IST AUSTRIA

Institute of Science and Technology

The Complexity of Evolutionary Games on Graphs

Krishnendu Chatterjee and Rasmus Ibsen-Jensen and Martin A Nowak

The Complexity of Evolutionary Games on Graphs

Krishnendu Chatterjee[†]   Rasmus Ibsen-Jensen[†]   Martin A. Nowak[‡]

[†] IST Austria

[‡] PED, Harvard University

Abstract

Evolution occurs in populations of reproducing individuals. The structure of the population affects the outcome of the evolutionary process. Evolutionary graph theory is a powerful approach to study this phenomenon. There are two graphs. The interaction graph specifies who interacts with whom for payoff in the context of evolution. The replacement graph specifies who competes with whom for reproduction. The vertices of the two graphs are the same, and each vertex corresponds to an individual of the population. The fitness (or the reproductive rate) is a non-negative number, and depends on the payoff. A key quantity is the fixation probability of a new mutant. It is defined as the probability that a newly introduced mutant (on a single vertex) generates a lineage of offspring which eventually takes over the entire population of resident individuals. The basic computational questions are as follows: (i) the qualitative question asks whether the fixation probability is positive; and (ii) the quantitative approximation question asks for an approximation of the fixation probability. Our main results are as follows: First, we consider a special case of the general problem, where the residents do not reproduce. We show that the qualitative question is NP-complete, and the quantitative approximation question is #P-complete, and the hardness results hold even in the special case where the interaction and the replacement graphs coincide. Second, we show that in general both the qualitative and the quantitative approximation questions are PSPACE-complete. The PSPACE-hardness result for quantitative approximation holds even when the fitness is always positive.

Keywords: *Evolution; Evolutionary games on graphs; Fixation probability; Computational complexity.*

1   Introduction

In this work we study the basic computational questions for evolutionary games on graphs, and present complexity results for them. We start with a description of the model of evolution on graphs and its significance. We then state the basic computational questions and present our results.

Evolutionary dynamics with constant selection. Evolutionary dynamics act on populations. The composition of the population changes over time under the influence of 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 [8, 13, 17, 18]: Consider a population of N *resident* individuals, each with a non-negative fitness value, r. A single *mutant* with fitness value 1 is introduced in the population as the initialization step[1]. Then the following step is repeated. At each time step, one individual is chosen proportional to the fitness to reproduce and one individual is chosen uniformly at random for death. The offspring of the reproduced individual replaces the dead individual. This so-called Moran 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 $(1-r)/(1-r^N)$. The correlation between the relative fitness of the mutant (with respect to resident fitness, i.e., $1/r$) and the fixation probability is a measure of the effect of natural selection in that population structure [22, 15, 27]. A neutral mutant, $r = 1$, has fixation probability $1/N$. 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 game dynamics. The fitness values of individual types (resident and mutant) need not be constant, but could themselves depend on the composition of the population. This idea brings us to evolutionary game theory, where the individuals of a population interact with each other to receive a payoff. There could be two strategies, R and M, and a payoff matrix. The payoff of an individual is the average payoff of the interactions (see [18, Section 7.1]; also Section 2). Standard evolutionary game theory assumes a well-mixed population structure, which means all individuals interact with equal probability. Again a fundamental question is the fixation probability of a mutant [8, 13, 17, 18], which quantifies whether or not a mutant is favored by natural selection.

Evolutionary graph theory: Informal model. The outcome of an evolutionary process is dependent on population structure. Evolutionary graph theory studies this phenomenon. The individuals of the population occupy the vertices of a graph. The links (edges) determine who interacts with whom. Evolutionary graph theory describes evolutionary dynamics in spatially structured population where most interactions and competitions occur mainly among neighbors in physical space [20, 15, 23, 6, 9]. Another application is cultural evolution (spread of ideas and behaviors) in social networks [11]. Finally, the hierarchy of cellular proliferation and differentiation in the human body, which is crucial for physiological function and for reducing cancer initiation, is described by evolutionary graph theory [21]. The evolutionary graph theory considers directed graphs because interactions between individuals need not be symmetric [21, 15]: Examples of population structures and evolutionary processes that resemble directed graphs include somatic evolution of cancer either in epithelial tissue [21, 19] or in the hematopoeitic system [16], the spatial distribution of microbial and other populations along flowing water gradients or social networks, where some people are more influential than others [1].

Evolutionary graph theory: Basic results. For the case of constant fitness (which means residents with relative fitness r and mutants with relative fitness 1) graphs have been identified that maintain the same selection pressure as the well mixed population, that amplify selection, or that reduce selection. For example, a star graph is an amplifier of selection, because the fixation probability of the mutant is given by $\frac{1-r^2}{1-r^{2N}}$; hence the star graph squares the relative fitness [15]. In contrast, 'isothermal graphs' where the in-degree and out-degree of all vertices coincide (such as regular undirected graphs) have the same fixation probability as the Moran process, $\frac{1-r}{1-r^N}$ [15, 3]. There are some graphs and update rules that enhance the evolution of cooperation, which is a particular strategy in evolutionary games, for example in the well-known Prisoner's dilemma [15, 23]. Evolution of cooperation is a major topic in evolutionary biology, because cooperation is seen as a main component

---

[1]In the literature, an alternative notion is to consider that the mutant have fitness r and the residents have fitness 1, we follow the notation that leads to simpler formulas

for the creative tendency of evolution. A crucial aspect of evolutionary graph theory is the computation of the fixation probability of an invading mutant.

The formal model and computational questions. In the study of evolutionary games on graphs in general there are two graphs (that have the same vertices) [15, 24]. The "interaction graph" specifies who interacts with whom for payoff. The "replacement graph" specifies who competes with whom for reproduction. The fitness (or reproductive rate) of an individual is a non-negative number, which depends on the payoff. We consider two natural fitness functions that map payoffs to fitness: (1) *linear bounded fitness*, where the fitness is the payoff but at least 0; and (2) *exponential fitness*, where the fitness is an exponential function of the payoff (and thus always strictly positive). The initial step is the introduction of a mutant uniformly at random and then at each step a vertex is chosen proportional to the fitness. The individual in the vertex reproduces to a successor uniformly at random among the successors in the replacement graph. The relevant computational questions for evolution on graphs are as follows: (1) the *qualitative* question asks whether the fixation probability is positive; and (2) the *quantitative approximation* question asks, given $\epsilon > 0$, to compute an approximation of the fixation probability within an additive error of ϵ.

Special cases of the model. In this work we will establish several complexity results, and our most interesting results are the lower bounds. Hence we consider several special cases of the model, and present lower bounds for the special cases, which establish hardness of even several restricted cases of the general problem. The special cases we consider are as follows:

1. In the general model the interaction and replacement graphs are different (we refer to the model as the I&R model), and an important special case is where these two graphs coincide (we refer to the model as the IEQR model) [15, 23].
2. A second special case is when the residents cannot reproduce. This special case is motivated by ecological dynamics, and represents an ecosystem occupied by resident species. The spatial structure of the ecosystem is given by a graph. An invading species (mutant) is introduced, such that the mutant has a competitive advantage in the sense that once a position is occupied by the invading species then the resident cannot get it back. Another scenario that this special case represents is that a mutant enters an empty geographic location, and the question is whether the mutant can spread (hence the residents, being non-existing, cannot reproduce).

Our contributions. While previous results characterized the fixation probabilities of specific graphs (such as star or regular undirected graphs), the complexity of computing the fixation probability for arbitrary input graphs has been open (explicitly referred to as an important open problem in a survey [26, Open Problem 2.1 and 2.2]). We study the computational complexity of the basic questions for evolution on graphs and our results are as follows:

1. First, we consider the model with no resident reproduction and linear bounded fitness. We show that (i) the qualitative decision question is NP-complete both for the I&R and IEQR models; and (ii) the quantitative approximation (and also the exact) problem is #P-complete both for the I&R and IEQR models, where $\epsilon > 0$ is part of the input and specified in binary.
2. Second, we consider the model with resident reproduction and linear bounded fitness. We show that both the qualitative and the quantitative questions are PSPACE-complete for the general I&R model; and the exact fixation probability can be computed in exponential time.
3. Finally, we consider the case of exponential fitness function. The following problems can be solved in polynomial time: (i) the qualitative problem for the general I&R model; and (ii) the quantitative approximation for the general I&R model with no resident reproduction. We show that with resident reproduction the quantitative approximation problem is PSPACE-complete for the general I&R model.

