Faster Monte-Carlo Algorithms for Fixation Probability of the Moran Process on Undirected Graphs[*]

Krishnendu Chatterjee[1], Rasmus Ibsen-Jensen[2], and Martin A. Nowak[3]

1 **IST Austria, Klosterneuburg, Austria**
`krish@ist.ac.at`

2 **IST Austria, Klosterneuburg, Austria**
`ribsen@ist.ac.at`

3 **Program for Evolutionary Dynamics, Harvard University, Cambridge, USA**
`martin_nowak@harvard.edu`

Abstract

Evolutionary graph theory studies the evolutionary dynamics in a population structure given as a connected graph. Each node of the graph represents an individual of the population, and edges determine how offspring are placed. We consider the classical birth-death Moran process where there are two types of individuals, namely, the residents with fitness 1 and mutants with fitness r. The fitness indicates the reproductive strength. The evolutionary dynamics happens as follows: in the initial step, in a population of all resident individuals a mutant is introduced, and then at each step, an individual is chosen proportional to the fitness of its type to reproduce, and the offspring replaces a neighbor uniformly at random. The process stops when all individuals are either residents or mutants. The probability that all individuals in the end are mutants is called the fixation probability, which is a key factor in the rate of evolution. We consider the problem of approximating the fixation probability.

The class of algorithms that is extremely relevant for approximation of the fixation probabilities is the Monte-Carlo simulation of the process. Previous results present a polynomial-time Monte-Carlo algorithm for undirected graphs when r is given in unary. First, we present a simple modification: instead of simulating each step, we discard *ineffective* steps, where no node changes type (i.e., either residents replace residents, or mutants replace mutants). Using the above simple modification and our result that the number of effective steps is concentrated around the expected number of effective steps, we present faster polynomial-time Monte-Carlo algorithms for undirected graphs. Our algorithms are always at least a factor $O(n^2/\log n)$ faster as compared to the previous algorithms, where n is the number of nodes, and is polynomial even if r is given in binary. We also present lower bounds showing that the upper bound on the expected number of effective steps we present is asymptotically tight for undirected graphs.

1998 ACM Subject Classification J.1.1 Biology and genetics, E.1.3 Graphs and networks

Keywords and phrases Graph algorithms, Evolutionary biology, Monte-Carlo algorithms

Digital Object Identifier 10.4230/LIPIcs.MFCS.2017.61

[*] Some proofs are missing. See the full version [2]

1 Introduction

In this work we present faster Monte-Carlo algorithms for approximation of the fixation probability of the fundamental Moran process on population structures with symmetric interactions. We start with the description of the problem.

Evolutionary dynamics. Evolutionary dynamics act on populations, where the composition of the population changes over time due to mutation and selection. Mutation generates new types and selection changes the relative abundance of different types. A fundamental concept in evolutionary dynamics is the fixation probability of a new mutant [7, 11, 13, 14]: Consider a population of n *resident* individuals, each with a fitness value 1. A single *mutant* with non-negative fitness value r is introduced in the population as the initialization step. Intuitively, the fitness represents the reproductive strength. In the classical Moran process the following *birth-death* stochastic steps are repeated: At each time step, one individual is chosen at random proportional to the fitness to reproduce and one other individual is chosen uniformly at random for death. The offspring of the reproduced individual replaces the dead individual. This stochastic process continues until either all individuals are mutants or all individuals are residents. The *fixation probability* is the probability that the mutants take over the population, which means all individuals are mutants. A standard calculation shows that the fixation probability is given by $(1-(1/r))/(1-(1/r^n))$. The correlation between the relative fitness r of the mutant and the fixation probability is a measure of the effect of natural selection. The rate of evolution, which is the rate at which subsequent mutations accumulate in the population, is proportional to the fixation probability, the mutation rate, and the population size n. Hence fixation probability is a fundamental concept in evolution.

Evolutionary graph theory. While the basic Moran process happens in well-mixed population (all individuals interact uniformly with all others), a fundamental extension is to study the process on population structures. Evolutionary graph theory studies this phenomenon. The individuals of the population occupy the nodes of a connected graph. The links (edges) determine who interacts with whom. Basically, in the birth-death step, for the death for replacement, a neighbor of the reproducing individual is chosen uniformly at random. Evolutionary graph theory describes evolutionary dynamics in spatially structured population where most interactions and competitions occur mainly among neighbors in physical space [12, 3, 8, 15]. Undirected graphs represent population structures where the interactions are symmetric, whereas directed graphs allow for asymmetric interactions. The fixation probability depends on the population structure [12, 1, 9, 4]. Thus, the fundamental computational problem in evolutionary graph theory is as follows: given a population structure (i.e., a graph), the relative fitness r, and $\epsilon > 0$, compute an ϵ-approximation of the fixation probability.

Monte-Carlo algorithms. A particularly important class of algorithms for biologists is the Monte-Carlo algorithms, because it is simple and easy to interpret. The Monte-Carlo algorithm for the Moran process basically requires to simulate the process, and from the statistics obtain an approximation of the fixation probability. Hence, the basic question we address in this work is simple Monte-Carlo algorithms for approximating the fixation probability. It was shown in [6] that simple simulation can take exponential time on directed graphs and thus we focus on undirected graphs. The main previous algorithmic result in this area [5] presents a polynomial-time Monte-Carlo algorithm for undirected graphs when r is given in unary. The main result of [5] shows that for undirected graphs it suffices to run each simulation for polynomially many steps.

