The Complexity of Reachability in Randomized Sabotage Games[*]

Dominik Klein, Frank G. Radmacher, and Wolfgang Thomas

Lehrstuhl für Informatik 7, RWTH Aachen University, Germany

```
dominik.klein@rwth-aachen.de
radmacher@automata.rwth-aachen.de
thomas@automata.rwth-aachen.de
```

Abstract. We analyze a model of fault-tolerant systems in a probabilistic setting. The model has been introduced under the name of "sabotage games". A reachability problem over graphs is considered, where a "Runner" starts from a vertex u and seeks to reach some vertex in a target set F while, after each move, the adversary "Blocker" deletes one edge. Extending work by Löding and Rohde (who showed PSPACE-completeness of this reachability problem), we consider the randomized case (a "game against nature") in which the deleted edges are chosen at random, each existing edge with the same probability. In this much weaker model, we show that, for any probability p and $\varepsilon > 0$, the following problem is again PSPACE-complete: Given a game graph with u and F and a probability p' in the interval $[p - \varepsilon, p + \varepsilon]$, is there a strategy for Runner to reach F with probability $\geq p'$? Our result extends the PSPACE-completeness of Papadimitriou's "dynamic graph reliability"; there, the probabilities of edge failures may depend both on the edge and on the current position of Runner.

Key words: games, reachability, probabilistic systems, fault-tolerant systems

1 Introduction

The subject of this paper is a model of fault-tolerant computation in which a reachability objective over a network is confronted with the failure of connections (edges). It is well known that adding dynamics to originally static models makes their analysis much harder – in our case, these dynamics are generated by the feature of vanishing edges in graphs. We build on hardness results of Löding and Rohde that are explained in more detail below. In the present paper we combine the aspect of dynamics with probability assumptions, which makes the model "coarser" or "softer". We show that, even in the probabilistic framework, the hardness phenomena of the standard dynamic model are still valid. Technically

speaking, we show that the results of Löding and Rohde are preserved in the more general randomized framework.

Specifically, we consider a two player game based on the model of (discrete) "sabotage games" suggested by van Benthem (cf. [12]). These games are played on graphs which edges may be multi-edges. A player, called "Runner", moves along edges and wants to reach a vertex in a set F from a given initial vertex u. After each move of Runner, the adversary, called "Blocker", may delete an edge; and in this way Runner and Blocker move in alternation. The algorithmic problem of "solving this game" asks for a winning strategy for Runner that enables him to reach a vertex in F against any choice of Blocker in deleting edges. The theory of these games was developed by Löding and Rohde; see [5,6,4,10,11]. Also, other winning conditions (more general than reachability) were considered; see [5,11,12].

In the present paper, we modify the sabotage games in a way that corresponds to a more realistic modeling: The second player Blocker is replaced by random fault. In each turn an edge (which may be a multi-edge) between two nodes is hit – each existing (multi-)edge with the same probability – and its multiplicity is reduced (resp. deleted in case of a single edge). So, in this approach, the player Runner is not faced with a Blocker, but rather has to play against "nature" [7]. There are several scenarios that motivate this model, e.g. the "Traveling Researcher Problem" as formulated by van Benthem [12], or the analysis of routing problems in networks where routing is subject to change, reacting to failures of connections. In such cases, it is rarely realistic to assume that there is an omniscient adversary that manipulates the world. The faults in natural scenarios are usually better modeled as random events. In our work, we use the term "randomized sabotage game" to emphasize our starting point, the sabotage games; but one might prefer to speak of reachability games against nature.

Our studies extend previous results in two ways: On the one hand, this paper extends the classical framework of sabotage games in which Löding and Rohde showed the PSpace-completeness of solving sabotage games with reachability winning conditions [5]. The natural question arises whether this result can be transferred when replacing the adversary player Blocker by arbitrary fault. On the other hand, our work is closely related to a similar question which was studied by Papadimitriou in his work on "games against nature" [7]. He considered the problem of "dynamic graph reliability" (DGR), where each edge e fails with a probability $p(e, v)$ after each turn, i.e. the probability depends on both the edge e and the current position v of Runner. Papadimitriou's game model against nature is rather strong, since for all edges the probabilities of deletion can be adjusted after each turn; his proof for the PSpace-hardness of DGR heavily depends on these adjustments. In fact, all problems that are considered in [7,8] as "games against nature" allow a precise adjustment of the probability for arbitrary large games, so that a reduction from the PSpace-complete problem $SSAT$ [7,3] is possible (which is a stochastic version of SAT, with radmoized quantifiers instead of universal quantifiers). However, it should be noted that randomized sabotage

games are not a special case of DGR, since, in randomized sabotage games, exactly one edge is deleted in each turn. In this paper, we pursue the question of whether Papadimitriou's result can be extended to a game model with a uniform probability distribution (e.g. in each turn, one of the n edges is deleted with probability $p = \frac{1}{n}$).

Our main result says that, in randomized sabotage games with a uniform distribution of failures where exactly one edge is deleted per turn, for any $p \in [0,1]$ and $\varepsilon > 0$ the following problem is PSPACE-complete: Given a game arena with origin u, a distinguished set F, and a probability p' in the interval $[p - \varepsilon, p + \varepsilon]$, does Runner have a strategy to reach F from u with probability $\geq p'$?

The remainder of this paper is structured as follows. In Section 2 we introduce the basic notions of randomized sabotage games. Section 3 is concerned with the PSPACE-hardness of the reachability version of randomized sabotage games. Here we start from the construction of Löding and Rohde [5] on PSPACE-hardness for the original sabotage game. For infinitely many probabilities $p_{k,n} \in [0,1]$ we reduce the PSPACE-complete problem of *Quantified Boolean Formulae* (QBF) (see [8], problem "QSAT") to the question of whether, given a randomized sabotage game with u and F, the goal set F is reachable with a probability of $p_{k,n}$. In order to complete the proof of our main result, we show in Section 4 that the set of the probabilities $p_{k,n}$ is dense in the interval $[0,1]$, and that the parameters k and n can be computed efficiently such that $p_{k,n} \in [p - \varepsilon, p + \varepsilon]$. In Section 5 we address perspectives which are treated in ongoing work.