Our main results are summarized in Table 1 and our key contributions are the lower bounds. We will present the relevant aspects of the lower bounds, and the upper bounds and other technical details are in the full version.

Remark 1. *Note that the quantitative approximation problem is not defined as a decision problem. For #P upper bound for the approximate (as well as exact) fixation probability for the no resident reproduction, we mean that given the number of solutions to an #P problem we can compute in polynomial time the exact fixation probability. For PSPACE upper bounds, we can compute an approximation of the fixation probability in polynomial space.*

Related complexity result. To the best of our knowledge, previous to our results, there was only one computational complexity result for evolutionary games on graphs. For the precise computation of the fixation

probability, NP-hardness for evolutionary games on graphs (named as frequency dependent selection) in the IEQR model was stated in [15]. We present stronger lower bounds: NP-hardness even for the qualitative problem and #P-hardness even for approximation. The problem of computing evolutionary stable strategies without the population structure but for any number of strategy types has been considered in [7], whereas evolutionary games on graphs consider two strategy types (resident and mutants) but the evolutionary dynamics operates on a population structure. The problem of time scale (or speed) of evolutionary processes has been studied in different contexts [4, 30], which are related to mixing time of Markov chains; and the problem of evolvability of functions in polynomial time has also been studied [29, 12]. However, none of these works consider evolutionary games on graphs. The computational study of the stochastic process on graphs induced by evolutionary games has largely been left open [26, Open Problem 2.1 and 2.2], which we mostly solve in this work.

Technical contributions. The complexity study of evolutionary games on graphs brings together many diverse fields of studies related to computer science (namely, game theory, graph theory, evolutionary stochastic processes, and computational complexity): it involves the study of stochastic processes which arise in the context of evolution, and requires the analysis of stochastic processes in combination with graph theory. Our main results are computational complexity results for the analysis of the fundamental evolutionary stochastic processes on graphs, and our main technical contribution is to develop novel gadgets on graphs that in combination with the evolutionary stochastic processes can mimic runs of a polynomial-space Turing machine (for the PSPACE lower bounds), or has the ability to count the number of matchings in bipartite graphs (for the #P-hardness).

	No Resident Reproduction		Resident Reproduction	
	IEQR model	I&R model	IEQR model	I&R model
Qual.	NP-c (**(LB)** Thm. 3)	NP-c (**(UB)** Thm. 3)	NP-h, PSPACE	PSPACE-c (**(LB)** Thm. 5, **(UB)** Thm. 5)
Appr.	#P-c (**(LB)** Thm. 4)	#P-c (**(UB)** Thm. 4)	#P-h, PSPACE	PSPACE-c (**(LB)** Thm. 5, **(UB)** Thm. 5)

Table 1: Complexity of evolution on graphs with linear bounded fitness. Qual is short-hand for qualitative and appr for approximation. Our main contributions of lower bounds (LB) and upper bounds (UB) are boldfaced. NP-c (resp., #P-c, PSPACE-c) means NP-complete (resp., #P-complete, PSPACE-complete). Similarly, NP-h (resp., #P-h) means NP-hard (resp., #P-hard).

2 Models of Evolution on Graphs

We present the basic definitions of the different models of evolution on graphs and the computational questions.

Evolutionary graphs. An *evolutionary graph* $G = (V, E_I, E_R)$ consists of a finite set V of vertices; a set $E_I \subseteq V \times V$ of *interaction* edges; and a set $E_R \subseteq V \times V$ of *replacement* (or reproduction) edges [24]. The sets E_I and E_R consist of directed edges, and the graph $G_I = (V, E_I)$ is called the interaction graph, and $G_R = (V, E_R)$ is called the replacement graph. The graph G_I is responsible for determining the interaction of individuals in the graph (which affects the fitness or payoff), and the graph G_R captures the underlying structure for reproduction and replacement of individuals in the graph. Given an edge (v, u) we say u is a *successor* of v and v is a *predecessor* of u.

Payoff of individuals. Each vertex of the graph will be occupied by one of two types of individuals, namely, the *resident* type and the *mutant* type. In evolutionary games, along with the evolutionary graph there is a payoff matrix, which is defined as follows:

$$\begin{array}{c} \begin{array}{cc} R & M \end{array} \\ \begin{array}{c} R \\ M \end{array} \left(\begin{array}{cc} a & b \\ c & d \end{array} \right) \end{array}$$

where the entries of the matrix are rational numbers and represent the payoff of an interaction, i.e., a (resp., b) is the payoff of a resident type interacting with another resident (resp., mutant) type, and c (resp., d) is the payoff of a mutant type interacting with a resident (resp., mutant) type. Given two vertices, x and y, we denote by $\mathsf{pay}(x, y)$ the payoff of the type of vertex x versus the type of vertex y.

Fitness of individuals. The fitness of an individual denotes the fecundity (or reproductive rate) and must be a non-negative number. Let $E_I(v) = \{u \mid (v, u) \in E_I\}$ denote the set of interaction successors of v. We define two natural (but not equivalent) ways of defining the fitness of v, denoted as $f(v)$, as follows:

1. *Linear bounded fitness.* The linear bounded fitness is the average payoff of the interactions but at least 0, i.e., $f(v) = \max\left\{\frac{\sum_{u \in E_I(v)} \mathsf{pay}(v,u)}{|E_I(v)|}, 0\right\}$.

2. *Exponential fitness.* The exponential fitness is an exponential function of the average payoff of the interactions, i.e., $f(v) = \exp\left(\frac{\sum_{u \in E_I(v)} \mathsf{pay}(v,u)}{|E_I(v)|}\right)$.

We will use LBF (resp., ExF) to refer to the linear bounded (resp., exponential) fitness function.

The evolutionary process. The evolutionary process we consider is the classical *birth-death* process.
1. Initially all vertices of the graph are of the resident type and a mutant type is introduced uniformly at random at one of the vertices of the graph.
2. Repeat the following step (referred to as a *generation*): In every generation, a vertex v is selected proportional to the fitness of the individual at the vertex to reproduce[2]. A new born individual replaces one of the replacement successors of v, i.e., it replaces a vertex chosen uniformly at random from the set $E_R(v) = \{u \mid (v, u) \in E_R\}$.

Step 2 (or generations) is repeated until nothing can change (in particular, if all vertices have fitness 0 or have the same type, then nothing can change).

Fixation probability. The most relevant question from an evolutionary perspective is the *fixation probability* which is the probability that the mutant takes over the population, i.e., eventually all vertices become mutants.

Computational questions. Given an evolutionary graph, a payoff matrix, and the payoff to fitness function (linear bounded, or exponential) we consider the following questions:
1. the *qualitative* decision question asks whether the fixation probability is positive; and
2. the *quantitative approximation* question, given $\epsilon > 0$, asks to compute an approximation of the fixation probability within an additive error of ϵ.

Special cases. In this work we will establish several complexity bounds for the problem, and our most interesting results are the lower bounds. Hence we consider several special cases and establish lower bounds for them.
1. **Constant fitness with density constraints.** A special case of the payoff matrix is the *constant fitness* (aka constant selection) matrix defined as follows: $a = b = r$ and $c = d = 1$, for some constant $r \geq 0$. Along with the evolutionary graph and the payoff matrix, we have two thresholds, namely, θ_R and θ_M, for the resident type and the mutant type, respectively. Intuitively, the thresholds represent a *density constraint*, and if an individual is surrounded by a lot of individuals of the same type, then its reproductive strength decreases. The density constraint is relevant in many applications of evolution (see books [2, page 470] [25, page 320], also see Remark 2). Let the selected vertex for reproduction be v. Let $\mathsf{Same}(v)$ denote the number of vertices in $E_I(v)$ that are of the same type as v. If v is a mutant type, and $\frac{\mathsf{Same}(v)}{|E_I(v)|} \leq \theta_M$ (resp., if v is a resident type, and $\frac{\mathsf{Same}(v)}{|E_I(v)|} \leq \theta_R$), then the individual gives birth to an individual of the same type. Note that the density constraint implies that if the constraint is violated, then the selected individual does not reproduce.
2. **The I&R and IEQR models.** An important special case is when the interaction and replacement graphs coincide, i.e., $E_I = E_R$ [15, 23]. We refer to the general model as the I&R *model* (with possibly different interaction and replacement graphs) and the special case where the graphs coincide as the IEQR *model*.
3. **No resident reproduction.** Another special case is when the payoff matrix is the constant payoff matrix with $r = 0$. In this case, the resident types cannot reproduce. This represents the scenario that a mutant has an advantage over the residents such that if a mutant occupies a position, then the residents cannot win it back.