Table 1 Comparison with previous work, for constant $r > 1$. We denote by n, Δ, τ, and ϵ, the number of nodes, the maximum degree, the random variable for the fixation time, and the approximation factor, respectively. The results in the column "All steps" is from [5], except that we present the dependency on Δ, which was considered as n in [5]. The results of the column "Effective steps" is the results of this paper

	All steps	Effective steps
#steps in expectation	$O(n^2\Delta^2)$	$O(n\Delta)$
Concentration bounds	$\Pr[\tau \geq \frac{n^2\Delta^2 rx}{r-1}] \leq 1/x$	$\Pr[\tau \geq \frac{6n\Delta x}{\min(r-1,1)}] \leq 2^{-x}$
Sampling a step	$O(1)$	$O(\Delta)$
Fixation algo	$O(n^6\Delta^2\epsilon^{-4})$	$O(n^2\Delta^2\epsilon^{-2}(\log n + \log \epsilon^{-1}))$

Our contributions. In this work our main contributions are as follows:

1. *Faster algorithm for undirected graphs* First, we present a simple modification: instead of simulating each step, we discard *ineffective* steps, where no node changes type (i.e., either residents replace residents, or mutants replace mutants). We then show that the number of effective steps is concentrated around the expected number of effective steps. The sampling of each effective step is more complicated though than sampling of each step. We then present an efficient algorithm for sampling of the effective steps, which requires $O(m)$ preprocessing and then $O(\Delta)$ time for sampling, where m is the number of edges and Δ is the maximum degree. Combining all our results we obtain faster polynomial-time Monte-Carlo algorithms: Our algorithms are always at least a factor $n^2/\log n$ times a constant (in most cases $n^3/\log n$ times a constant) faster as compared to the previous algorithm, and is polynomial even if r is given in binary. We present a comparison in Table 1, for constant $r > 1$ (since the previous algorithm is not in polynomial time for r in binary). For a detailed comparison see the full version [2].
2. *Lower bounds* We also present lower bounds showing that the upper bound on the expected number of effective steps we present is asymptotically tight for undirected graphs.

Related complexity result. While in this work we consider evolutionary graph theory, a related problem is evolutionary games on graphs (which studies the problem of frequency dependent selection). The approximation problem for evolutionary games on graphs is considerably harder (e.g., PSPACE-completeness results have been established) [10].

Technical contributions. Note that for the problem we consider the goal is not to design complicated efficient algorithms, but simple algorithms that are efficient. By simple, we mean something that is related to the process itself, as biologists understand and interpret the Moran process well. Our main technical contribution is a simple idea to discard ineffective steps, which is intuitive, and we show that the simple modification leads to significantly faster algorithms. We show a gain of factor $O(n\Delta)$ due to the effective steps, then lose a factor of $O(\Delta)$ due to sampling, and our other improvements are due to better concentration results. We also present an interesting family of graphs for the lower bound examples. Technical proofs omitted due to lack of space are in the full version [2].

2 Moran process on graphs

Connected graph and type function. We consider the population structure represented as a connected graph. There is a connected *graph* $G = (V, E)$, of n nodes and m edges, and two types $T = \{t_1, t_2\}$. The two types represent residents and mutants, and in the technical exposition we refer to them as t_1 and t_2 for elegant notation. We say that a node v is a *successor* of a node u if $(u, v) \in E$. The graph is *undirected* if for all $(u, v) \in E$ we also have $(v, u) \in E$, otherwise it is *directed*. There is a *type function* f mapping each node v to a type $t \in T$. Each type t is in turn associated with a positive integer $w(t)$, the type's fitness denoting the corresponding reproductive strength. Without loss of generality, we will assume that $r = w(t_1) \geq w(t_2) = 1$, for some number r (the process we consider does not change under scaling, and r denotes relative fitness). Let $W(f) = \sum_{u \in V} w(f(u))$ be the total fitness. For a node v let $\deg v$ be the degree of v in G. Also, let $\Delta = \max_{v \in V} \deg v$ be the maximum degree of a node. For a type t and type function f, let $V_{t,f}$ be the nodes mapped to t by f. Given a type t and a node v, let $f[v \to t]$ denote the following function: $f[v \to t](u) = t$ if $u = v$ and $f(u)$ otherwise.

Moran process on graphs. We consider the following classical Moran birth-death process where a *dynamic evolution step* of the process changes a type function from f to f' as follows:

1. First a node v is picked at random with probability proportional to $w(f(v))$, i.e. each node v has probability of being picked equal to $\frac{w(f(v))}{W(f)}$.
2. Next, a successor u of v is picked uniformly at random.
3. The type of u is then changed to $f(v)$. In other words, $f' = f[u \to f(v)]$.

Fixation. A type t *fixates* in a type function f if f maps all nodes to t. Given a type function f, repeated applications of the dynamic evolution step generate a sequence of type functions $f = f_1, f_2, \dots, f_\infty$. Note that if a type has fixated (for some type t) in f_i then it has also fixated in f_j for $i < j$. We say that a process has *fixation time* i if f_i has fixated but f_{i-1} has not. We say that an initial type function f has fixation probability p for a type t, if the probability that t eventually fixates (over the probability measure on sequences generated by repeated applications of the dynamic evolution step f)

Basic questions. We consider the following basic questions:

1. *Fixation problem* Given a type t, what is the fixation probability of t averaged over the n initial type functions with a single node mapping to t?
2. *Extinction problem* Given a type t, what is the fixation probability of t averaged over the n initial type functions with a single node *not* mapping to t?
3. *Generalized fixation problem* Given a graph, a type t and an type function f what is the fixation probability of t in G, when the initial type function is f?