2 The Randomized Sabotage Game

A sabotage game is played on a graph (V, E), where V is a set of vertices. The vertices are connected by a set of edges, given by $E \subseteq V \times V$. We will assume undirected graphs from now on, i.e. $(u, v) \in E \Rightarrow (v, u) \in E$; however, the ideas presented here also work for directed graphs in the same way. It should also be noted that, while in the "classical" notion of sabotage games multi-edges are allowed, we will restrict ourselves to single edges only. Clearly, the hardness result presented here also holds for the case of multi-edges (and also, the problem still belongs to PSPACE).

A *randomized sabotage game* is a tuple $\mathcal{G} = (G, v_{\text{in}})$ with a graph $G = (V, E_{\text{in}})$ and the initial vertex $v_{\text{in}} \in V$. A position of the game is a tuple (v_i, E_i). The initial position is $(v_{\text{in}}, E_{\text{in}})$. In each turn of the game, the player – called Runner – chooses an outgoing edge (v_n, v_{n+1}) in vertex v_n of position (v_n, E_n) and moves to vertex v_{n+1}. Then, a dice with $|E_n|$ sides is thrown and the chosen edge e is removed from E_n. We define $E_{n+1} := E_n \setminus \{e\}$. After this turn, the new position of the game is (v_{n+1}, E_{n+1}); we say that Runner has *reached* the vertex v_{n+1}.

Clearly, since edges are only deleted and not added, each play and the number of positions are finite. We only consider *reachability* as winning condition in this paper: For a randomized sabotage game $\mathcal{G} = ((V, E_{\text{in}}), v_{\text{in}})$ with a set of *final vertices* $F \subseteq V$, Runner wins a play iff he reaches a final vertex $v \in F$.

3

For the probabilistic analysis, we build up the game tree $t_\mathcal{G}$ for any randomized sabotage game $\mathcal{G}$. It is convenient to introduce tree nodes also for the intermediate positions that result from nature's moves, i.e. from edge deletions (in the following nature's positions are marked with an overline). The root of the game tree is $(v_{\text{in}}, E_{\text{in}})$, where Runner starts to move. From a position (v, E) with $v \notin F$ where Runner moves, the successor nodes are all positions $\overline{(v', E)}$ with $(v, v') \in E$ (a position (v, E), with $v \in F$ or $(v, v') \notin E$ for all v', is a leaf). Now, the successors of $\overline{(v', E)}$ are the positions (v', E') where E' results from E by an edge deletion.

To each node of Runner we associate the probability for Runner to win the subgame starting in this node. This probability is computed inductively in the obvious way, starting with 1 and 0 at a leaf (v, E) depending on whether $v \in F$ or not. For an inner node s of Runner with successors s_i of Nature, suppose that s_i has k successors $s_{i1}, \ldots, s_{ik}$ (where again Runner moves) which have, by induction, probabilities p_{ij} for Runner to win. We associate to each s_i the probability $p_i := \frac{1}{k} \sum_j p_{ij}$; then we pick the maximal p_i that occurs and associate it to s (a node with this maximal probability will be chosen by Runner). We say that Runner wins with probability p if the root of the game tree has a probability p.

Let p be an arbitrary number in $[0, 1]$. The Problem *Randomized Reachability Game for probability p* is the following:

> Given a randomized sabotage game $\mathcal{G} = ((V, E_{\text{in}}), v_{\text{in}})$ and a designated set $F \subseteq V$, does Runner win this game with probability $\geq p$?

Löding and Rohde have already shown that solving classical sabotage games with reachability winning condition is PSPACE-complete [5]. So, the randomized sabotage game problem for probability $p = 1$ is PSPACE-hard. On the other hand, the problem of whether Runner wins a randomized sabotage game with a probability $p > 0$ is decidable in linear time, because Runner wins with a probability > 0 iff there is a path from the initial to a final vertex.

Our main result says that the problem remains PSPACE-hard if we restrict the probability to any interval: For any fixed $p \in [0, 1]$ and $\varepsilon > 0$, the randomized reachability game problem for a probability p' which may vary in the interval $[p - \varepsilon, p + \varepsilon]$ is PSPACE-complete. More precisely:

Theorem 2.1. *For each fixed $p \in [0, 1]$ and $\varepsilon > 0$, the following is* PSPACE-*complete: Given a randomized sabotage game $\mathcal{G}$ with goal set F and a probability $p' \in [p - \varepsilon, p + \varepsilon]$, does Runner win $\mathcal{G}$ with probability $\geq p'$?*

For the proof we use a parametrized reduction of the problem *Quantified Boolean Formulae* (QBF): For each QBF-sentence φ, we construct a family of instances $\mathcal{G}_{\varphi,k,n}$ and $p_{k,n}$ such that, for each k and n, the sentence φ is true iff, over $\mathcal{G}_{\varphi,k,n}$ with u and F, Runner wins with probability $\geq p_{k,n}$. Furthermore, we guarantee that, given $p, \varepsilon > 0$, the probability $p_{k,n}$ can be chosen in $[p - \varepsilon, p + \varepsilon]$ for suitable k and n, and that this choice can be made in polynomial time. The proof that the problem belongs to PSPACE is easy, using standard techniques

4

from the analysis of finite games. The idea is to generate the game tree in a depth-first manner, with a storage for paths (and some auxiliary information); see [2,8]. In the remainder we treat only the hardness proof. The next section provides the indicated reduction, and in the subsequent section, we address the question of the distribution and efficient computation of the probabilities $p_{k,n}$.

3 PSPACE-Hardness of the Reachability Game

In order to prove the PSPACE-hardness, we use a parametrized reduction from the problem Quantified Boolean Formulae (QBF), which is known to be PSPACE-complete (cf. [8], problem "QSAT"). The reduction uses parts of the construction of Löding and Rohde [5]. The basic strategy is to construct an arena in such a way that, in a first part of the game, Runner can select the assignments for existential quantified variables of the formula, and that he is forced to choose certain assignments for the universal quantified variables. Then, this assignment is verified in a second part.

