research. WHO methods and data sources for country-level causes of death 2000-2016. March 2018.

[4] Richard Desper, Feng Jiang, Olli-P Kallioniemi, Holger Moch, Christos H Papadimitriou, and Alejandro A Schäffer. Inferring tree models for oncogenesis from comparative genome hybridization data. *Journal of computational biology*, 6(1):37–51, 1999.

[5] Mithila Ferdous, Alexander W Friedrich, Hajo Grundmann, Richard F de Boer, Peter D Croughs, Md Atiqul Islam, Marjolein FQ Kluytmans-van den Bergh, Anna MD Kooistra-Smid, and John WA Rossen. Molecular characterization and phylogeny of Shiga toxin–producing Escherichia coli isolates obtained from two Dutch regions using whole genome sequencing. *Clinical Microbiology and Infection*, 22(7):642–e1, 2016.

[6] Magali Olivier, Monica Hollstein, and Pierre Hainaut. TP53 mutations in human cancers: origins, consequences, and clinical use. *Cold Spring Harbor perspectives in biology*, 2(1):a001008, 2010.

[7] Marius Raica and Anca Maria Cimpean. Platelet-derived growth factor (PDGF)/PDGF receptors (PDGFR) axis as target for antitumor and antiangiogenic therapy. *Pharmaceuticals*, 3(3):572–599, 2010.

[8] Sophie F Roerink, Nobuo Sasaki, Henry Lee-Six, Matthew D Young, Ludmil B Alexandrov, Sam Behjati, Thomas J Mitchell, Sebastian Grossmann, Howard Lightfoot, David A Egan, et al. Intra-tumour diversification in colorectal cancer at the single-cell level. *Nature*, 556(7702):457, 2018.

[9] Russell Schwartz and Alejandro A Schäffer. The evolution of tumour phylogenetics: principles and practice. *Nature Reviews Genetics*, 18(4):213, 2017.