BIOINFORMATICS APPLICATIONS NOTE Vol. 30 no. 23 2014, pages 3427–3429
doi:10.1093/bioinformatics/btu562

Genetics and population analysis Advance Access publication August 22, 2014

Cosi2: an efficient simulator of exact and approximate coalescent with selection

Ilya Shlyakhter[1,2,*], Pardis C. Sabeti[1,2] and Stephen F. Schaffner[1]
[1]Broad Institute of MIT and Harvard, MA 02142 and [2]Department of Organismic and Evolutionary Biology, Harvard University, Cambridge, MA 02138, USA

Associate Editor: Jeffrey Barrett

ABSTRACT

Motivation: Efficient simulation of population genetic samples under a given demographic model is a prerequisite for many analyses. Coalescent theory provides an efficient framework for such simulations, but simulating longer regions and higher recombination rates remains challenging. Simulators based on a Markovian approximation to the coalescent scale well, but do not support simulation of selection. Gene conversion is not supported by any published coalescent simulators that support selection.

Results: We describe *cosi2*, an efficient simulator that supports both exact and approximate coalescent simulation with positive selection. *cosi2* improves on the speed of existing exact simulators, and permits further speedup in approximate mode while retaining support for selection. *cosi2* supports a wide range of demographic scenarios, including recombination hot spots, gene conversion, population size changes, population structure and migration.

cosi2 implements coalescent machinery efficiently by tracking only a small subset of the Ancestral Recombination Graph, sampling only relevant recombination events, and using augmented skip lists to represent tracked genetic segments. To preserve support for selection in approximate mode, the Markov approximation is implemented not by moving along the chromosome but by performing a standard backwards-in-time coalescent simulation while restricting coalescence to node pairs with overlapping or near-overlapping genetic material. We describe the algorithms used by *cosi2* and present comparisons with existing selection simulators.

Availability and implementation: A free C++ implementation of *cosi2* is available at http://broadinstitute.org/mpg/cosi2.

Contact: ilya@broadinstitute.org

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on May 11, 2014; revised on August 3, 2014; accepted on August 16, 2014

1 INTRODUCTION

A wide variety of population genetic analyses depend on efficient simulation of samples under a demographic model. The most efficient simulation method works backwards in time from the sample, first simulating the sample's genealogy according to coalescent theory and then placing mutations on that genealogy. Because of crossing over and gene conversion, the overall genealogy for a simulated region is not a tree but a directed acyclic graph whose size grows quickly with region length and with recombination rate. To address this simulation bottleneck, a Markovian approximation was proposed and implemented (Yang *et al*., 2014). However, the Markovian approximation does not readily allow for the modeling of positive selection. Simulators that do support positive selection (e.g. Ewing and Hermisson, 2010 and Teshima and Innan, 2009) suffer from the performance limitations of the traditional coalescent, and do not support all commonly needed features (variable genetic maps, gene conversion, structured populations) within a single framework.

Here, we describe the simulator *cosi2*, which implements the standard coalescent simulation algorithm but which does so using several optimizations that greatly reduce memory and CPU requirements. The use of the original coalescent allows *cosi2* to support simulation of positive selection using the standard structured coalescent approach. For additional efficiency, the Markovian approximation can be enabled while retaining support for simulating selection. *Cosi2* supports population structure, population size changes, bottlenecks and migrations. Recombination rate can be varied along the simulated region, and the program includes a utility to generate genetic maps with recombination hotspots. *Cosi2* also supports a simple model of gene conversion.

2 ALGORITHM DESCRIPTION

Here, we describe the different optimizations which, taken together, greatly improve the scaling behavior of coalescent simulations (Table 1). We also describe the implementation of positive selection modeling in *cosi2*. Additional details of the algorithms are given in the Supplemental Information.

2.1 Notation

An ancestral recombination graph (ARG) is a directed graph, where each node represents a chromosome contributing some genetic material to the present-day sample. Let G be an ARG and n be its node. $ch(n)$ is the set of n's children. $G_0 \equiv \{n \in G \mid ch(n) = \emptyset\}$. PL denotes the set of physical locations along the chromosome, represented as the unit interval $[0, 1]$. $inh(n_i, n_j)$ denotes the subset of PL (possibly empty) inherited by n_j from n_i. $L(n, s) \equiv \{n' \in G_0 \mid inh(n, n') \cap s \neq \emptyset\}$. $locs(n) \equiv \{x : PL \mid L(n, \{x\}) \neq \emptyset\}$. $n^b \equiv min(locs(n))$, $n^e \equiv max(locs(n))$. $coal(n_i, n_j, n_c)$ means n_c resulted from

*To whom correspondence should be addressed.

coalescence of n_i and n_j; $recomb(n_r, x, n_i, n_j)$ means recombining n_r at x produced nodes n_i and n_j.