Formally, an instance of the problem QBF is a special quantified boolean formula, more precisely: Given a boolean expression ϑ in conjunctive normal form over boolean variables $x_1, \ldots, x_m$, is $\exists x_1 \forall x_2 \ldots Q_m x_m \vartheta$ true? Without loss of generality, one requires the formula to start with an existential quantifier. If m is even, $Q_m = \forall$; otherwise $Q_m = \exists$. For each instance φ of QBF, we construct a game arena $\mathcal{G}_{\varphi,k,n}$ and a rational number $p_{k,n}$ such that φ is true iff Runner has a strategy to reach a final vertex in $\mathcal{G}_{\varphi,k,n}$ exactly with probability $p_{k,n}$. Thereby the parameter k is an arbitrary natural number ≥ 2, and the parameter $n \in \mathbb{N}$ essentially has to be greater than the size of φ, i.e. $n \geq c \cdot |\varphi|$ for some constant c. If Runner plays suboptimally or if the formula is false, the maximum probability of winning is strictly lower than $p_{k,n}$; so the reduction meets the formulation of our game problem.

The arena consists of four types of subparts or "gadgets": The parametrization, existential, universal, and verification gadgets. In the parametrization gadget, one can, by adding or removing edges, adjust the probability $p_{k,n}$.

The outline of the proof is the following: We first introduce a construction to simulate a kind of multi-edge; this is convenient for a feasible analysis of the probabilistic outcome in a framework where only single edges are allowed. Then, we briefly recall the construction for the existential, universal, and verification gadgets [5], and adapt the argument to meet our model of a play against nature. In a second step, we introduce the parametrization gadget to adjust the probability $p_{k,n}$. Finally, we use this construction to prove our main result (Theorem 2.1).

3.1 The l-Edge Construction

In the following proofs, it will be necessary to link two vertices u and v in such a way that the connection is rather strong, i.e. there needs to be a number of several subsequent deletions until Runner is no longer able to move from u to v or

vice versa. This is achieved by the construction shown in Figure 1, which we will call an "l-edge" from now on ($l \geq 1$). The circled vertex is a goal belonging to the set F of final vertices. Here, if after $|l|$ deletions all the $|l|$ edges between u and the middle vertices are deleted, Runner is disconnected from v. If $|l - 1|$ or less edges have been deleted anywhere in the game graph, there is at least one path from u over some middle vertex to v and an edge between that middle vertex and the final vertex. Then, Runner can move from u (resp. v) to that middle vertex and is able to reach v (resp. u) in the next step, or he wins immediately by moving to the (circled) final vertex.
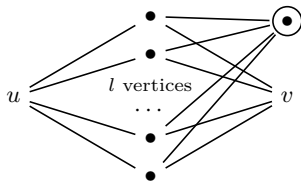

Fig. 1. An l-edge from u to v.

Lemma 3.1. *Given a randomized sabotage game with an l-edge between two nodes u and v, Runner can guarantee to reach v via this l-edge iff he can guarantee to reach u within $l - 1$ moves.*

For a sufficiently high l, depending on the structure of the game-arena, it is clear that Runner cannot be hindered from winning by edge deletions in an l-edge; such l-edges will be called "∞-edges" to indicate that destruction of the connection can be considered impossible. (For classical sabotage games, l can be bounded by the total number of vertices in the game arena, because if Runner has a winning strategy in a classical sabotage game, he has also a winning strategy without visiting any vertex twice [5]. For the constructions in this paper where ∞-edges are used, it will be sufficient to consider ∞-edges as 4-edges.)

Remark 3.2. In order to sharpen the hardness result of this paper to randomized sabotage games with a *unique* goal, one may intend to merge final vertices to one vertex. But this is not always possible: Consider an l-edge to a final vertex v. Since we do not consider graphs with multi-edges, v cannot be merged with the other final vertex from the l-edge construction without breaking Lemma 3.1. For this reason, in this paper, PSPACE-hardness is shown only for randomized sabotage games with at least two final vertices.

3.2   Existential, Universal, and Verification Gadgets

In this section, we briefly recall constructions from [5] to have a self-contained exposition. We introduce the existential and universal gadgets (applied according to the quantifier structure of the given formula), and the verification gadget (corresponding to its quantifier-free kernel).
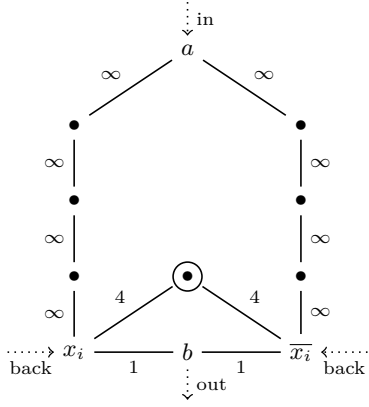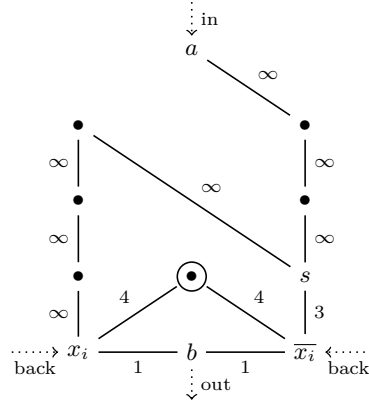
Fig. 2. The ∃-gadget for x_i with i odd. **Fig. 3.** The ∀-gadget for x_i with i even.

The Existential Gadget: Intuitively, the existential component allows Runner to set an existential-quantified variable to true or false. The gadget is depicted in Figure 2. The node a is the input vertex for this gadget, and x_i (resp. $\overline{x_i}$) is the variable vertex of this component which Runner intends to visit if he sets x_i to false (resp. true). The vertex b is the exit node of this gadget; it coincides with the in-vertex of the next gadget (i.e. the universal gadget for x_{i+1}, or the verification gadget if x_i is the last quantified variable). The "back"-edges from x_i and $\overline{x_i}$ lead directly to the last gadget of the construction, the verification gadget. Later, Runner possibly move back via these edges to verify his assignment of the variable x_i. (We will see later that taking these edges as a shortcut, starting from the existential gadget, directly to the verification gadget, is useless for Runner.)