Remark 2 (Matrix encoding of density constraints in LBF). *For many of our lower bounds, we will use constant selection with density constraints, and we argue that the density constraints of our lower bounds, are special cases of the linear bounded fitness without any density constraints. In our results for lower bounds we consider two types of density constraints: (1) $\theta_M = \frac{1}{2} - \delta$, for $0 < \delta < 1/10$ (in Section 3 and Section 4), where there is no resident reproduction (hence θ_R is irrelevant); and and (2) $\theta_M = \theta_R = 0$ in Section 5. In all the lower bounds, the payoff matrix is constant. These two density constraints can be encoded as a payoff matrix (that is*

---

[2]If every vertex has fitness 0, then no vertex is selected for reproduction.

not constant) with linear bounded fitness function as follows:

$$
\begin{array}{c}
\begin{array}{cc} R & M \end{array} \\
\begin{array}{c} R \\ M \end{array}
\begin{pmatrix} 0 & 0 \\ 1 & -1 \end{pmatrix}
\end{array} ;
\qquad\qquad
\begin{array}{c}
\begin{array}{cc} R & M \end{array} \\
\begin{array}{c} R \\ M \end{array}
\begin{pmatrix} -N & 1 \\ 1 & -N \end{pmatrix}
\end{array} .
$$

The first payoff matrix encodes that a vertex that is a mutant can reproduce only if strictly less than half of the successors in E_I are mutants, and thus encode $\theta_M = \frac{1}{2} - \delta$, for $0 < \delta < 1/10$, in graphs where the outdegree is at most five. The second matrix (for a graph with N vertices) encodes that a vertex can reproduce only if all the successors in E_I are of the opposite type.

3 Qualitative Analysis: No Resident Reproduction with LBF

In this section we establish two results for the no resident reproduction model with LBF: the qualitative analysis problem is (1) in NP for the general I&R model (details in appendix); and (2) is NP-hard in the special case of IEQR model, and even in a special case of LBF, where we have constant fitness with density constraints, (using density constraints mentioned in Remark 2).

Lower bound. We present an NP lower bound, and we will prove it for the IEQR model with no resident reproduction. Moreover, since there is no resident reproduction, the threshold θ_R does not matter. We will present a reduction from the 3-SAT problem (which is NP-complete [5, 14, 10]) and use threshold θ_M as $\frac{1}{2} - \delta$, for any $0 < \delta \le \frac{1}{10}$. However it would be easy to modify our construction for any threshold θ_M in $(0, 1)$. The "right" way to think of the threshold is that it is $\frac{1}{2}$ and that the density constraint uses a strict inequality. The upper bound is chosen because we will use vertices with degree five or less.

Notations. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of n Boolean variables. Consider a 3-CNF formula $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, where each C_i is a *clause* of a list of (precisely) three *literals* (where a literal is a variable x or its negation $\overline{x}$, where $x \in X$). Each clause represents a disjunction of the literals that appear in it. An instance of the 3-SAT problem, given a 3-CNF formula φ, asks whether there exists a satisfying assignment. We will now construct an evolutionary graph $G(\varphi)$, given an instance of a 3-SAT problem, with (i) $E_I = E_R$, (ii) no resident reproduction, and (iii) threshold $\theta_M = \frac{1}{2} - \delta$, for $0 < \delta \le \frac{1}{10}$ such that there is a satisfying assignment iff the answer to the qualitative decision problem is YES. We first present two gadget constructions that will be used in the reduction.

Predecessor gadget. We present a *predecessor* gadget for a vertex pair (u, v) such that v is the only successor of u. The gadget ensures the following property (namely, the *predecessor gadget* property): if all vertices become mutants, then the vertex u must have become a mutant before vertex v. The construction of the gadget is as follows: Add a new *dummy* vertex u'. Let the successors of u be v and u', and the successor of u' be only v. Then the only way for u' to become a mutant is if u is a mutant, since u is the only predecessor of u'. But u' can only become a mutant if u is a mutant and v is not (since otherwise the threshold condition with $\theta_M = \frac{1}{2} - \delta$ is not satisfied for u, for any $0 < \delta \le \frac{1}{10}$). Hence, if all vertices become mutant, then u must become a mutant before v. We will denote by $\mathsf{PredEdges}(u, v, u')$ the set $\{(u, v), (u, u'), (u', v)\}$ of edges of the predecessor gadget.

(Extended) Binary tree gadget. Given a vertex rt, and a set L of vertices, we will denote by $\mathsf{BinTr}(\mathsf{rt}, L)$ a binary tree with rt as root and L as leaf vertices. In a binary tree, every non-leaf vertex has out-degree 2. Note that the binary tree gadget adds additional vertices, and has $O(|L|)$ vertices. By an abuse of notation we will use $\mathsf{BinTr}(\mathsf{rt}, L)$ to denote both the set of vertices and the set of edges of the binary tree, and it would either be clear from the context or explicitly mentioned. Given a binary tree T and an *extension* vertex $z \notin T$, an *extended binary tree* (EBT) consists of T and an edge from every non-leaf vertex to z. Given a root vertex rt, a set of L of leaf vertices, and an extension vertex z, we denote by $\mathsf{ExBinTr}(\mathsf{rt}, L, z)$ the edge set of the extended binary tree that extends the binary tree of rt and L. We will explicitly use the following property for an EBT (namely, *qualitative EBT (QEBT)* property):

- *(QEBT Property).* In an EBT, every non-leaf vertex has out-degree 3, and for density constraint with threshold $\frac{1}{2} - \delta$, for $0 < \delta \le \frac{1}{10}$ (the construction works even if δ is up to $\frac{1}{6}$), if the root becomes a mutant and z is not a mutant, then root can be responsible for making every vertex in the tree a mutant. However, note that if z is a mutant, then any vertex in the tree with out-degree 3 cannot make both the children mutants due to the density constraint.

The evolutionary graph $G(\varphi)$. We now present the evolutionary graph $G(\varphi)$ where we first describe the vertex set and then the edges.

The vertex set. The set V of vertices is as follows: The vertex $v_\top$ will be the start vertex; and the vertices $z_\perp$, $y_\perp$, and $z'_\perp$ are end vertices (that will form a predecessor gadget for $(z_\perp, y_\perp)$ with dummy vertex $z'_\perp$). We have a vertex c_i for each clause C_i (namely, clause vertices); and one for each literal c_i^1, c_i^2, and c_i^3 in the clause (namely, clause-literal vertices). Similarly, we have a vertex x_i for each variable in X (namely, variable vertices), and vertices x_i^t and x_i^f (namely, variable-value vertices) to represent the truth values to be assigned to x_i. Corresponding to x_i^t and x_i^f we also have vertices u_i^t and u_i^f (namely, duplicate vertices). The vertex v_0 forms a predecessor gadget (using the dummy vertex v_0') to u_1^t. Let $L_i^t = \{\widehat{c}_k^j \mid 1 \leq k \leq m,\ 1 \leq j \leq 3,\ c_k^j = x_i\}$ denote a copy of the clause-literal vertices that correspond to x_i and $L_i^f = \{\widehat{c}_k^j \mid 1 \leq k \leq m,\ 1 \leq j \leq 3,\ c_k^j = \overline{x}_i\}$ denote a copy of the clause-literal vertices that correspond to negation of x_i. The set $\mathsf{BinTr}(x_i^t, L_i^t)$ (resp. $\mathsf{BinTr}(x_i^f, L_i^f)$) represents the vertices of a binary tree with the root vertex x_i^t (resp. x_i^f) and leaf vertices L_i^t (resp. L_i^f).

