Counting Novikov worldlines on potential paternity graphs

Michal Horanský

December 1, 2025

1 Introduction

Note: $P_{a \to b}$ here denoted a simple directed path from vertex a to vertex b.

Definition 1 *A* ***potential paternity graph*** *G is a directed graph with vertices V such that*

$$\exists V_0 \subset V : (\forall a \in V, \exists b \in V_0 : \exists P_{b \to a} \quad \textit{and} \quad \forall c \in V_0, \nexists d \in (V - V_0) : \exists P_{d \to c})$$

Simply put, every vertex in G can be reached from V_0, and no arc points at V_0.

Note that G is not required to be connected or simple.

Definition 2 *A* ***worldline*** *is a simple directed path $P_{a \to b}$ on G such that*

1. *either $a \in V_0$*
2. *or there exists an arc $b \to a$, and the vertices of $P_{a \to b}$ therefore can form a directed cycle.*

In the first case, the worldline is said to be **rooted**; in the second case, it is said to be **looped**.

Definition 3 *A* ***realisation*** *of G is a subset of its vertices, $R \subset V$, such that R can be partitioned into worldlines.*

The empty set is called the ***trivial realisation*** *of G.*

Example: consider the potential paternity graph A formed by $N+1$ vertices $a_0 \dots a_N$ such that there is an arc $a_i \to a_{i+1}$ for $i = 0 \dots N-1$, and $V_0 = \{a_0\}$. Then, each worldline has the shape $\{a_0, a_1 \dots a_m\}$ for $m \in \{0 \dots N\}$. Therefore, there are $N+1$ possible worldlines in A, and therefore $N+2$ valid realisations (since no two worldlines are disjoint), also counting the trivial realisation.

Even though the partitioning of a realisation into worldlines may not be unique, the number of rooted worldlines is uniquely specified for a realisation (but not the number of looped worldlines: for an example, consider a complete directed graph with a looped worldline of more than one vertex: this worldline may be further partitioned into smaller looped worldlines). The number of rooted worldlines in a realisation is trivially equal to the number of vertices in the realisation which belong into V_0: this is called the **inclusion number** of the realisation.

2 Problems

The following are questions regarding potential paternity graphs which are the subjects of this inquiry.

2.1 Counting realisations

Given a potential paternity graph G, how many realisations of G are there?

2.2 Iterating through realisations

We wish to find a deterministic algorithm which iterates through the set of all realisations of G.

2.3 Iterating through realisations with preferentiality

Given a preference P which orders the set of realisations, we wish to find a deterministic algorithm which iterates through the set of all realisations of G (denoted $\mathcal{R}$) in order given by P. There are restrictions on the form of P.

Typically, P can be expressed as a sequence of preferences $(P_1, P_2 \dots P_\pi)$ such that each each P_i partially orders $\mathcal{R}$. Then, P applies to $\mathcal{R}$ like so: for two realisations $a, b \in \mathcal{R}$, if P_1 orders a, b, that order is applied; if not, P_2 is checked and so on. It other words, (P_i) defines a *lexicographical* ordering. Note that P must impose a total order.

The form of P_i can be chosen from the following categories:

- An ordering is chosen for the vertices V of G. For two realisations a, b, the first vertex according to the vertex ordering which is included in one but not both realisations is considered: the realisation which includes this vertex preceded the realisation which does not. (This is also a lexicographical order.)

- Same as above, but only ordering vertices in V_0 of G: inclusion of the first vertex in V_0 which is not included in both a and b will order them.
- For two realisations a, b, the realisation with a higher number of vertices precedes the other one.
- For two realisations a, b, the realisation with a higher inclusion number precedes the other one.

Note that the first of these always imposes a total order, and thus its inclusion in the preference sequence is sufficient for P to be a total order; the sequence also always terminates at that point, since no subsequent (lower priority) preference can affect P.

2.4 Realisation inclusion check

This is only an interesting question if the question in Sec. 2.3 proves to be too difficult to be answered. For a given subset of vertices $v \in V$ we wish to determine whether v is a realisation of the potential paternity graph. What is the fastest way to do this?