Of course, Runner has a very high probability of winning the game within the existential gadget (especially in an l-edge construction for an ∞-edge). But we are only interested in the worst-case scenario, where edges are deleted in the following precise manner:

When it is Runner's turn and he is currently residing in vertex a, he will move either left or right and can reach x_i (resp. $\overline{x_i}$) in four turns. When Runner moves towards x_i (resp. $\overline{x_i}$), the 4-edge from x_i (resp. $\overline{x_i}$) to the final vertex may be subsequently subject to deletion so that Runner ends up at node x_i (resp. $\overline{x_i}$) with no connection to the final vertex left. If Runner then moves towards $\overline{x_i}$ (resp. x_i) and the edge between b and $\overline{x_i}$ (resp. x_i) is deleted, he is forced to exit the gadget via b and move onwards. The 4-edge from $\overline{x_i}$ (resp. x_i) to the final vertex remains untouched so far. If Runner is later forced to move back to x_i or $\overline{x_i}$ from the verification gadget, he can only guarantee a win in one of these vertices.

The Universal Gadget: In the universal component a truth value for the all-quantified variables is chosen arbitrarily, but this choice can be considered to

7

Runner's disadvantage in the worst-case. Runner can be forced to move in one or the other direction and has to set x_i to true or false, respectively. The gadget is depicted in Figure 3. A path through this gadget starts in node a and is intended to exit via node b, which coincides with the in-vertex of the next gadget (i.e. the existential gadget for x_{i+1}, or the verification gadget if x_i is the last quantified variable). Again, only the worst cases are important for now; Runner is able to win the game immediately in all other cases. Clearly, Runner is going to move in the direction of vertex s. There are two interesting scenarios which may happen with a probability > 0:

In the first scenario, the 3-edge to $\overline{x_i}$ is deleted completely. Then, Runner can only guarantee to leave the gadget at b via x_i (but no visit of $\overline{x_i}$ or the (circled) final vertex), because the 4-edge from x_i to the final vertex and the 1-edge to $\overline{x_i}$ may be deleted successively. In this case, the 4-edge between $\overline{x_i}$ and the final vertex remains untouched.

In the second case, only the 4-edge from $\overline{x_i}$ to the final vertex is subject to deletion. At s, Runner is intended to move downward to $\overline{x_i}$ and leave the gadget at b. Thereby the 4-edge between $\overline{x_i}$ and the final vertex (which was already reduced to a 1-edge) is deleted completely, and after this, the 1-edge to x_i is deleted. Consequently, the 4-edge from x_i to the final vertex is untouched. If Runner "misbehaves" in the sense that he moves from s to the left, it may happen that the final vertex becomes completely disconnected from both x_i and $\overline{x_i}$; in this case, Runner cannot win in this vertex if he is forced to move back to x_i or $\overline{x_i}$ from the verification gadget.

The Verification Gadget: The verification gadget is constructed in such a way that, when Runner arrives here, he can only force a win if the assignment for the variables which has been chosen beforehand satisfies the quantifier-free kernel of the formula.
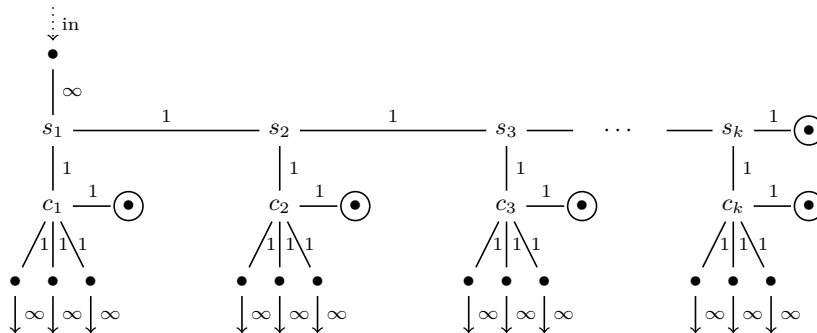

Fig. 4. The verification gadget for a formula with k clauses.

The verification gadget for a QBF-formula with k clauses $C_1, \ldots, C_k$ is depicted in Figure 4. Its in-vertex coincides with exit vertex b of the last existential/universal gadget. For a clause $C_i = (\neg)x_{i_1} \vee (\neg)x_{i_2} \vee (\neg)x_{i_3}$, there are three paths, each from c_i via a single edge and an ∞-edge (the literal edge L_{ij}) back to the variable vertex x_{i_j} in the corresponding gadgets. Again, a look at the interesting scenarios is important:

Assume that Runner has chosen the appropriate assignments of the variables for a satisfiable formula. He reaches the first selection vertex s_1 via the ∞-edge from the last existential/universal gadget. If Runner is in s_i and the edge to c_i is deleted, he has to proceed to s_{i+1}. Now, assume that the edge between s_i and s_{i+1} is removed. Then, Runner is forced to move towards c_i. If the quantifier-free kernel of the QBF-formula is satisfied with the chosen assignment of Runner, then there is at least one literal that satisfies the clause C_i. Runner chooses to move alongside the corresponding literal edge L_{ij} back to x_{i_j}, into the gadget where he has chosen the assignment (and wins there by moving to the final vertex). In any other case Runner is able to win immediately by moving via s_k or via s_i, c_i to the (circled) final vertex. Note that he only has a chance of *always* winning if his chosen assignment actually fulfills the quantifier-free kernel of the formula.

If he did not choose a correct assignment or if the formula is not satisfiable, there is at least one clause that falsifies the QBF-formula, say clause c_i. If he is forced to go to c_i, there may be no path that leads him back to the final vertex of an existential/universal gadget.