The edge set. We now describe the edge set:
- There is an edge from the initial vertex $v_\top$ to the first clause vertex c_1; and we have two predecessor gadgets; (i) $(z_\perp, y_\perp)$ with dummy vertex $z'_\perp$; and (ii) (v_0, u_1^t) with dummy vertex v_0'.
- For each clause vertex c_i, there are five edges, three to clause-literal vertices c_i^j (for $j = 1, 2, 3$) of the clause, one to the next clause vertex (for c_m this next vertex is x_1), and to the vertex u_1^t.
- For each variable vertex x_i, there are three edges: to x_i^t and x_i^f, and to the next variable vertex x_{i+1} (for x_n the next vertex is v_0).
- Each duplicate vertex u_i^t has three edges: to u_i^f, to x_i^t, and to $y_\perp$. Similarly, each vertex u_i^f has three edges: to u_{i+1}^t (u_n^f has edge to $z_\perp$ instead), to x_i^f, and to $y_\perp$.
- Finally, we have the EBT with x_i^α (for $\alpha \in \{t, f\}$) as root, L_i^α as leaf vertices and $y_\perp$ as the extension vertex. For each vertex in L_i^α, for $\alpha \in \{t, f\}$, we add edges to the corresponding clause-literal vertex and to u_1^t. This ensures that every internal vertex of the binary tree has degree three, and leaf vertices have degree two.

The formal description of the vertices and edges is in the appendix (also see Figure 1 for a pictorial illustration).

Basic facts. We mention some basic facts about the evolutionary graph obtained.
1. First, observe that the predecessor gadget property implies that for fixation the vertex v_0 must become a mutant before vertex u_1^t; and vertex $z_\perp$ before vertex $y_\perp$.
2. Second, for a vertex with degree ℓ, it can reproduce a mutant as long as at most $\ell \cdot (\frac{1}{2} - \delta)$ successors are mutants. In particular, for vertices with five (resp. three) successors, like the clause (resp. variable) vertices, it can reproduce a mutant until at most three (resp. two) successors are mutants, because of the bounds on θ_M. If a vertex has out-degree two (or one), then it can reproduce a mutant until at most one successor is a mutant, because of the bounds on θ_M. The conditions follow from the density constraint with threshold $\frac{1}{2} - \delta$.

Using the above facts and the QEBT property, we show that the graph $G(\varphi)$ has positive fixation probability iff φ is satisfiable. The main results are summarized below.

Theorem 3. *The qualitative decision question for no resident reproduction in both the general* I&R *model and the* IEQR *model with* LBF *is NP-complete.*

4 Quantitative Approximation: No Resident Reproduction with LBF

In this section we show that in the no resident reproduction model with LBF the following assertions hold: (i) the precise fixation probability can be computed in #P (for the general I&R model, and the details are in the appendix); and (ii) for $\epsilon > 0$, the problem of approximating the fixation probability within an additive error of ϵ is #P-hard (even in the IEQR model). Again in our lower bound we will consider a special case of LBF where we have constant fitness with density constraint.

Perfect matching in bipartite graphs. We present a reduction from the computation of the number of perfect matchings in a bipartite graph $G = (V, E)$. In a bipartite graph G, the vertex set V is partitioned into vertices V_ℓ (left vertices) and V_r (right vertices) and all edges go from a vertex in V_ℓ to a vertex in V_r (i.e., $E \subseteq V_\ell \times V_r$).

We also have $|V_\ell| = |V_r| = n$. A *perfect matching* PM is a set $\{e_1, e_2, \ldots, e_n\}$ of n edges from E such that for every vertex $v_\ell \in V_\ell$ (resp. $v_r \in V_r$) there exists an edge $e_\ell = (v_\ell, v_r')$ (resp. $e_r = (v_\ell', v_r)$) in PM. Given a bipartite graph, the problem of computing the number of distinct perfect matchings was shown by Valiant [28] to be #P-complete.

Uniform degree property. First, we will consider bipartite graphs for which there exists an integer k such that all vertices in V_ℓ have either degree 2^k or 1. We refer to the property as the *uniform degree* property. A reduction from general bipartite graphs to graphs with uniform degree property is in the appendix.

Perfect binary trees. We will consider perfect binary trees as gadgets.

- A *perfect binary tree* (PBT) is a balanced binary tree (every internal vertex has exactly two children) with all leaves at the same level (i.e. with 2^k leaf vertices, for some non-negative integer k). For a PBT we will use the following property, which we refer to as the *probabilistic PBT (PPBT)* property: if the root becomes a mutant, then eventually all vertices in a path from the root to some leaf will become mutants, where such a path is chosen uniformly at random. Since every non-leaf vertex has out-degree two, due to the density constraint, each internal vertex can make one of its children (chosen uniformly) a mutant and hence the PPBT property follows.

The graph $\mathrm{Red}(G)$**.** Given a bipartite graph G with the uniform degree property, let the vertex sets be V_ℓ and V_r, respectively. Let $E(v) = \{u \mid (v, u) \in E\}$ denote the successors of a vertex $v \in V_\ell$. Let $V_\ell^k = \{v \in V_\ell \mid |E(v)| = 2^k\}$ be the set of vertices with degree 2^k; and $V_\ell^1 = V_\ell \setminus V_\ell^k$ be the set of vertices in V_ℓ with degree 1. Our reduction, denoted $\mathrm{Red}(G)$, will construct an evolutionary graph (with $E_I = E_R$ and hence we only specify one set of edges), which consists of three parts: part 1 sub-graph, then edges related to V_r, and a copy of part 1 with some additional edges. We first describe the part 1 sub-graph and then its copy.

- *(Part 1).* We have a start vertex v_s, a final vertex $y_\perp$, and we create an EBT B_s as follows: $\mathsf{ExBinTr}(v_s, V_\ell, y_\perp)$, i.e., the start vertex is the root, V_ℓ is the set of leaf vertices, and $y_\perp$ is the extension vertex. For every vertex $v \in V_\ell^k$, let $E(v) = \{u^1, u^2, \ldots, u^j\}$, and we consider a set $L_v^k = \{u_v^1, u_v^2, \ldots, u_v^j\}$ of $j = 2^k$ vertices and construct a PBT $P_v = \mathsf{BinTr}(v, L_v^k)$. Note that B_s is an EBT (but the underlying binary tree is not necessarily perfect).

- *(Edges related to V_r).* From every vertex $v \in V_\ell^k$, and every u_v^i in L_v^k, we add two edges: one to $u^i \in E(v)$ and one to $y_\perp$. From every vertex $v \in V_\ell^1$ (with degree 1), we add two edges: to the unique $u \in E(v)$ and to $y_\perp$. Every vertex in V_r has an edge to $y_\perp$.

- *(Copy 1 of Part 1 with additional edges).* First, we create a copy of the part of the graph described in part 1, along with one additional vertex $z_\perp$. For every vertex v of part 1, let the corresponding vertex in the copy be called $\overline{v}$, and the copy of the extension vertex is $\overline{y}_\perp$. We describe the difference in the copy as compared to the graph of part 1: (i) first there is an edge from $y_\perp$ to the copy $\overline{v}_s$ of the start vertex; (ii) for every vertex $\overline{z}$ which is a copy of a non-leaf vertex z in P_v, for some $v \in V_\ell^k$, (i.e., $z \notin L_v^k$), there are three additional edges from $\overline{z}$: (a) to z (i.e., from the copy to the original vertex), (b) to $\overline{y}_\perp$, and (c) to $z_\perp$; and (iii) for every vertex $\overline{z}$ which is a copy of a leaf vertex z in P_v, for some $v \in V_\ell^k$, (i.e., $z \in L_v^k$), there is only one edge which goes to z (i.e., there is no edge to V_r or $y_\perp$, but an edge from the copy to the original vertex). Hence in the copy of P_v, for any v, internal vertices have degree five, and leaf vertices have degree 1.

- Finally, we have the following edges: $\{(y_\perp, \overline{y}_\perp), (y_\perp, z_\perp), (\overline{y}_\perp, z_\perp)\}$.