▶ Remark. Note that in the *neutral* case when $r = 1$, the fixation problem has answer $1/n$ and extinction problem has answer $1 - 1/n$. Hence, in the rest of the paper we will consider $r > 1$. Also, to keep the presentation focused, in the main article, we will consider fixation and extinction of type t_1. In the full version [2] we also present another algorithm for the extinction of t_2.

Results. We will focus on undirected graphs. For undirected graphs, we will give new FPRAS (fully polynomial, randomized approximation scheme) for the fixation and the extinction problem, and a polynomial-time algorithm for an additive approximation of the generalized fixation problem. There exists previous FPRAS for the fixation and extinction problems [5]. Our upper bounds are at least a factor of $O(\frac{n^2}{\log n})$ (most cases $O(\frac{n^3}{\log n})$) better and always in Poly$(n, 1/\epsilon)$, whereas the previous algorithms are not in polynomial time for r given in binary.

3 Discarding ineffective steps

We consider undirected graphs. Previous work by Diaz et al. [5] showed that the expected number of dynamic evolution steps till fixation is polynomial, and then used it to give a polynomial-time Monte-Carlo algorithm. Our goal is to improve the quite high polynomial-time complexity, while giving a Monte-Carlo algorithm. To achieve this we define the notion of effective steps.

Effective steps. A dynamic evolution step, which changes the type function from f to f', is *effective* if $f \neq f'$ (and *ineffective* otherwise). The idea is that steps in which no node changes type (because the two nodes selected in the dynamic evolution step already had the same type) can be discarded, without changing which type fixates/gets eliminated.

Two challenges. The two challenges are as follows:

1. *Number of steps* The first challenge is to establish that the expected number of effective steps is asymptotically smaller than the expected number of all steps. We will establish a factor $O(n\Delta)$ improvement (recall Δ is the maximum degree).
2. *Sampling* Sampling an effective step is harder than sampling a normal step. Thus it is not clear that considering effective steps leads to a faster algorithm. We consider the problem of efficiently sampling an effective step in a later section, see Section 5. We show that sampling an effective step can be done in $O(\Delta)$ time (after $O(m)$ preprocessing).

Notation. For a type function f, let $\Gamma_v(f)$ be the subset of successors of v, such that $u \in \Gamma_v(f)$ iff $f(v) \neq f(u)$. Also, let $W'(f) = \sum_u w(f(u)) \cdot \frac{|\Gamma_u(f)|}{\deg u}$.

Modified dynamic evolution step. Formally, we consider the following *modified dynamic evolution step* (that changes the type function from f to f' and assumes that f does not map all nodes to the same type):

1. First a node v is picked at random with probability proportional to $p(v) = w(f(v)) \cdot \frac{|\Gamma_v(f)|}{\deg v}$ i.e. each node v has probability of being picked equal to $\frac{p(v)}{W'(f)}$.
2. Next, a successor u of v is picked uniformly at random among $\Gamma_v(f)$.
3. The type of u is then changed to $f(v)$, i.e., $f' = f[u \to f(v)]$.

In the following lemma we show that the modified dynamic evolution step corresponds to the dynamic evolution step except for discarding steps in which no change was made.

▶ **Lemma 1.** *Fix any type function f such that neither type has fixated. Let f_d (resp., f_m) be the next type function under dynamic evolution step (resp., modified dynamic evolution step). Then,* $\Pr[f \neq f_d] > 0$ *and for all type functions f' we have:* $\Pr[f' = f_d \mid f \neq f_d] = \Pr[f' = f_m]$.

Potential function ψ. Similar to [5] we consider the *potential function* $\psi = \sum_{v \in V_{t_1,f}} \frac{1}{\deg v}$ (recall that $V_{t_1,f}$ is the set of nodes of type t_1). We now lower bound the expected difference in potential per modified evolutionary step.

▶ **Lemma 2.** *Let f be a type function such that neither type has fixated. Apply a modified dynamic evolution step on f to obtain f'. Then,*

$$\mathbb{E}[\psi(f') - \psi(f)] \geq \frac{r-1}{\Delta \cdot (r+1)} \quad .$$

Proof. Observe that f differs from f' for exactly one node u. More precisely, let v be the node picked in line 1 of the modified dynamic evolution step and let u be the node picked in line 2. Then, $f' = f[u \to f(v)]$. The probability to select v is $\frac{p(v)}{W'(f)}$. The probability to then pick u is $\frac{1}{|\Gamma_v(f)|}$.

We have that

- If $f(u) = t_2$ (and thus, since it got picked $f(v) = t_1$), then $\psi(f') - \psi(f) = \frac{1}{\deg u}$.
- If $f(u) = t_1$ (and thus, since it got picked $f(v) = t_2$), then $\psi(f') - \psi(f) = -\frac{1}{\deg u}$.