As a side remark, one should note that it is never optimal for Runner to take a "back"-edge (i.e. a literal edge L_{ij}) as a shortcut, moving directly from some x_i (resp. $\overline{x_i}$) of an existential/universal gadget to the verification gadget. In this case, the 1-edge connecting c_i and the ∞-edge from x_i (resp. $\overline{x_i}$) to the verification gadget may be destroyed. Then, Runner has to move back and loses his chance of always winning the game.

We introduced so far the construction from [5] which suffices to show PSPACE-hardness for $p = 1$: Using the gadgets above, Runner does have a winning strategy iff the given formula is true. For a QBF-formula φ, we call the game arena of this construction $\mathcal{G}_\varphi$.

3.3 The Parametrization Gadget

The parametrization gadget is the initial part of the game arena that is constructed; it is used to "adjust" the overall winning probability of Runner. Runner starts from the initial vertex in this gadget. For each $k \geq 1$, we define the parametrization gadget $\mathcal{H}_k$; it is depicted in Figure 5.

We reduce the question of whether the QBF-formula φ is true to the reachability problem over certain game arenas that result from combining $\mathcal{H}_k$ with $\mathcal{G}_\varphi$ as a graph $\mathcal{H}_k \circ \mathcal{G}_\varphi$: The out-vertex b in $\mathcal{H}_k$ is identified with the in-vertex a of the first existential gadget of $\mathcal{G}_\varphi$. Assume $\mathcal{G}_\varphi$ has n_0 edges. We get modifications of $\mathcal{G}_\varphi$ with any number $n \geq n_0$ of edges by adding artificial extra edges (without changing the behavior in the discussed worst case); for instance, this can be
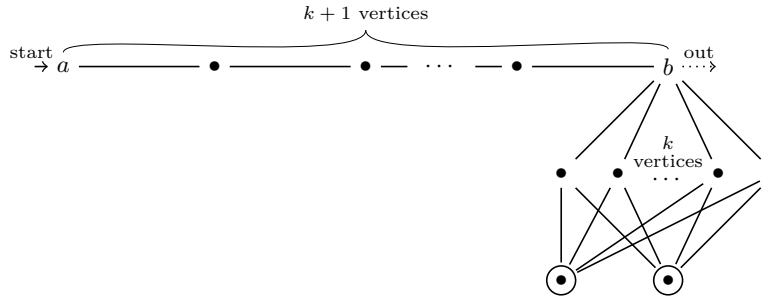
Fig. 5. The parametrization gadget $\mathcal{H}_k$.

achieved by adding a path with a dead end. We call this game arena $\mathcal{G}_\varphi^n$. In the sequel, we work with

$$\mathcal{G}_{\varphi,k,n} := \mathcal{H}_k \circ \mathcal{G}_\varphi^n .$$

Since $\mathcal{H}_k$ has $4k$ edges, the overall number of edges in our game arena $\mathcal{G}_{\varphi,k,n}$ is $4k + n$. Let

$$p_{k,n} := \text{probability for Runner to traverse the}$$
$$\text{parametrization gadget } \mathcal{H}_k \text{ of } \mathcal{G}_{\varphi,k,n} .$$

Lemma 3.3. *Runner wins the randomized reachability game over $\mathcal{G}_{\varphi,k,n}$ for probability $p_{k,n}$ iff the QBF-formula φ is true.*

Proof. First note the following: If Runner resides at vertex b, and in $\mathcal{H}_k$ exactly the k edges below vertex b have been deleted so far, then Runner wins with probability 1 iff φ is true (this follows immediately from the classical construction by Löding and Rohde [5]).

Now assume that φ is true. In the parametrization gadget Runner starts at node a and obviously moves towards b. Clearly, if any edge between his current position and b is deleted, he loses the game immediately. However, if he succeeds in getting to b, he will always win the game: First, assume the case that, in his k steps towards b, only edges in $\mathcal{H}_k$ were subject to deletion. Then, Runner always wins by moving in the first existential gadget and traversing $\mathcal{G}_\varphi^n$, as mentioned before. If we assume the other case that at least one of the k deletions took place outside of $\mathcal{H}_k$, there is at least one edge leading from b downward to some middle node, say b', and there are at least two edges leading from b' to the two final vertices in the parametrization gadget. Then, Runner wins by moving from b downward to b' and then, depending on the next deletion, by moving to one of the two final vertices. In both cases, Runner wins over $\mathcal{G}_{\varphi,k,n}$ exactly with the probability $p_{k,n}$ of traversing the parametrization gadget $\mathcal{H}_k$ from node a to node b.

Now, assume that φ is false, and that only the k edges below vertex b in $\mathcal{H}_k$ are subject to deletion while Runner moves towards b (this may happen with a probability > 0). Then, Runner's only chance to win from b is by moving towards

10

some final vertex in $\mathcal{G}_\varphi^n$. Since φ is false, his winning probability in b is strictly smaller than 1, and hence his overall winning probability for $\mathcal{G}_{\varphi,k,n}$ is strictly smaller than $p_{k,n}$. □

The computation of the probabilities $p_{k,n}$ only depends on the parametrization gadget $\mathcal{H}_k$ and n: Clearly $p_{1,n} = 1$. For $k \geq 2$, the winning probability is obtained from the probability of not failing in the first step, multiplied by the probability of not failing in the second step, etc., until the probability of not failing in the $(k-2)$-th step, where Runner tries to get to b within one step. After the first step, Runner has still to cross $k-1$ edges; neither of them may be deleted. Overall, there are $4k+n$ edges, so Runner does not lose if any of the other $4k + n - (k-1)$ edges is deleted. In the last step before reaching b, $k-2$ edges have been deleted, so $4k + n - (k-2)$ edges are left in the game. If any other than the edge between Runner's current position and b is deleted, Runner is able to reach b. Generally, in the i-th step there are $4k + n - (i-1)$ edges left; and in order for Runner to still be able to reach b, one of the $3k + n + 1$ edges that are not between Runner's current position and b has to be deleted. Altogether,

$$p_{k,n} = \frac{3k+n+1}{4k+n} \cdots \frac{3k+n+1}{4k+n-(k-2)} = \prod_{i=0}^{k-2} \frac{3k+n+1}{4k+n-i} \ .$$