We denote by $\widehat{n}$ the number of vertices in $\mathrm{Red}(G)$, and note that $\widehat{n} = O(m)$, where m is the number of edges in G.

Example. We consider the graph G with six vertices, where $V_\ell = \{v_1, v_2, v_3\}$ and $V_r = \{v_4, v_5, v_6\}$, such that v_1 and v_2 each have edges to v_4 and v_5 and v_3 has an edge to v_6. See Figure 2 for an illustration. Observe that G satisfies the uniform degree property. In Figure 3 we have part 1 of the graph $\mathrm{Red}(G)$ along with V_r. In Figure 4 we have the remainder of $\mathrm{Red}(G)$.

The process of fixation in $\mathrm{Red}(G)$**.** The process of fixation in $\mathrm{Red}(G)$ can be decomposed in two phases. The first phase (Phase 1) is over when $y_\perp$ becomes a mutant; and the second phase (Phase 2) is over with the fixation. A key property of Phase 2 is as follows: vertices in V_r cannot become a mutant after $y_\perp$ has become a mutant: This is because for each vertex u in V_r, every predecessor v of u has exactly two successors, and one them is $y_\perp$ (and hence the density constraint with threshold $\frac{1}{2} - \delta$ ensures that if $y_\perp$ is a mutant, then vertices in V_r cannot become mutants after that).

- *Phase 1.* In Phase 1, the vertex v_s must be the first vertex to become a mutant (since it has no predecessor). After v_s, all vertices in B_s turn into mutants (by the QEBT property). Once a vertex $v \in V_\ell^k$ becomes a mutant, then a path in the PBT P_v under v is chosen uniformly at random to become mutants (by the PPBT property), and then the leaf of the path can make the corresponding vertex in V_r a mutant. Once a vertex v in V_ℓ^1 with degree 1 becomes a mutant, then it can reproduce a mutant to the unique neighbor in V_r. In the end, some vertex in V_r reproduces a mutant to $y_\perp$ and Phase 1 ends.
- In Phase 2, first the copy $\overline{v}_s$ becomes a mutant from $y_\perp$. After $\overline{v}_s$, all vertices which are copy of vertices in B_s become mutants (again by the QEBT property). Once copies of vertices in V_ℓ^k become mutant, then the tree underneath them in the copy become mutants. Consider a vertex $\overline{u}$ which is a copy of a vertex $u \in P_v$, for some $v \in V_\ell^k$, and there are two cases: (i) if u is a non-leaf vertex, then $\overline{u}$ has degree five, and can reproduce mutants until the two children in the tree and the original vertex u are mutants (note if $\overline{y}_\perp$ or $\overline{z}_\perp$ is a mutant, then both the children and the original copy cannot all become mutants due to the density constraint); (ii) if u is a leaf-vertex, then $\overline{u}$ has degree one, and can reproduce mutant for u. Finally, $y_\perp$ makes $\overline{y}_\perp$ a mutant, which then makes $z_\perp$ a mutant.

Fixation and a perfect matching. Observe that fixation implies that all vertices in V_r have become mutant, and no vertex in V_r can become a mutant in the second phase. Each vertex in V_ℓ is responsible for making at most one neighbor in V_r a mutant (for vertices with degree 1 it is the unique successor in V_r, and for vertices with degree 2^k, it corresponds to the leaf of the path in the perfect binary tree chosen uniformly at random by the PPBT property). This defines a perfect matching. Conversely, given a perfect matching, Phase 1 and Phase 2 of fixation can be described using the pairs of the matching (to chose paths uniformly at random in the perfect binary trees). Thus given fixation, it defines a perfect matching, and we say that fixation has *used* the perfect matching.

Exact fixation probability. Consider some perfect matching PM. Observe that if there are $s > 0$ perfect matchings, then the exact fixation probability is $s \cdot x_{\mathsf{PM}}$, where x_{PM} is the probability that we have fixation and used PM. This is because each perfect matching has the same probability to be the chosen matching in Phase 1 by the PPBT property. In Phase 2, any vertex v which is either a vertex in V_ℓ^1 or a leaf in P_v, for $v \in V_\ell^k$, cannot reproduce by the key property of Phase 2 (and thus can be viewed as having no out-going edges). Thus in Phase 2, by symmetry, the probability x_{PM} of fixation for a perfect matching PM is independent of PM. Given appropriate approximations of x_{PM} and the fixation probability in $\mathrm{Red}(G)$ the number of perfect matching in G can be obtained.

Theorem 4. *The quantitative approximation problem, where the approximation number $0 < \epsilon < 1$ is given in binary, for no resident reproduction in both the general* I&R *model and the* IEQR *model with* LBF *is #P-complete (and even the exact fixation probability can be computed in #P).*

5 Qualitative Analysis and Quantitative Approximation: I&R Model with Resident Reproduction and LBF

In this section we will establish the polynomial space upper bound (details in appendix) and lower bound in the I&R model with resident reproduction, when the fitness function is LBF.

Lower bound. We present two lower bounds: (i) the qualitative decision question is PSPACE-hard; and (ii) the question that given an evolutionary graph with the promise that the fixation probability is close to either 0 or 1, deciding which is the case is PSPACE-hard (which implies PSPACE-hardness for the quantitative approximation problem). For simplicity, we present our lower bounds in two steps. We will first reduce the problem to a problem which we call *concurrent-if*, and then show that the concurrent-if problem is PSPACE-complete.

Concurrent-if problem. The intuitive description of the concurrent-if problem is as follows: it consists of a set of Boolean variables, and a set of if statements where each conditional is a conjunction of some of the Boolean variables or their negation, and if the conditional is true, then a Boolean variable is set to a truth value. At each step, any of the if-statements can be executed. The process ends either when the first Boolean variable is true or nothing can change (i.e., the conditional of all if-statements are false). Note that the execution can loop, and perhaps run forever. We first define an if-statement.

If-statement. Let $B = \{b_1, b_2, \ldots, b_n\}$ be a set of n Boolean variables. An if-statement s is as follows: $\bigwedge(\mathsf{cn}_1, \ldots, \mathsf{cn}_k) \Rightarrow b_i := \mathsf{val}$, where $1 \le i, k \le n$, val is either true or false, each cn_j is either a Boolean

variable b_ℓ or its negation $\bar{b}_\ell$. An if-statement is *satisfied* if each of the cn_j is true (i.e., cn_j is true if one of the following holds: if cn_j is b_ℓ and b_ℓ is true, or cn_j is $\bar{b}_\ell$ and b_ℓ is false).

Concurrent-if system. A concurrent-if system consists of a set $B = \{b_1, b_2, \ldots, b_n\}$ of n Boolean variables and a set $P = \{s_1, s_2, \ldots, s_m\}$ of m if-statements over the Boolean variables in B. The set of statements defines an execution from an initial setting of the Boolean variables as follows: repeatedly, a satisfied if-statement $\bigwedge(\mathsf{cn}_1, \ldots, \mathsf{cn}_k) \Rightarrow b_i := \mathsf{val}$ is selected and then b_i is set to val. If the first Boolean variable b_1 is eventually true, then the execution is *accepted*. If at each point of an execution there is at most one satisfied if-statement, then we say that the execution is *deterministic*.

The decision problem. Given a concurrent-if system, the associated decision problem is as follows: Given a set B of Boolean variables, an initial setting of the variables of B, and a set P of concurrent if-statements, such that the execution e from the initial setting is deterministic, whether e is accepting. First we establish that the decision problem for concurrent-if systems is PSPACE-complete (details in appendix).

Reduction of the concurrent-if problem to evolutionary games on graphs We describe how we encode the Boolean variables and the if-statements of a concurrent-if system in evolutionary games on graphs.

Density constraint. Again our lower bound result will be for a special case of LBF, where we have constant fitness with density constraints (recall Remark 2). Our construction will be for $\theta_R = \theta_M = 0$, but a similar construction will work for any choice of $\theta_R, \theta_M \in [0, 1)$. The thresholds $\theta_R = \theta_M = 0$ indicates that a vertex v can reproduce precisely as long as all its successors in E_I are of the opposite type of v.