Below we use the following notations:

$$E_{12} = \{(v,u) \in E \mid f(v) = t_1 \text{ and } f(u) = t_2\}; E_{21} = \{(v,u) \in E \mid f(v) = t_2 \text{ and } f(u) = t_1\}.$$

Thus,

$$\begin{aligned}
&\mathbb{E}[\psi(f') - \psi(f)] = \\
&\sum_{(v,u)\in E_{12}} \left(\frac{p(v)}{W'(f)} \cdot \frac{1}{|\Gamma_v(f)|} \cdot \frac{1}{\deg u}\right) - \sum_{(v,u)\in E_{21}} \left(\frac{p(v)}{W'(f)} \cdot \frac{1}{|\Gamma_v(f)|} \cdot \frac{1}{\deg u}\right) \\
&= \sum_{(v,u)\in E_{12}} \left(\frac{w(f(v))}{W'(f) \cdot (\deg u) \cdot (\deg v)}\right) - \sum_{(v,u)\in E_{21}} \left(\frac{w(f(v))}{W'(f) \cdot (\deg u) \cdot (\deg v)}\right) .
\end{aligned}$$

Using that the graph is undirected we get,

$$\begin{aligned}
\mathbb{E}[\psi(f) - \psi(f')] &= \sum_{(v,u)\in E_{12}} \left(\frac{w(f(v)) - w(f(u))}{W'(f) \cdot (\deg u) \cdot (\deg v)}\right) \\
&= \frac{1}{W'(f)} \sum_{(v,u)\in E_{12}} \left(\frac{r-1}{\min(\deg u, \deg v) \cdot \max(\deg u, \deg v)}\right) \\
&\geq \frac{r-1}{\Delta \cdot W'(f)} \sum_{(v,u)\in E_{12}} \frac{1}{\min(\deg u, \deg v)} = \frac{r-1}{\Delta \cdot W'(f)} \cdot S \ ,
\end{aligned}$$

where $S = \sum_{(v,u)\in E_{12}} \frac{1}{\min(\deg u, \deg v)}$. Note that in the second equality we use that for two numbers a, b, their product is equal to $\min(a,b) \cdot \max(a,b)$. By definition of $W'(f)$, we have

$$\begin{aligned}
W'(f) &= \sum_u w(f(u)) \cdot \frac{|\Gamma_u(f)|}{\deg u} = \sum_u \sum_{v \in \Gamma_u(f)} \frac{w(f(u))}{\deg u} = \sum_{\substack{(v,u)\in E \\ f(u) \neq f(v)}} \frac{w(f(u))}{\deg u} \\
&= \sum_{(v,u)\in E_{12}} \left(\frac{w(f(u))}{\deg u} + \frac{w(f(v))}{\deg v}\right) \leq \sum_{(v,u)\in E_{12}} \frac{w(f(u)) + w(f(v))}{\min(\deg u, \deg v)} = (r+1) \cdot S \ .
\end{aligned}$$

Thus, we see that $\mathbb{E}[\psi(f') - \psi(f)] \geq \frac{r-1}{\Delta \cdot (r+1)}$, as desired. This completes the proof. ◀

▶ **Lemma 3.** *Let $r = x\Delta$ for some number $x > 0$. Let f be a type function such that neither type has fixated. Apply a modified dynamic evolution step on f to obtain f'. The probability that $|V_{t_1,f'}| = |V_{t_1,f}| + 1$ is at least $\frac{x}{x+1}$ (otherwise, $|V_{t_1,f'}| = |V_{t_1,f}| - 1$).*

▶ **Lemma 4.** *Consider an upper bound ℓ, for each starting type function, on the expected number of (effective) steps to fixation. Then for any starting type function the probability that fixation requires more than $2 \cdot \ell \cdot x$ (effective) steps is at most 2^{-x}.*

We now present the main theorem of this section, which we obtain using the above lemmas, and techniques from [5].

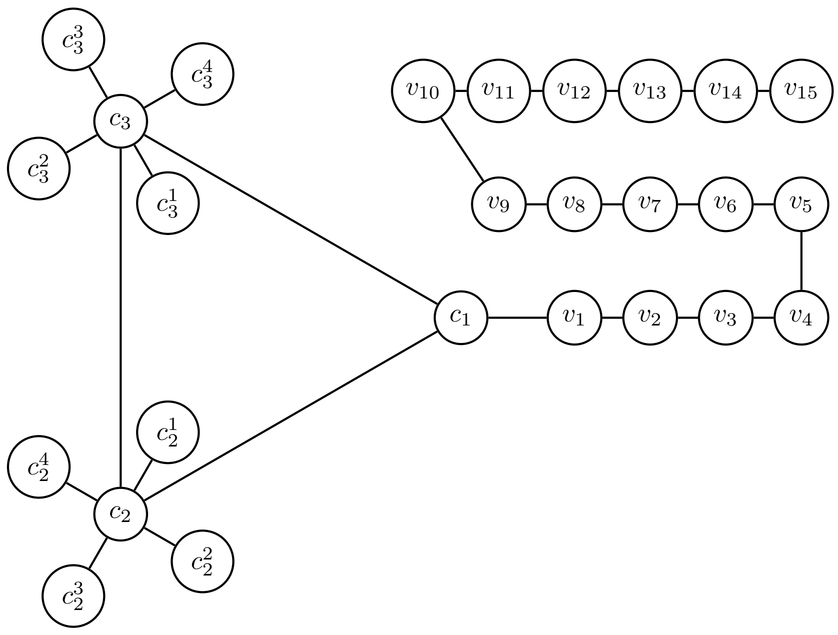

Figure 1 Example of a member of the family that attains the lower bound for undirected graphs. (Specifically, it is $G^{6,31}$)

▶ **Theorem 5.** *Let t_1 and t_2 be the two types, such that $r = w(t_1) > w(t_2) = 1$. Let Δ be the maximum degree. Let k be the number of nodes of type t_2 in the initial type function. The following assertions hold:*

- Bounds dependent on r
  1. Expected steps *The process requires at most $3k\Delta/\min(r-1,1)$ effective steps in expectation, before fixation is reached.*
  2. Probability *For any integer $x \geq 1$, after $6xn\Delta/\min(r-1,1)$ effective steps, the probability that the process has not fixated is at most 2^{-x}, irrespective of the initial type function.*
- Bounds independent on r
  1. Expected steps *The process requires at most $2nk\Delta^2$ effective steps in expectation, before fixation is reached.*
  2. Probability *For any integer $x \geq 1$, after $4xn^2\Delta^2$ effective steps, the probability that the process has not fixated is at most 2^{-x}, irrespective of the initial type function.*
- Bounds for $r \geq 2\Delta$
  1. Expected steps *The process requires at most $3k$ effective steps in expectation, before fixation is reached.*
  2. Probability *For any integer $x \geq 1$, after $6xn$ effective steps, the probability that the process has not fixated is at most 2^{-x}, irrespective of the initial type function.*

4 Lower bound for undirected graphs

In this section, we will argue that our bound on the expected number of effective steps is essentially tight, for fixed r.