We can summarize these observations in the following theorem:

Theorem 3.4. *Given a QBF-formula φ so that $\mathcal{G}_\varphi$ has n_0 edges, for all $k, n \in \mathbb{N}$ with $k \geq 2$ and $n \geq n_0$ the following holds: Runner wins the randomized reachability game over $\mathcal{G}_{\varphi,k,n}$ for probability $p_{k,n} = \prod_{i=0}^{k-2} \frac{3k+n+1}{4k+n-i}$ iff the QBF-formula φ is true.*

In order to use this theorem for a reduction to the randomized sabotage game, we need to show that the game arena can be constructed in polynomial time. In the following Lemma, we show that the size of the constructed game $\mathcal{G}_{\varphi,k,n}$ is linear in the size of the inputs φ, k, and n:

Lemma 3.5. *The size of $\mathcal{G}_\varphi$ is linear in $|\varphi|$, and the size of $\mathcal{H}_k$ is linear in k.*

Proof. For the first part, it is sufficient to realize that the size of each gadget can be bounded by a constant. Since the number of gadgets is linear in the size of φ, the number of vertices and edges of $\mathcal{G}_\varphi$ is linear in $|\varphi|$. The only problem might be the ∞-edges; but by detailed observation, we see that each ∞-edge can be replaced by a 4-edge, and the construction still works in the same way.

For the second part, it suffices to note that $\mathcal{H}_k$ has $2k + 3$ vertices and $4k$ edges. □

Now, a preliminary result can be formulated in the following form:

Corollary 3.6. *For all $k \geq 2$, the following problem is* PSpace*-hard: Given a randomized sabotage game with n edges, does Runner win with a probability $\geq p_{k,n}$?*

3.4 Towards the PSPACE-Hardness for Arbitrary Probabilities

We already have a reduction of QBF to randomized sabotage games with a varying probability $p_{k,n}$ (which depends on the given game graph). By a closer look at the term $p_{k,n}$ we see that the probability $p_{k,n}$ can be adjusted arbitrary close to 0 and arbitrary close to 1; more precisely: For a fixed k, we have $\lim_{n\to\infty} p_{k,n} = 1$; and for a fixed n, we have $\lim_{k\to\infty} p_{k,n} = 0$. We will show a stronger result, namely that the probabilities $p_{k,n}$ form a dense set in the interval $[0,1]$, and that k and n can be computed efficiently such that $p_{k,n}$ is in a given interval. More precisely, we shall show the following:

Theorem 3.7. *The set of probabilities $\{p_{k,n} \mid k,n \in \mathbb{N},\ k \geq 2\}$ is dense in the interval $[0,1]$. Moreover, given $n_0 \in \mathbb{N}$, $p \in [0,1]$, and an $\varepsilon > 0$, there exist $k,n \in \mathbb{N}$ with $k \geq 2$ and $n \geq n_0$ such that $p_{k,n} \in [p-\varepsilon, p+\varepsilon]$; the computation of such k, n, and $p_{k,n}$ is polynomial in the* numerical values *of n_0, $\frac{1}{p}$, and $\frac{1}{\varepsilon}$.*

The proof of this theorem is the subject of Section 4.

Note that Theorem 3.7 provides a pseudo-polynomial algorithm, since the computation is only polynomial in the *numerical values* of n_0, $\frac{1}{p}$, and $\frac{1}{\varepsilon}$ (and not in their *lengths*, which are logarithmic in the numerical values). For our needs – i.e. a polynomial time reduction to prove Theorem 2.1 – this is no restriction: The parameter n_0 corresponds to the number of edges in the input game (which has already a polynomial representation), and p and ε are fixed values (i.e. formally they do not belong to the problem instance).

Now, we prove our main result:

Proof (of Theorem 2.1). For arbitrary p and ε, we give a reduction from QBF to the randomized sabotage game problem where only probabilities in the interval $[p-\varepsilon, p+\varepsilon]$ are allowed. Note that p and ε do not belong to the problem instance; so they are considered constant in the following.

Given a QBF-formula φ, we need to compute a game $\mathcal{G}_{\varphi,k,n}$ and a $p_{k,n} \in [p-\varepsilon, p+\varepsilon]$ such that Runner wins $\mathcal{G}_{\varphi,k,n}$ with a probability $\geq p_{k,n}$ iff φ is true. Given φ, we first apply the construction of Section 3.2 to construct an equivalent sabotage game $\mathcal{G}_\varphi$. Let n_0 be the number of edges of $\mathcal{G}_\varphi$, which is linear in the size of φ according to Lemma 3.5. Then, we can compute $k \geq 2$, $n \geq n_0$, and $p_{k,n}$ according to Theorem 3.7. For a fixed ε, the computations are polynomial in n_0 and hence polynomial in $|\varphi|$. Now, we extend $\mathcal{G}_\varphi$ to an equivalent sabotage game with n edges, denoted $\mathcal{G}_\varphi^n$. This can be achieved by adding $n-n_0$ dummy-edges (e.g. we can add a path with a dead end). Thereafter, we construct the randomized sabotage game $\mathcal{G}_{\varphi,k,n}$ by combining the parametrization gadget $\mathcal{H}_k$ with the game arena $\mathcal{G}_\varphi^n$.

The claimed equivalence of φ to the stated randomized reachability game problem for probability $p_{k,n}$ holds due to Theorem 3.4. The requirement that $p_{k,n}$ is in the interval $[p-\varepsilon, p+\varepsilon]$ follows from Theorem 3.7. $\qquad\square$

4 On the Distribution and Computation of the Probabilities $p_{k,n}$

This section deals with the proof of Theorem 3.7: Given $n_0 \in \mathbb{N}$, $p \in [0,1]$, and an $\varepsilon > 0$, we can construct $k > 2$ and $n \geq n_0$ in polynomial time with respect to the numerical values of n_0, $\frac{1}{p}$, and $\frac{1}{\varepsilon}$, such that $p_{k,n}$ is in the interval $[p-\varepsilon, p+\varepsilon]$.