Ideas and gadgets behind the reduction. We first introduce some key ideas and gadgets behind the reduction.
- *States which are nearly always a mutant/resident:* Similar to the previous lower bounds, we have a vertex v_s without any predecessor in E_R. Thus, if v_s is not made a mutant at the start, then it cannot become a mutant. Hence we only consider the case when v_s is a mutant in the beginning and stays a mutant forever. We also have a vertex $\widehat{v}_s$, and our construction ensures that it stays a resident until all other vertices are mutants and then (after a few more steps) all vertices become mutants, and we get fixation. We use the vertices v_s and $\widehat{v}_s$ to ensure that a given vertex has a desired type, and otherwise the vertex cannot reproduce. Our construction ensures (by the density constraint) the following properties:
    - A vertex v with $\widehat{v}_s$ as a successor under E_I can only reproduce if it is a mutant (using the density constraint and $\widehat{v}_s$ is a resident). Similarly, a vertex v with v_s as a successor under E_I can only reproduce if it is a resident.
- *Boolean-value gadgets:* We describe how to implement *boolean-value* gadgets in evolutionary graphs for the Boolean variables of the concurrent-if system. Each boolean-value gadget j consist of four vertices v_{tv}^j (the *true-value-vertex*), v_{fv}^j (the *false-value-vertex*), v_{ts}^j (the *true-setter-vertex*) and v_{fs}^j (the *false-setter-vertex*). In the second phase (the execution of the concurrent-if system phase) each boolean-value gadget is such that the two setters, v_{ts}^j and v_{fs}^j, are mutants. Also, at most one of the value vertices v_{tv}^j and v_{fv}^j, can be a mutant at any given point. If v_{tv}^j is a mutant, then the value of j is *true*. If v_{fv}^j is a mutant, then the value of j is *false*. If neither is a mutant, then we say that j has *no value*. The edge set is as follows: (i) both v_{ts}^j and v_{fs}^j have $\widehat{v}_s, v_{tv}^j, v_{fv}^j$ as successors under E_I; (ii) v_{ts}^j (resp., v_{fs}^j) has only v_{tv}^j (resp., v_{fv}^j) as a successor under E_R (see Figure 5). The purpose of the edges in E_I are as follows: the edge to $\widehat{v}_s$ enforces that the setter vertex is a mutant before reproduction; and the other two edges enforce that only if the gadget has no value (i.e., both value vertices are resident), then the setter vertex can reproduce a mutant (by the density constraint and that $\theta_R = \theta_M = 0$). Observe that when the gadget has no value, then each of the setter vertices can set the value of the gadget to either true or false with positive probability in any such step.
- *If-statement gadgets:* Each if-statement gadget, for the if-statement $\bigwedge(\mathsf{cn}_1, \ldots, \mathsf{cn}_k) \Rightarrow b_i := \mathsf{val}$, is implemented using a single vertex v (the if-statement-vertex). The if-statement gadget works under the requirement that v is a resident, and our construction will ensure that in the second phase (the execution of the concurrent-if system phase) each if-statement-vertex v is a resident. The edge set is as follows:
    1. The vertex v has the following edges in E_i: an edge to v_s; and for each Boolean variable j in $(\mathsf{cn}_1, \ldots, \mathsf{cn}_k)$ an edge to v_{tv}^j, and for each negation of a Boolean variable j' in $(\mathsf{cn}_1, \ldots, \mathsf{cn}_k)$ an edge to $v_{fv}^{j'}$.
    2. The vertex v has v_{fv}^i (resp., v_{tv}^i) as successor under E_R if val is true (resp., false).

The purpose of the edges in E_I are as follows: the edge to v_s enforces that the if-statement-vertex is a

9

resident before v reproduces; the other edges enforces that each literal in $(\mathsf{cn}_1, \ldots, \mathsf{cn}_k)$ has the correct value before reproduction. Consider the case where val is true (the case where it is false is similar). If v can reproduce at a given point in time, then $\bigwedge(\mathsf{cn}_1, \ldots, \mathsf{cn}_k)$ must be true. In that case, if the boolean-value-gadget for b_i has value false, then v reproduces to set b_i to no value. This then allows the setter-vertices of b_i to reproduce, and set b_i eventually to a value again. Observe that even though v tries to set b_i to true, the value of b_i might not be set to true immediately. The process is as follows: v tries to set b_i to true by ensuring that if it is false, then it sets it to no value, and ensures that the true-setter vertex has positive probability to set it to true. Hence eventually with probability 1 it is set to true. Note that given $\bigwedge(\mathsf{cn}_1, \ldots, \mathsf{cn}_k)$ is still true, v can simply reproduce until b_i becomes true. Since there is a fixed positive probability that the setter-vertices will set b_i to either value, eventually b_i becomes true with probability 1. We will only use the boolean-value gadgets for deterministic executions and thus, the condition $\bigwedge(\mathsf{cn}_1, \ldots, \mathsf{cn}_k)$ remains true until b_i becomes true. This is because the execution is deterministic and thus, no other if-statement is satisfied in the current situation as long as b_i is false or has no value. Especially, for the next if-statement to be satisfied it must depend on b_i being true.

Using the above gadgets, given a concurrent-if system instance, we construct a evolutionary graph such that informally the following assertion hold: fixation happens with probability 0 if the concurrent-if systems does not accept, otherwise the fixation probability is close to 1. Theorem 5 summarizes the result of this section.

Theorem 5. *For the general* I&R *model with resident reproduction and* LBF*, the following assertions hold: (1) The qualitative decision problem is PSPACE-complete; and (2) the quantitative approximation problem can be solved in polynomial space, and even given the promise that the fixation probability is close to either 0 or 1, deciding which is the case is PSPACE-hard (hence the quantitative approximation problem is PSPACE-hard).*

6 Complexity Results for the Exponential Fitness Function ExF

In this section we consider the case where the fitness of an individual at a vertex is an exponential function of the payoff, and we do not have density constraint. We first present the results, and describe how to obtain them.
1. *Result 1.* The qualitative problem can be solved in polynomial time.
2. *Result 2.* For the no resident reproduction case (i.e., when the fitness of a resident is set to 0), the quantitative approximation problem can be solved in polynomial time.
3. *Result 3.* For the resident reproduction case, we have the same complexity bounds as in the case where we have the LBF.

Key ideas for lower bound of Result 3. Result 1, Result 2, and upper bound of Result 3 are in the appendix. We present the key ideas of the lower bound of Result 3.

The key idea. The key idea is as follows:
1. *First step:* First, we consider the problem with constant payoff along with density constraints and argue that the PSPACE hardness result holds even in the case where either mutants or residents fixate within an exponential number of steps with high probability.
2. *Second step:* In the hardness proof in the model with density constraints we require that a vertex can reproduce iff all its successors are of the opposite type. In the model with fitness exponential of payoff, there is always a positive probability to reproduce. Thus even if a vertex has all its successors of the opposite type, it can still reproduce, and we refer to such reproductions as "undesired reproductions" (for the hardness proof). We show that a payoff matrix (with exponential payoff and no density constraints) can encode that if a vertex does not have all its successes of the other type, then the probability to reproduce is exponentially small (i.e., the undesired reproduction probability is exponentially small). Since in the hardness result of the previous item, the fixation happens within exponentially many steps, using union bounds it is easy to argue that the probability that an undesired reproduction happens before fixation is negligible.

Theorem 6. *For the general* I&R *model with the fitness function is exponential function of the payoff, where each payoff of the matrix is polynomial in the size of the graph, the following assertion holds: the quantitative approximation problem can be solved in polynomial space, and even given the promise that the fixation probability is close to either 0 or 1, deciding which is the case is PSPACE-hard (hence the quantitative approximation problem is PSPACE-hard).*

References

[1] B. Allen, C. Sample, Y. A. Dementieva, R. C. Medeiros, C. Paoletti, and M. A. Nowak. The molecular clock of neutral evolution can be accelerated or slowed by asymmetric spatial structure. arxiv:1409.5459, 2014.

[2] N. Barton, D. E. Briggs, J. A. Eisen, D. B. Goldstein, and N. H. Patel. *Evolution*. Cold Spring Harbor Laboratory Press, 2007.