We construct our lower bound graph $G^{\Delta,n}$, for given Δ, n (sufficiently large), but fixed $r > 1$, as follows. We will argue that fixation of $G^{\Delta,n}$ takes $\Omega(k\Delta)$ effective steps, if there are initially exactly k members of type t_2. For simplicity, we consider $\Delta > 2$ and $n > 4\Delta$ (it is easy to see using similar techniques that for lines, where $\Delta = 2$, the expected fixation time is $\Omega(k)$ - basically because t_1 is going to fixate with pr. $\approx 1 - 1/r$, using a proof like Lemma 6, and converting the k nodes of type t_2 takes at least k efficient steps). There are two parts to the graph: A line of $\approx n/2$ nodes and a stars-on-a-cycle graph of $\approx n/2$. There is 1 edge from the one of the stars in the stars-on-a-cycle graph to the line. More formally, the graph is as follows: Let $x := \lfloor n/(2\Delta - 2) \rfloor$. There are nodes $V_C = \{c_1, \dots, c_x\}$, such that c_i is connected to c_{i-1} and c_{i+1} for $1 < i < x$. Also, c_1 is connected to c_x. The nodes V_C are the centers of the stars in the stars-on-a-cycle graph. For each i, such that $2 \leq i \leq x$, the node c_i is connected to a set of leaves $V_C^i = \{c_i^1, \dots, c_i^{\Delta-2}\}$. The set $V_C \cup \bigcup_{i=2}^{x} V_C^i$ forms the stars-on-a-cycle graph. Note that c_1 is only connected to c_2 and c_x in the stars-on-a-cycle graph. We have that the stars-on-a-cycle graph consists of $s = (x-1) \cdot (\Delta - 1) + 1 \approx n/2$ nodes. There are also nodes $V_L = \{\ell_1, \dots, \ell_{n-s}\}$, such that node ℓ_i is connected to ℓ_{i-1} and ℓ_{i+1} for $1 < i < n/2$. The nodes V_L forms the line and consists of $n - s \geq n/2$ nodes. The node c_1 is connected to ℓ_1. There is an illustration of $G^{6,31}$ in Figure 1.

We first argue that if at least one of $V_L' = \{\ell_{\lceil n/4 \rceil}, \dots, \ell_{n-s}\}$ is initially of type t_1, then with pr. lower bounded by a number depending only on r, type t_1 fixates (note that $|V_L'| \geq n/4$ and thus, even if there is only a single node of type t_1 initially placed uniformly at random, it is in V_L' with pr. $\geq 1/4$).

▶ **Lemma 6.** *With pr. above $\frac{1-1/r}{2}$ if at least one of V_L' is initially of type t_1, then t_1 fixates.*

The proof is based on applying the gambler's ruin twice. Once to find out that the pr. that V_L eventually becomes all t_1 is above $\frac{1-1/r}{2}$ (it is nearly $1 - 1/r$ in fact) and once to find out that if V_L is at some point all t_1, then the pr. that t_2 fixates is exponentially small with base r and exponent $n - s$. See the full version [2] for the proof.

Whenever a node of V_C^i, for some i, changes type, we say that a *leaf-step* occurred. We will next consider the pr. that an effective step is a leaf-step.

▶ **Lemma 7.** *The pr. that an effective step is a leaf-step is at most $\frac{r}{\Delta}$.*

The proof is quite direct and considers that the probability that a leaf gets selected for reproduction over a center node in the stars-on-a-cycle graph. See the full version [2] for the proof.

We are now ready for the theorem.

▶ **Theorem 8.** *Let $r > 1$ be some fixed constant. Consider $\Delta > 2$ (the maximum degree of the graph), $n > 4\Delta$ (sufficiently big), and some k such that $0 < k < n$. Then, if there are initially k members of type t_2 placed uniformly at random, the expected fixation time of $G^{\Delta,n}$ is above $\frac{k\Delta(1-1/r)}{32r}$ effective steps.*

Proof. Even if $k = n - 1$, we have that with pr. at least $\frac{1}{4}$, the lone node of type t_1 is initially in V_L'. If so, by Lemma 6, type t_1 is going to fixate with pr. at least $\frac{1-1/r}{2}$. Note that even for $\Delta = 3$, at least $\frac{n}{4}$ nodes of the graphs are in $V' := \bigcup_{i=2}^{x} V_C^i$ (i.e. the leaves of the stars-on-a-cycle graph). In expectation $\frac{k}{4}$ nodes of V' are thus initially of type t_2. For fixation for t_1 to occur, we must thus make that many leaf-steps. Any effective step is a leaf-step with pr. at most $\frac{r}{\Delta}$ by Lemma 7. Hence, with pr. $\frac{1}{4} \cdot \frac{1-1/r}{2}$ ($\frac{1}{4}$ is the probability that at least one node of type t_1 is in V_L' and $\frac{1-1/r}{2}$ is a lower bound on the fixation probability if a node of V_L' is of type t_1) we must make $\frac{k\Delta}{4r}$ effective steps before fixation in expectation, implying that the expected fixation time is at least $\frac{k\Delta(1-1/r)}{32r}$ effective steps. ◀

5 Sampling an effective step

In this section, we consider the problem of sampling an effective step. It is quite straightforward to do so in $O(m)$ time. We will present a data-structure that after $O(m)$ preprocessing can sample and update the distribution in $O(\Delta)$ time. For this result we assume that a uniformly random number can be selected between 0 and x for any number $x \leq n \cdot w(t)$ in constant time, a model that was also implicitly assumed in previous works [5][1].