The idea is to first adjust the probability $p_{k,n}$ arbitrary close to 1, and then go with steps of length below any given $\varepsilon > 0$ arbitrary close to 0; so, we hit every ε-neighborhood in the interval $[0,1]$.

In order to adjust the probability $p_{k,n}$ arbitrary close to 1, we first choose $k = 2$ and a sufficiently high $n \geq n_0$. (We can artificially increase n by adding a path with a dead end.) We will show that it suffices to choose $n := \max\{n_0, \lceil \frac{1}{\varepsilon} \rceil\}$. For this choice we obtain $p_{2,n} \geq 1 - \varepsilon$ $(\geq p - \varepsilon)$. Then, we decrease the probability by stepwise incrementing k by 1 (changing $\mathcal{H}_k$ to $\mathcal{H}_{k+1}$ and keeping n constant). It will turn out that (with the choice of n as above) the probability decreases by a value that is lower than $\frac{1}{4k+n+4}$ $(\leq \varepsilon)$. Iterating this, the values converge to 0, and we hit the interval $[p-\varepsilon, p+\varepsilon]$. Hence, the set of probabilities $\{p_{k,n} \mid k, n \in \mathbb{N}, \ k \geq 2\}$ is dense in the interval $[0,1]$. Furthermore, we will show that it will be sufficient to increase k at most up to $8n$. For this choice, we obtain $p_{k,n} \leq \varepsilon$ $(\leq p + \varepsilon)$.

For the complexity analysis, note the following: After each step, the algorithm has to check efficiently whether $p_{k,n} \in [p - \varepsilon, p + \varepsilon]$. The computation of the term $p_{k,n}$ is pseudo-polynomial in k, n, and the test for $p_{k,n} \leq p + \varepsilon$ is in addition polynomial in $\frac{1}{p}$ and $\frac{1}{\varepsilon}$. Since k and n are pseudo-linear in n_0 and $\frac{1}{\varepsilon}$, the whole procedure is pseudo-polynomial in n_0, $\frac{1}{p}$, and $\frac{1}{\varepsilon}$.

Four claims remain to be proved:

- The adjustment of $p_{k,n}$ arbitrary close to 1 with the proposed choice of n, i.e. given $\varepsilon > 0$, for $n \geq \frac{1}{\varepsilon}$ holds $p_{2,n} \geq 1 - \varepsilon$.
- The adjustment of $p_{k,n}$ arbitrary close to 0 with the proposed choice of k, i.e. given $\varepsilon > 0$ and $n \geq \frac{1}{\varepsilon}$, for $k \geq 8n$ holds $p_{k,n} \leq \varepsilon$.
- The estimation $p_{k,n} - p_{k+1,n} < \frac{1}{4k+n+4}$.
- The test for $p_{k,n} \in [p - \varepsilon, p + \varepsilon]$ is pseudo-polynomial in k, n, $\frac{1}{p}$ and $\frac{1}{\varepsilon}$.

These claims are shown in the rest of this section:

Lemma 4.1. *Given $\varepsilon > 0$, for $n \geq \lceil \frac{1}{\varepsilon} \rceil$ we have $p_{2,n} \geq 1 - \varepsilon$.*

Proof. Since $n \geq \lceil \frac{1}{\varepsilon} \rceil \geq \frac{1}{\varepsilon} \geq \frac{1}{\varepsilon} - 8$ for $\varepsilon > 0$, the result follows from

$$p_{2,n} = \frac{n+7}{n+8} \geq 1 - \varepsilon \iff n \geq \frac{1}{\varepsilon} - 8 \ .$$

$\square$

Lemma 4.2. *Given $\varepsilon > 0$ and $n \in \mathbb{N}$ with $n \geq \frac{1}{\varepsilon}$ and $n \geq 4$, for $k \geq 8n$ we have $p_{k,n} < \varepsilon$.*

Proof. First note that we have at least $n \geq 1$ and $k \geq 8$. Then

$$p_{k,n} = \prod_{i=0}^{k-2} \frac{3k+n+1}{4k+n-i} \leq \left(\frac{3k+n+1}{3.5k+n}\right)^{\frac{k}{2}} \leq \left(\frac{4k+1}{4.5k}\right)^{\frac{k}{2}} \leq \left(\frac{4.125k}{4.5k}\right)^{\frac{k}{2}}$$

$$= \left(\frac{11k}{12k}\right)^{\frac{k}{2}} = \left(\frac{11}{12}\right)^{\frac{k}{2}} \leq \left(\frac{11}{12}\right)^{4n} < \varepsilon \ .$$

The inequality $\left(\frac{11}{12}\right)^{4n} < \varepsilon$ remains to be shown. Since $\frac{1}{n} \leq \varepsilon$, it is sufficient to show that $\left(\frac{11}{12}\right)^{4n} < \frac{1}{n}$:

$$\left(\frac{11}{12}\right)^{4n} < \frac{1}{n} \iff n^{\frac{1}{4n}} < \frac{12}{11} \iff \sqrt[n]{n^{\frac{1}{4}}} \leq \sqrt{2^{\frac{1}{4}}} < \frac{12}{11} \ .$$

The inequality $\sqrt[n]{n^{\frac{1}{4}}} \leq \sqrt{2^{\frac{1}{4}}}$ is equivalent to $n^2 \leq 2^n$ and holds for all $n \geq 4$. $\square$

Lemma 4.3. *For $k, n \in \mathbb{N}$ with $k \geq 2$, we have $p_{k,n} - p_{k+1,n} < \frac{1}{4k+n+4}$.*

Proof. In this proof we use the substitution $m := 4k + n + 4$.