[3] M. Broom and J. Rychtár. An analysis of the fixation probability of a mutant on special classes of non-directed graphs. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, 464(2098):2609–2627, 2008.

[4] K. Chatterjee, A. Pavlogiannis, B. Adlam, and M. A. Nowak. The time scale of evolutionary innovation. *PLoS Computational Biology*, 10(9), 2014.

[5] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.

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

[7] K. Etessami and A. Lochbihler. The computational complexity of evolutionarily stable strategies. *Int. J. Game Theory*, 37(1):93–113, 2008.

[8] W. Ewens. *Mathematical Population Genetics 1: I. Theoretical Introduction*. Interdisciplinary Applied Mathematics. Springer, 2004.

[9] M. Frean, P. B. Rainey, and A. Traulsen. The effect of population structure on the rate of evolution. *Proceedings of the Royal Society B: Biological Sciences*, 280(1762), 2013.

[10] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[11] A. L. Hill, D. G. Rand, M. A. Nowak, and N. A. Christakis. Emotions as infectious diseases in a large social network: the SISa model. *Proceedings of the Royal Society B: Biological Sciences*, 277:3827–3835, 2010.

[12] V. Kanade. Evolution with recombination. In *FOCS*, pages 837–846, 2011.

[13] S. Karlin and H. M. Taylor. *A First Course in Stochastic Processes, Second Edition*. Academic Press, 2 edition, Apr. 1975.

[14] L. A. Levin. Universal sequential search problems. *Problems of Information Transmission*, 9(3):265–266, 1973.

[15] E. Lieberman, C. Hauert, and M. A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316, Jan. 2005.

[16] F. Michor, T. P. Hughes, Y. Iwasa, S. Branford, N. P. Shah, C. L. Sawyers, and M. A. Nowak. Dynamics of chronic myeloid leukaemia. *Nature*, 435(7046):1267–1270, 06 2005.

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

[18] M. A. Nowak. *Evolutionary Dynamics: Exploring the Equations of Life*. Harvard University Press, 2006.

[19] M. A. Nowak, N. L. Komarova, A. Sengupta, P. V. Jallepalli, I.-M. Shih, B. Vogelstein, and C. Lengauer. The role of chromosomal instability in tumor initiation. *Proceedings of the National Academy of Sciences*, 99(25):16226–16231, 2002.

[20] M. A. Nowak and R. M. May. Evolutionary games and spatial chaos. *Nature*, 359:826, 1992.

[21] M. A. Nowak, F. Michor, and Y. Iwasa. The linear process of somatic evolution. *Proceedings of the National Academy of Sciences*, 100(25):14966–14969, 2003.

[22] M. A. Nowak, A. Sasaki, C. Taylor, and D. Fudenberg. Emergence of cooperation and evolutionary stability in finite populations. *Nature*, 428(6983):646–650, 2004.

[23] H. Ohtsuki, C. Hauert, E. Lieberman, and M. A. Nowak. A simple rule for the evolution of cooperation on graphs and social networks. *Nature*, 441:502–505, 2006.

[24] H. Ohtsuki, J. M. Pacheco, and M. A. Nowak. Evolutionary graph theory: Breaking the symmetry between interaction and replacement. *Journal of Theoretical Biology*, 246(4):681 – 694, 2007.

[25] S. Otto and T. Day. *A Biologist's Guide to Mathematical Modeling in Ecology and Evolution*. Princeton University Press, 2011.

[26] P. Shakarian, P. Roos, and A. Johnson. A review of evolutionary graph theory with applications to game theory. *Biosystems*, 107(2):66 – 80, 2012.

[27] C. E. Tarnita, H. Ohtsuki, T. Antal, F. Fu, and M. A. Nowak. Strategy selection in structured populations. *Journal of Theoretical Biology*, 259(3):570–81, 2009.

[28] L. G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

[29] L. G. Valiant. Evolvability. *J. ACM*, 56(1), 2009.

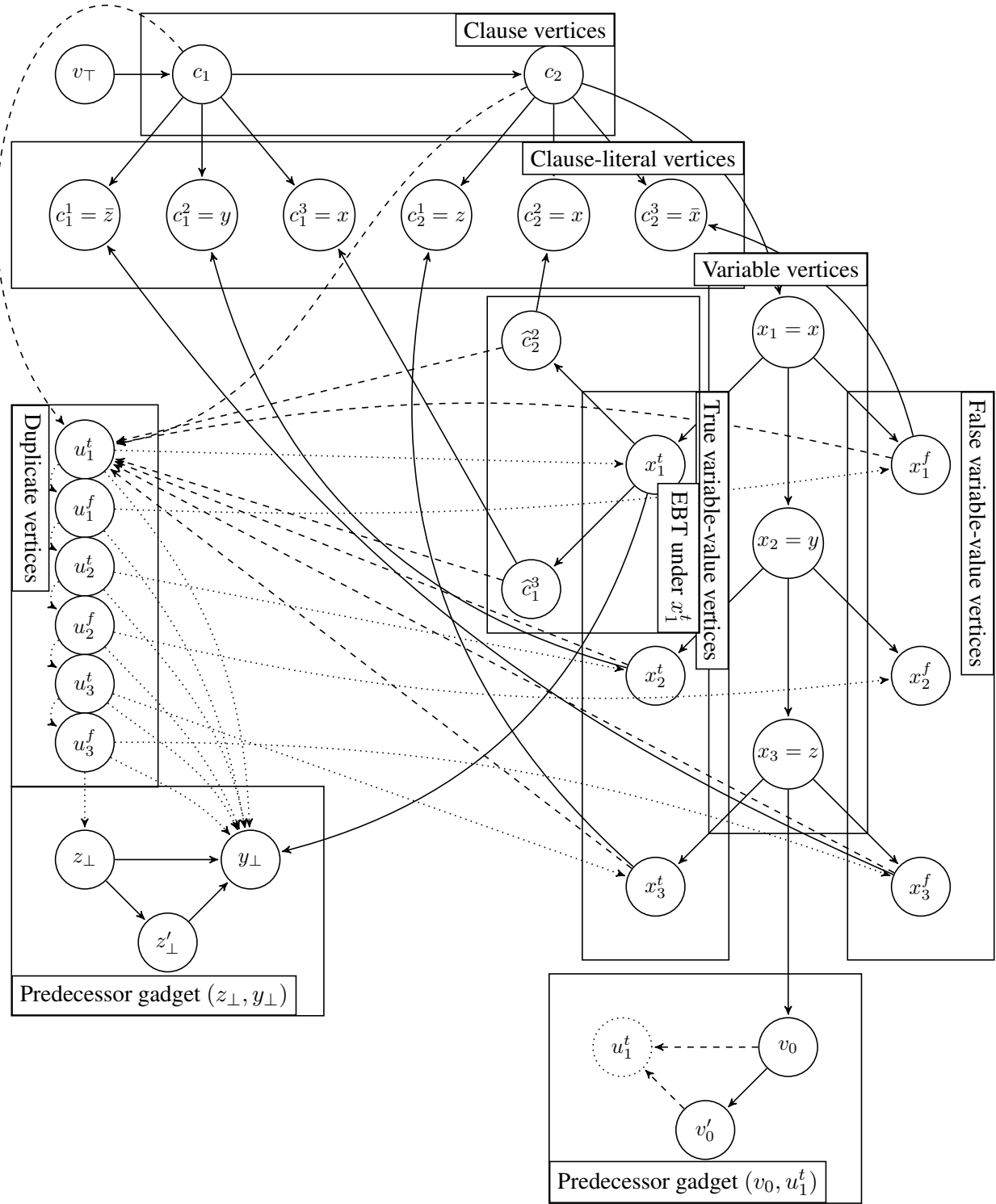[30] N. Vishnoi. The speed of evolution. In *SODA 2015*. ACM-SIAM, 2015.