▶ Remark. If we consider a weaker model, that requires constant time for each random bit, then we need $O(\log n)$ random bits in expectation and additional $O(\Delta)$ amortized time, using a similar data-structure (i.e., a total of $O(\Delta + \log n)$ amortized time in expectation). The argument for the weaker model is presented in the full version [2]. In this more restrictive model [5] would use $O(\log n)$ time per step for sampling.

Sketch of data-structure. We first sketch a list data-structure that supports (1) inserting elements; (2) removing elements; and (3) finding a random element; such that each operation takes (amortized or expected) $O(1)$ time. The idea based on dynamic arrays is as follows:

1. **Insertion** Inserting elements takes $O(1)$ amortized time in a dynamic array, using the standard construction.
2. **Deletion** Deleting elements is handled by changing the corresponding element to a null-value and then rebuilding the array, without the null-values, if more than half the elements have been deleted since the last rebuild. Again, this takes $O(1)$ amortized time.
3. **Find random element** Repeatedly pick a uniformly random entry. If it is not null, then output it. Since the array is at least half full, this takes in expectation at most 2 attempts and thus expected $O(1)$ time.

At all times we keep a doubly linked list of empty slots, to find a slot for insertion in $O(1)$ time.

Data-structure. The idea is then as follows. We have 2Δ such list data-structures, one for each pair of type and degree. We also have a weight associated to each list, which is the sum of the weight of all nodes in the list, according to the modified dynamic evolution step. When the current type function is f, we represent each node v as follows: The corresponding list data-structure contains $|\Gamma_v(f)|$ copies of v (and v keeps track of the locations in a doubly linked list). Each node v also keeps track of $\Gamma_v(f)$, using another list data-structure. It is easy to construct the initial data-structure in $O(m)$ time (note: $\sum_v |\Gamma_v(f)| \leq 2m$).

Updating the data-structure. We can then update the data-structure when the current type function f changes to $f[u \to t]$ (all updates have that form for some t and u), by removing u from the list data-structure $(f(u), \deg u)$ containing it and adding it to $(t, \deg u)$. Note that if we removed x' copies of u from $(f(u), \deg u)$ we add $\deg u - x'$ to $(t, \deg u)$. Also, we update each neighbor v of u (by deleting or adding a copy to $(f(v), \deg v)$, depending on whether $f(v) = t$). We also keep the weight corresponding to each list updated and $\Gamma_v(f)$ for all nodes v. This takes at most 4Δ data-structure insertions or deletions, and thus $O(\Delta)$ amortized time in total.

[1] The construction of [5] was to store a list for t_1 and a list for t_2 and then first decide if a t_1 or t_2 node would be selected in this step (based on r and the number of nodes of the different types) and then pick a random such node. This works when all nodes of a type has the same weight but does not generalize to the case when each node can have a distinct weight based on the nodes successors like here

Sampling an effective step. Let f be the current type function. First, pick a random list L among the 2Δ lists, proportional to their weight. Then pick a random node v from L. Then pick a node at random in $\Gamma_v(f)$. This takes $O(\Delta)$ time in expectation.

▶ Remark. Observe that picking a random list among the 2Δ lists, proportional to their weight takes $O(\Delta)$ time to do naively: E.g. consider some ordering of the lists and let w_i be the total weight of list i (we keep this updated so it can be found in constant time). Pick a random number x between 1 and the total weight of all the lists (assumed to be doable in constant time). Iterate over the lists in order and when looking at list i, check if $x < \sum_{j=1}^{i} w_j$. If so, pick list i, otherwise continue to list $i+1$. By making a binary, balanced tree over the lists (similar to what is used for the more restrictive model, see the full version [2]), the time can be brought down to $O(\log \Delta)$ for this step - however the naive approach suffices for our application, because updates requires $O(\Delta)$ time.

This leads to the following theorem.

▶ **Theorem 9.** *An effective step can be sampled in (amortized and expected) $O(\Delta)$ time after $O(m)$ preprocessing, if a uniformly random integer between 0 and x, for any $0 < x \leq n \cdot w(t)$, can be found in constant time.*

6 Algorithms for approximating fixation probability

We present the algorithms for solving the fixation, extinction, and generalized fixation problems.