2.2 Tracking only the veneer of the ARG

Coalescent simulators typically work in two phases. First, they simulate the genealogy of the sample, known as the ARG. Second, they simulate mutations on that geneology. *Cosi2* combines these into one phase, avoiding construction of the full ARG. During simulation, only a thin frontier of the ARG is kept, along with bookkeeping information sufficient for placing mutations.

Standard coalescent simulations begin by initializing the ARG with the present-day population sample G_0. A sequence of coalescences and recombinations is then sampled, updating G: for a coalescence of nodes n_i and n_j producing n_c, $G \leftarrow G \cup \{n_c\}$; for a recombination of node n_r into nodes n_i and n_j, $G \leftarrow G \cup \{n_i, n_j\}$. Then, mutations are placed on the ARG. For mutation placed above node n at chromosomal location $x \in locs(n)$, $L(n, \{x\})$ is found by tracing down the ARG: $L(n, \{x\}) = G_0 \cap trace(n, x)$ where $trace(n, x) = \{n\} \cup (\cup\{trace(c, x) \mid c \in ch(n) \wedge x \in locs(c)\})$.

In *cosi2*, ARG nodes are discarded as soon as a parent is generated. For coalescence, $G \leftarrow G \cup \{n_c\} \setminus \{n_i, n_j\}$; for recombination, $G \leftarrow G \cup \{n_i, n_j\} \setminus n_r$. Not keeping the full ARG removes a computational bottleneck, but discards information needed to compute $L(n, \{x\})$ when placing mutations. To compensate, we store for each node n the full function $L_n(x) = L(n, \{x\})$, represented efficiently using skip lists (see below). L_n is updated incrementally during simulation: $coal(n_i, n_j, n_c) \Longrightarrow L_{n_c} = L_{n_i} \cup L_{n_j}$; $recomb\ (n_r, x, n_i, n_j) \Longrightarrow L_{n_i} = L_{n_r} \upharpoonright [n_i^b, x]$. Despite not being kept in memory, the full ARG can be logged to a file using an option described in the documentation.

2.3 Efficiently representing segment lists

The piecewise constant function L_n is represented efficiently at each node as a skip list (Pugh, 1990) of segments on which it is constant and non-empty. The skip pointers of the skip list are augmented with the total physical and genetic length of the skipped segments. Standard skip-list techniques then permit logarithmic-time updates of L_n during recombination, coalescence and gene conversion. Keeping just the veneer of the ARG works synergystically with skip lists: as nodes participating in recombination or coalescence are discarded immediately after the operation, large portions of their skip lists can be reused in constructing the skip-list representation of the result.

2.4 Efficient sampling of crossovers and gene conversions

The vast majority of crossover and gene conversion events do not change the ARG but do take time to track. These are either crossovers falling entirely to one side of a node's segment list or gene conversions falling entirely outside any segment. *Cosi2* directly samples only crossovers and gene conversions that actually change the ARG, by keeping the crossover and gene conversion rates of the individual nodes in a Fenwick tree data structure supporting quick access to the total rate and efficient sampling of event location weighted by the genetic map. The skip-list representation of segment lists allows incremental updating of the node's individual event rates.

Table 1. Simulation time comparison

Model parameters				Mean simulation time per replica (s)			
L	*k*	*r*	*f*	*msms*	*mbs*	*cosi2*	*cosi2* ^{apx}
3	40	1e−8	0.57	3	76	0.7	0.25
10	80	1e−8	0.57	40	$> 1e3$	5	1
30	200	2e−8	0.70	2786	$> 1e4$	162	21
60	200	2e−8	0.70	10286	$> 2e4$	686	71
3	40	$1e-8^{hs}$	0.57	n/s	69	0.76	0.26
10	80	$1e-8^{gc}$	0.57	n/s	n/s	9	3
10	40	1e−8	0.35	16.02	$> 1e3$	2.44	0.59
20	200	1e−8	0.80	431.6	$> 2e4$	25.7	4.65
10	40	1e−8	1.0	28.06	$> 1e3$	4.51	0.90

Note. Model parameters: *L*—region length in megabases, *k*—sample size, *r*—recombination rate (^{hs}—with hotspots, ^{gc}—with gene conversion), *f*—present-day frequency of selected allele. Mutation rate $\mu = 1e-8$. Selection coefficient: rows 1-6, $s = 0.0185$; row 7, $s = 0.02$; row 8, $s = 0.0185$; row 9, $s = 0.043$. n/s means not supported, *cosi2* ^{apx} is *cosi2* with Markov approximation with $u = 0.001$.