Figure 1: The graph $G(\varphi)$ for $\varphi = (\bar{z} \vee y \vee x) \wedge (z \vee x \vee \bar{x})$. The edges to u_1^t are dashed and the edge from u_i^α for all $1 \le i \le 3$ and $\alpha \in \{t, f\}$ are dotted, for readability. Also, the vertex u_1^t is included twice to make it clear that it is in a predecessor gadget.
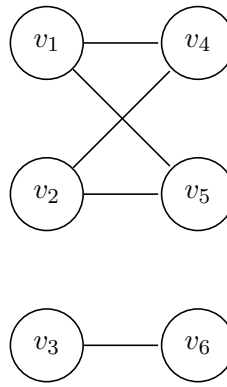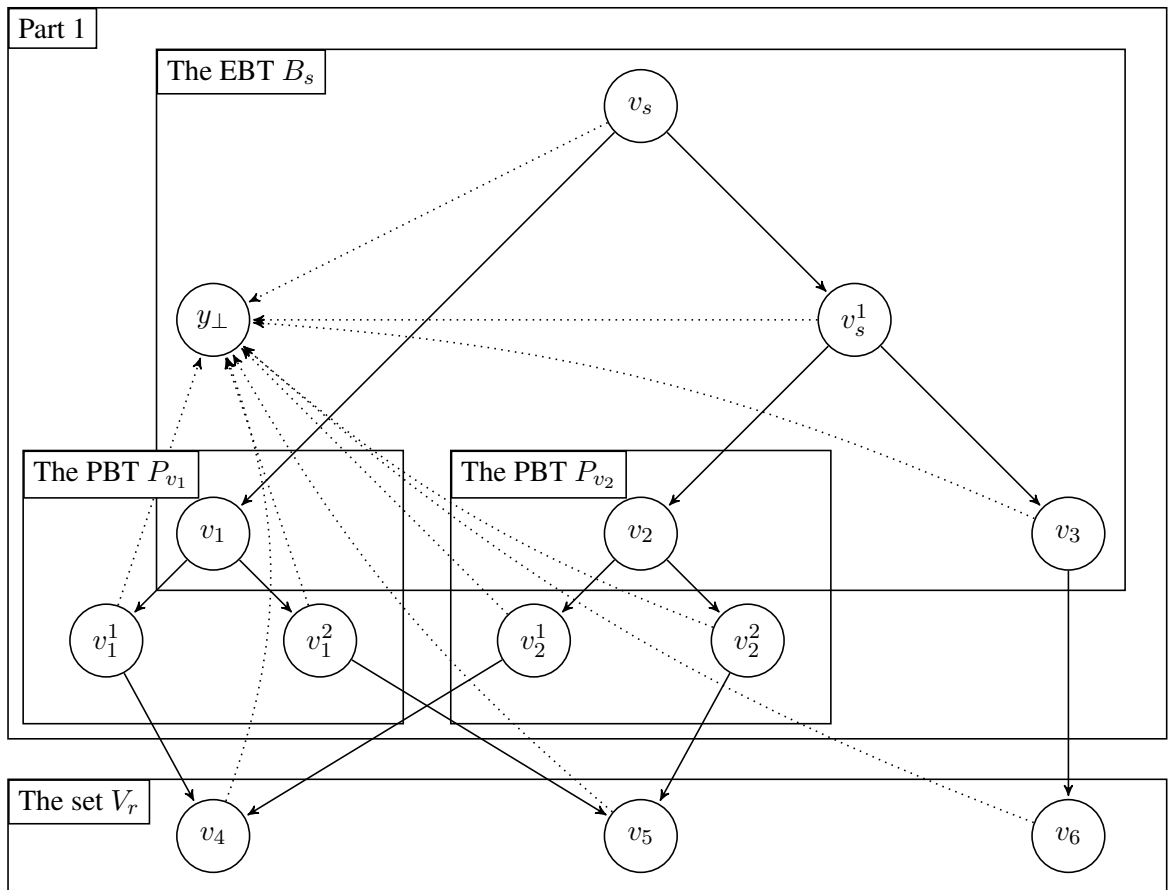
Figure 2: The graph G.

Figure 3: Part 1 and the edges related to V_r of the graph $\text{Red}(G)$. The edges to $y_\perp$ are dotted for readability.
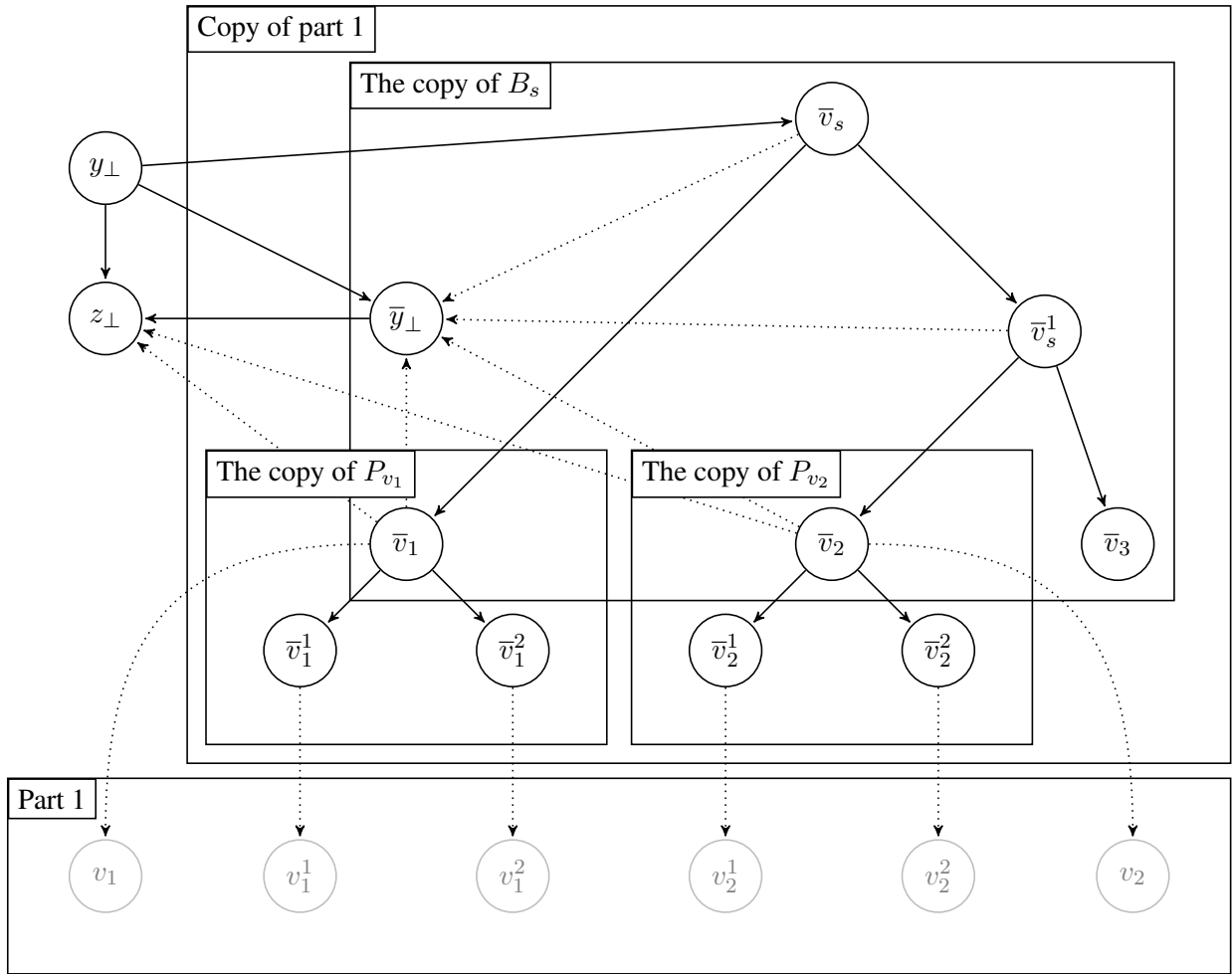
Figure 4: The copy of part 1 of the graph Red(G). Most edges to $\overline{y}_\perp$ and to $z_\perp$ are dotted for readability.
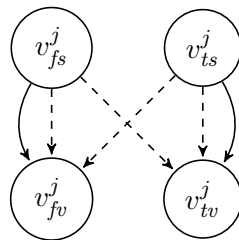

Figure 5: Boolean-value gadget: Dashed edges are in E_I and non-dashed are in E_R.