The Meta-simulation algorithm. Similar to [5], the algorithms are instantiating the following meta-simulation algorithm, that takes a distribution over initial type functions $\mathcal{D}$, type t and natural numbers u and z as input:

```
Function MetaSimulation($t$,$z$,$u$,$\mathcal{D}$)
  Let $y \leftarrow 0$;
  for ($i \in \{1, \ldots, z\}$) do
      Initialize a new simulation $I$ with initial type function $f$ picked according to $\mathcal{D}$;
      Let $j \leftarrow 0$;
      while ($I$ has not fixated) do
          if ($j \geq u$) then
              return Simulation took too long;
          Set $j \leftarrow j + 1$;
          Simulate an effective step in $I$;
      if ($t$ fixated in $I$) then
          Set $y \leftarrow y + 1$;
  return $y/z$;
```

Basic principle of simulation. Note that the meta-simulation algorithm uses $O(uz\Delta)$ time (by Theorem 9). In essence, the algorithm runs z simulations of the process and terminates with "Simulation took too long" iff some simulation took over u steps. Hence, whenever the algorithm returns a number it is the mean of z binary random variables, each equal to 1 with probability $\Pr[\mathcal{F}_t \mid \mathcal{E}_u]$, where $\mathcal{F}_t$ is the event that t fixates and $\mathcal{E}_u$ is the event that fixation happens before u effective steps, when the initial type function is picked according to $\mathcal{D}$ (we note that the conditional part was overlooked in [5], moreover, instead of steps we consider only effective steps). By ensuring that u is high enough and that the approximation is tight enough (basically, that z is high enough), we can use $\Pr[\mathcal{F}_t \mid \mathcal{E}_u]$ as an approximation of $\Pr[\mathcal{F}_t]$, as shown in the following lemma.

▶ **Lemma 10.** *Let $0 < \epsilon < 1$ be given. Let $\mathcal{X}, \mathcal{E}$ be a pair of events and x a number, such that $\Pr[\mathcal{E}] \geq 1 - \frac{\epsilon \cdot \Pr[\mathcal{X}]}{4}$ and that $x \in [(1-\epsilon/2)\Pr[\mathcal{X} \mid \mathcal{E}], (1+\epsilon/2)\Pr[\mathcal{X} \mid \mathcal{E}]]$. Then*

$$x \in [(1-\epsilon) \cdot \Pr[\mathcal{X}], (1+\epsilon) \cdot \Pr[\mathcal{X}]] \ .$$

The value of u: $u_{z,r}$. Consider some fixed value of z. The value of u is basically just picked so high that $\Pr[\mathcal{E}_u] \geq 1 - \frac{\epsilon \cdot \Pr[\mathcal{F}_t]}{4}$ (so that we can apply Lemma 10) and such that after taking union bound over the z trials, we have less than some constant probability of stopping. The right value of u is thus sensitive to r, but in all cases at most $O(n^2\Delta^2 \max(\log z, \log \epsilon^{-1}))$, because of Theorem 5. More precisely, we let

$$u_{z,r} = \begin{cases} 30n \cdot \max(\log z, \log \epsilon^{-1}) & \text{if } r \geq 2\Delta \\ \frac{30n\Delta}{\min(r-1,1)} \cdot \max(\log z, \log \epsilon^{-1}) & \text{if } 1 + \frac{1}{n \cdot \Delta} \leq r < 2\Delta \\ 20n^2\Delta^2 \cdot \max(\log z, \log \epsilon^{-1}) & \text{if } r < 1 + \frac{1}{n \cdot \Delta} \ . \end{cases}$$

Algorithm Algo1. We consider the fixation problem for t_1. Algorithm Algo1 is as follows:
1. Let $\mathcal{D}$ be the uniform distribution over the n type functions where exactly one node is t_1.
2. Return MetaSimulation(t_1,z,$u_{z,r}$,$\mathcal{D}$), for $z = 48 \cdot \frac{n}{\epsilon^2}$.

Algorithm Algo2. We consider the extinction problem for t_1. Algorithm Algo2 is as follows:
1. Let $\mathcal{D}$ be the uniform distribution over the n type functions where exactly one node is t_2.
2. Return MetaSimulation(t_1,z,$u_{z,r}$,$\mathcal{D}$), for $z = 24/\epsilon^2$.

Algorithm Algo3. We consider the problem of (additively) approximating the fixation probability given some type function f and type t. Algorithm Algo3 is as follows:
1. Let $\mathcal{D}$ be the distribution that assigns 1 to f.
2. Return MetaSimulation(t,z,$u_{z,r}$,$\mathcal{D}$), for $z = 6/\epsilon^2$.

▶ **Theorem 11.** *Let G be a connected undirected graph of n nodes with the highest degree Δ, divided into two types of nodes t_1, t_2, such that $r = w(t_1) > w(t_2) = 1$. Given $\frac{1}{2} > \epsilon > 0$, let $\alpha = n^2 \cdot \Delta \cdot \epsilon^{-2} \cdot \max(\log n, \log \epsilon^{-1})$ and $\beta = n \cdot \Delta \cdot \epsilon^{-2} \cdot \log \epsilon^{-1}$. Consider the running times:*

$$T(x) = \begin{cases} O(x) & \text{if } r \geq 2\Delta \\ O(\frac{x \cdot \Delta}{\min(r-1,1)}) & \text{if } 1 + \frac{1}{n \cdot \Delta} \leq r < 2\Delta \\ O(n \cdot \Delta^2 \cdot x) & \text{if } 1 < r < 1 + \frac{1}{n \cdot \Delta} \ . \end{cases}$$

- **Fixation (resp. Extinction) problem for t_1** *Algorithm* Algo1 *(resp.* Algo2*) is an FPRAS algorithm, with running time $T(\alpha)$ (resp. $T(\beta)$), that with probability at least $\frac{3}{4}$ outputs a number in $[(1-\epsilon) \cdot \rho, (1+\epsilon) \cdot \rho]$, where ρ is the solution of the fixation (resp. extinction) problem for t_1.*

- **Generalized fixation problem** *Given an initial type function f and a type t, there is an (additive approximation) algorithm,* Algo3*, with running time $T(\beta)$, that with probability at least $\frac{3}{4}$ outputs a number in $[\rho - \epsilon, \rho + \epsilon]$, where ρ is the solution of the generalized fixation problem given f and t.*

▶ Remark. There exists no known FPRAS for the generalized fixation problem and since the fixation probability might be exponentially small such an algorithm might not exist. (It is exponentially small for fixation of t_2, even in the Moran process (that is, when the graph is complete) when there initially is 1 node of type t_2)

Alternative algorithm for extinction for t_2. We also present an alternative algorithm for extinction for t_2 when r is big. This is completely different from the techniques of [5]. The alternative algorithm is based on the following result where we show for big r that $1/r$ is a good approximation of the extinction probability for t_2, and thus the algorithm is polynomial even for big r in binary.