2.5 Modeling of selection

Selection is implemented using the structured coalescent approach (Teshima and Innan, 2009). At start of simulation, sampled chromosomes are partitioned into two classes based on their allelic state at the selection site. Coalescence happens only within the same allelic class, with coalescence rate based on the frequency trajectory of the causal allele. The frequency trajectory of the selected allele can be deterministic (based on causal allele age and present-day frequency), or can be provided externally; the latter mode permits the use of stochastic trajectories generated for any selection model.

2.6 Implementing the Markovian approximation

We implement an approximation to the coalescent, in which coalescences are restricted to occur between nodes whose genetic information overlaps or nearly overlaps. This approximation was proved equivalent to the Markovian simulation of coalescent along the chromosome (McVean and Cardin, 2005). By modifying only the coalescence step within an existing coalescent simulator that supports selection, we add the ability to approximate the coalescent while preserving support for simulating selection. The approximation speeds up computation by reducing the number of coalescence and recombination events generated during simulation. For two nodes p, q where $p^e \ll q^b$, repeated cycles of $coal(p, q, r)$ followed by $recomb(r, x, p, q)$ where $p^e \leq x \leq q^b$ are avoided. Such cycles do not change the output when no coalescences involving r happen between $coal(p, q, r)$ and $recomb(r, x, p, q)$. In general, ignoring such cycles yields an approximation, the quality of which is checked empirically.

Sampling restricted coalescences requires knowing, at each simulation step, the number of coalesceable node pairs. We maintain this information incrementally using a dynamic data structure based on augmented interval trees. We define a *hull* of a node n as $H(n) = [n^b, n^e + u]$ where u is a parameter controlling the amount of approximation (smaller u means greater approximation). We declare two nodes coalesceable if their hulls overlap; u therefore specifies the maximum separation between coalescing nodes, as a fraction of the total length of the simulated region. We can dynamically maintain the count of coalesceable node pairs. Initially, the count equals $\binom{|G_0|}{2}$. Adding a hull $[n^b, n^e]$ adds $|G| - |\{n' \mid n'^e < n^b\}| - |\{n' \mid n'^b > n^e\}|$ to the count of hull intersections (Layer *et al.*, 2013). Hull removal is analogous. Because all coalescent operations (crossover, gene conversion, coalescence and migration) can be implemented as a sequence of hull additions and removals, we can maintain the count of coalesceable node pairs with only logarithmic overhead. Selecting a coalesceable pair uniformly at random requires maintenance of additional information; description of the implementation—also requiring only logarithmic time and space overhead—is given in Supplemental Information.

3 SUMMARY

cosi2 provides a combination of performance and supported demographic scenarios unavailable with existing selection-supporting simulators. We hope it becomes a useful tool for population geneticists studying positive selection.

Funding: PCS is funded by an NIH Innovator Award 1DP2OD006514-01 and by a Broad Institute SPARC award.

Conflict of interest: none declared.

REFERENCES

Ewing,G. and Hermisson,J. (2010) MSMS: a coalescent simulation program including recombination, demographic structure and selection at a single locus. *Bioinformatics*, **26**, 2064–2065.

Layer,R.M. *et al.* (2013) Binary interval search: a scalable algorithm for counting interval intersections. *Bioinformatics*, **29**, 1–7.

McVean,G.A. and Cardin,N.J. (2005) Approximating the coalescent with recombination. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, **360**, 1387–1393.

Pugh,W. (1990) Skip lists: a probabilistic alternative to balanced trees. *Commun. ACM*, **33**, 668–676.

Teshima,K. and Innan,H. (2009) mbs: modifying Hudson's ms software to generate samples of DNA sequences with a biallelic site under selection. *BMC Bioinformatics*, **10**, 166.

Yang,T. *et al.* (2014) Critical assessment of coalescent simulators in modeling recombination hotspots in genomic sequences. *BMC Bioinformatics*, **15**, 3.