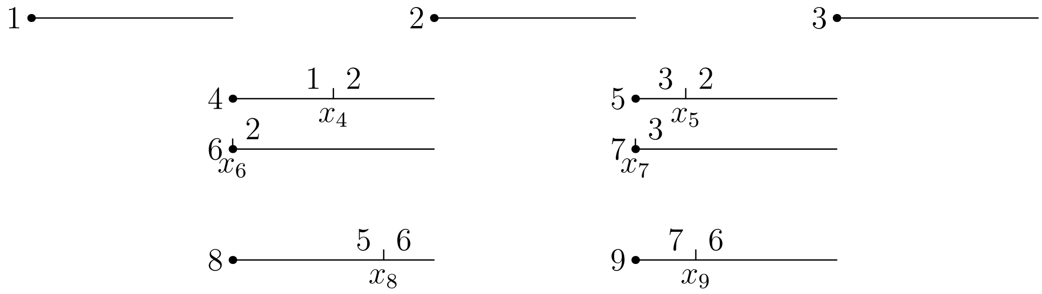
recording msprime ‘coalescence records’ from forward time simulations

Jaime Ashander

Goal Our goal is to encode the ARG produced from a simulation that proceeds forward in time in `msprime`-style coalesence records. One way to do this is to simulate forward, sample, and then work backwards. In theory, one could also produce ”partial” records going forward. Up to present, this is what we have done — producing records as described in *Going forward* below and augmenting any singleton remaining with a “phantom” sibling (all having the same ID. These can be loaded in to `msprime` but don’t actually meet the specs — there are many redundant trees encoded. It seems that to meet `msprime` format requirements we will still need to traverse backward from a given set of samples to prune off incomplete records.

A scenario Below each chromosome is labeled on the left with an ID; a single breakpoint, labelled below the chromosome, represents a recombination during the meiosis that produced the chromosome. Above, the haplotypes are labelled by the ID for the parent that produced them.

Consider a lineage descended from three parent chromosomes: $1, 2, 3$. Mating between these produced a first generation of two individuals $(4, 6)$ and $(5, 7)$. A subsequent mating between these produced gametes 8 and 9 in the second generation.



We describe coalescence records in the format $(left, right, node, (children), time)$ of `msprime`. Label the right endpoint of the chromosome by L and the left by 0.0. Further, label the current generation 0, the parent -1 and the grandparent -2. Below, I refer a pair (or more) combining backwards in time as "coalescence" and a haplotype finding itt parent but not coalescing with another as a "singleton".

Going forward

We could imagine constructing something like the records going forward in time. Each meiosis provides some information (not all of it ultimately useful).

In the grandparent generation, there is one singleton coalescence to 1:

```
A = (0, x_4, 1, (4, ), -2)
```

two different coalescence to 2 and a singleton left over:

```
B = (0, x_5, 2, (6, ), -2)
C = (x_5, x_4,  2, (5, 6), -2)
D = (x_4, L,  2, (4, 5, 6), -2)
```

and one coalescence to 3 with one singleton left:

```
E = (0, x_5, 3, (5, 7), -2)
F = (x_5, L, 3, (7, ), -2)
```

In the parent generation, there is one coalescence to 6:

```
G = (x_8, L,  6, (8, 9), -1)
```

and some singletons:

```
H = (0, x_8, 5, (8, ), -1)
I = (0, x_9, 7, (9, ), -1)
J = (x_9, x_8,  6, (9, ), -1)
```

Note that while going forward we cannot discard records that will not ultimately be used as we have no *a priori* criterion. For example, records relating to chromosome 4 are never used in sampling coalescence from 8 and 9 but this is not known going forward. Thus, whether or not we construct records in forward time along with the simulation we will need to need to use backward-time analysis to combine the records to a minimal set as `msprime` requires (see *Combining Records* below).

Going backward

Moving back in time, in the parent generation, a haplotype on the right coalesces in chromosome 6,

```
A' = (x_8, L, 6, (8, 9), -1)
```

The rest of the chromosome has yet to coalesce,

```
B' = (0, x_8, 5, (8, ), -1)
C' = (0, x_9, 7, (9, ), -1)
D' = (x_9, x_8, 6, (9, ), -1)
```

In the grandparent generation, the other parts coalesce

```
F' = (0, x_5, 3, (5, 7), -2)
G' = (x_9, x_8, 2, (5, 6), -2)
```

Combining Records

The singleton records in the parent generation (i.e., `H, I, J` in forward time or `B', C' D'` in backward time) can be followed up to the grandparent generation. Combined with the relevant coalescences in the grandparent generation (i.e., `C, D, F` in forward time or `F', G'` in backward time), these form complete records. For example, assume recombination points in chromosome 5 and 9 coincide and let $a = x_5 = x_9$. Then the complete records

```
(0, a, 3, (8, 9), -2)
(a, x_8, 2, (8, 9), -2)
```

Then, together with the one complete coalescence in the parent (`G` in forward time or `A'` in backward time)

```
(x_8, L, 6, (8, 9), -1)
```

these form a complete set of records.

Renumbering nodes and enumerating trees

Once the complete set of records is formed, to meet `msprime` requirements for a dense tree in the records, we renumber in ascending order from the leaves (the mapping from the chromosome numbers listed above is $8 \to 0; 9 \to 1; 6 \to 2; 3 \to 3; 2 \to 4$).

The complete set of records describes a dense tree sequence

```
(x_8, L, 2, (0, 1), -1)
(0, a, 3, (0, 1), -2)
(a, x_8, 4, (0, 1), -2)
```

and encodes the following sparse trees (moving left to right across the chromosome and encoding trees as integer vectors with `-1` signifying the root)

```
0, a: (3, 3, NA, -1, NA)
a, x_8: (4, 4, NA, NA, -1)
x_8, L: (2, 2, -1, NA, NA)
```