▶ **Theorem 12.** *Consider an undirected graph G and consider the extinction problem for t_2 on G. If $r \geq \max(\Delta^2, n)/\epsilon$, then $\frac{1}{r} \in [(1-\epsilon)\cdot\rho, (1+\epsilon)\cdot\rho]$, where ρ is the solution of the extinction problem for t_2.*

Proof sketch. We present a proof sketch, and details are in the full version [2]. We have two cases:

- By [5, Lemma 4], we have $\rho \geq \frac{1}{n+r}$. Thus, $(1+\epsilon)\cdot\rho \geq \frac{1}{r}$, as desired, since $n/\epsilon \leq r$.
- On the other hand, the probability of fixation for t_2 in the first effective step is at most $\frac{1}{r+1} < \frac{1}{r}$ (we show this in the full version [2]). The probability that fixation happens for t_2 after the first effective step is at most ϵ/r because of the following reason: By Lemma 3, the probability of increasing the number of members of t_2 is at most $p := \frac{1}{r/\Delta+1}$ and otherwise it decrements. We then model the problem as a Markov chain M with state space corresponding to the number of members of t_1, using p as the probability to decrease the current state. In M the starting state is state 2 (after the first effective step, if fixation did not happen, then the number of members of t_1 is 2). Using that $\Delta^2/\epsilon \leq r$, we see that the probability of absorption in state 0 of M from state 2 is less than ϵ/r. Hence, ρ is at most $(1+\epsilon)/r$ and $(1-\epsilon)\rho$ is thus less than $1/r$. ◀

▶ Remark. While Theorem 12 is for undirected graphs, a variant (with larger r and which requires the computation of the pr. that t_1 goes extinct in the first step) can be established even for directed graphs, see the full version [2].

Concluding remarks. In this work we present faster Monte-Carlo algorithms for approximating fixation probability for undirected graphs (see the full version [2] for detailed comparison). An interesting open question is whether the fixation probability can be approximated in polynomial time for directed graphs.

References

1 B. Adlam, K. Chatterjee, and M. A. Nowak. Amplifiers of selection. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 471(2181), 2015. doi:10.1098/rspa.2015.0114.

2 Krishnendu Chatterjee, Rasmus Ibsen-Jensen, and Martin Nowak. Faster monte-carlo algorithms for fixation probability of the moran process on undirected graphs. *CoRR*, abs/1706.06931, 2017. URL: http://arxiv.org/abs/1706.06931.

3 F. Débarre, C. Hauert, and M. Doebeli. Social evolution in structured populations. *Nature Communications*, 2014.

4 Josep Díaz, Leslie Ann Goldberg, George B. Mertzios, David Richerby, Maria Serna, and Paul G. Spirakis. On the fixation probability of superstars. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, 469(2156), 2013.

5 Josep Díaz, Leslie Ann Goldberg, George B. Mertzios, David Richerby, Maria Serna, and Paul G. Spirakis. Approximating Fixation Probabilities in the Generalized Moran Process. *Algorithmica*, 69(1):78–91, 2014 (Conference version SODA 2012).

6 Josep Díaz, Leslie Ann Goldberg, David Richerby, and Maria Serna. Absorption time of the Moran process. *Random Structures & Algorithms*, 48(1):137–159, 2016.

7 W.J. Ewens. *Mathematical Population Genetics 1: I. Theoretical Introduction*. Interdisciplinary Applied Mathematics. Springer, 2004.

8 Marcus Frean, Paul B. Rainey, and Arne Traulsen. The effect of population structure on the rate of evolution. *Proceedings of the Royal Society B: Biological Sciences*, 280(1762), 2013.

9 Andreas Galanis, Andreas Göbel, Leslie Ann Goldberg, John Lapinskas, and David Richerby. Amplifiers for the Moran Process. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55, pages 62:1–62:13, 2016.

10 Rasmus Ibsen-Jensen, Krishnendu Chatterjee, and Martin A Nowak. Computational complexity of ecological and evolutionary spatial dynamics. *Proceedings of the National Academy of Sciences*, 112(51):15636–15641, 2015.

11 Samuel Karlin and Howard M. Taylor. *A First Course in Stochastic Processes, Second Edition*. Academic Press, 2 edition, April 1975.

12 Erez Lieberman, Christoph Hauert, and Martin A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316, January 2005. `doi:10.1038/nature03204`.

13 P. A. P. Moran. *The Statistical Processes of Evolutionary Theory*. Oxford University Press, Oxford, 1962.

14 Martin A. Nowak. *Evolutionary Dynamics: Exploring the Equations of Life*. Harvard University Press, 2006.

15 Paulo Shakarian, Patrick Roos, and Anthony Johnson. A review of evolutionary graph theory with applications to game theory. *Biosystems*, 107(2):66–80, 2012.