$$p_{k,n} - p_{k+1,n} = \prod_{i=0}^{k-2} \frac{3k+n+1}{4k+n-i} - \prod_{i=0}^{k-1} \frac{3k+n+4}{4k+n+4-i}$$

$$\leq \prod_{i=0}^{k-2} \frac{3k+n+4+1}{4k+n+4-i} - \prod_{i=0}^{k-1} \frac{3k+n+4}{4k+n+4-i} = \prod_{i=0}^{k-2} \frac{m-k+1}{m-i} - \prod_{i=0}^{k-1} \frac{m-k}{m-i}$$

$$= \prod_{i=0}^{k-1} \frac{m-k+1}{m-i} - \prod_{i=0}^{k-1} \frac{m-k}{m-i} = \frac{(m-k+1)^{k-1} - (m-k)^{k-1}}{\prod_{i=0}^{k-1} m-i} \ .$$

Now we can use the equation $a^l - b^l = (a-b)(a^{l-1} + a^{l-2}b + \cdots + ab^{l-2} + b^{l-1})$ for the estimation $(d+1)^{k-1} - d^{k-1} = (d+1)^{k-2} + (d+1)^{k-3}d + \cdots + (d+1)d^{k-3} + d^{k-2} \leq (k-1)(d+1)^{k-2}$. We obtain $p_{k,n} - p_{k+1,n}$

$$\leq \frac{(k-1)(m-k+1)^{k-2}}{\prod_{i=0}^{k-1} m-i} = \frac{k-1}{m(m-1)} \prod_{i=2}^{k-1} \frac{(m-k+1)}{m-i} \leq \frac{k-1}{m(m-1)} \ .$$

Since $m > k$ for all $k, n \in \mathbb{N}$, we obtain $p_{k,n} - p_{k+1,n} < \frac{1}{m} = \frac{1}{4k+n+4}$. $\square$

Lemma 4.4. *The computation of the term $p_{k,n}$ is pseudo-polynomial in k and n. The test for $p_{k,n} \leq p + \varepsilon$ is pseudo-polynomial in k and n, and polynomial in $\frac{1}{p}$ and $\frac{1}{\varepsilon}$.*

Proof. First, we rewrite $p_{k,n}$ in the form

$$\frac{(3k+n+1)^{k-1}}{\prod_{i=0}^{k-2} 4k+n-i} \ .$$

Now, we compute the numerator and the denominator separately. For the computation, we can switch to binary encoding. Each multiplication can be performed in polynomial time in the length of its binary encoding [13]. We need $k-2$ multiplications (for this reason, the algorithm is only pseudo-polynomial). The division and comparison of two rational numbers can be done in polynomial time with respect to the length of their binary representations [13]. So, the quotient $p_{k,n}$ can be computed in pseudo-polynomial time with respect to k and n, and the test to check whether $p_{k,n} \leq p+\varepsilon$ is in addition polynomial in $\frac{1}{p}$ and $\frac{1}{\varepsilon}$.

$\square$

5   Perspectives

We have introduced randomized sabotage games, and showed that the reachability problem for a probability which may vary in a fixed interval $[p-\varepsilon, p+\varepsilon]$ is PSPACE-complete. This is a small contribution to the emerging research on the analysis of dynamical networks with aspects of randomness. As concrete open issues, we mention the following problems:

1. In our proof, it seems difficult to adjust the probability *exactly* to a given probability p (in our formulation this is the case $\varepsilon = 0$). It remains open whether this can be achieved by a refinement of the construction.
2. In our proof, we used the reachability problem with a target set F containing at least two vertices (see Remark 3.2). One task is to extend the result to cover also the case of a singleton as target set (note that in the non-randomized case, the singleton reachability problem is PSPACE-hard only if one allows multi-edges [5]).
3. The proof of our model depends on the restriction that exactly one edge per turn is deleted (rather than possibly multiple edge deletion occurring subject to given probabilities). Sharpening the mentioned problem of "dynamic graph reliability" [7] to probabilities that are independent of Runner's position, we can study the model where in every turn each edge fails with a probability $p(e)$, or even with probability $\frac{1}{n}$ in the uniform case.
4. Extending the model with a mechanism of restoration is a challenging task (for instance, *reactive Kripke models* [1] and *backup parity games* [11] address this issue). In [9] we developed a theory of dynamic networks where Runner and Blocker are replaced by two players, *Constructor* and *Destructor*, that add resp. delete vertices/edges, and the problem of guaranteeing certain network properties (like connectivity) is addressed. We are presently integrating probabilistic features into this model, starting from the present paper. Another interesting direction of research is to include more general winning conditions in appropriate logics [2].

15

References

1. Dov M. Gabbay. Introducing reactive Kripke semantics and arc accessibility. In *Pillars of Computer Science, Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*, volume 4800 of *Lecture Notes in Computer Science*, pages 292–341. Springer, 2008.
2. Dominik Klein. Solving Randomized Sabotage Games for Navigation in Networks. Diploma thesis, RWTH Aachen, 2008.
3. Michael L. Littman, Stephen M. Majercik, and Toniann Pitassi. Stochastic boolean satisfiability. *Journal of Automated Reasoning*, 27(3):251–296, 2001.
4. Christof Löding and Philipp Rohde. Model checking and satisfiability for sabotage modal logic. In *Proceedings of FSTTCS*, volume 2914 of *Lecture Notes in Computer Science*, pages 302–313. Springer, 2003.
5. Christof Löding and Philipp Rohde. Solving the sabotage game is PSPACE-hard. Technical Report AIB-05-2003, RWTH Aachen, 2003.
6. Christof Löding and Philipp Rohde. Solving the sabotage game is PSPACE-hard. In *Proceedings of MFCS*, volume 2747 of *Lecture Notes in Computer Science*, pages 531–540. Springer, 2003.
7. Christos H. Papadimitriou. Games against nature. *Journal of Computer and System Sciences*, 31(2):288–301, 1985.
8. Christos H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.
9. Frank G. Radmacher and Wolfgang Thomas. A game theoretic approach to the analysis of dynamic networks. In *Proceedings of VerAS*, volume 200 (2) of *Electronic Notes in Theoretical Computer Science*, pages 21–37. Elsevier, 2008.
10. Philipp Rohde. Moving in a crumbling network: The balanced case. In *Proceedings of CSL*, volume 3210 of *Lecture Notes in Computer Science*, pages 310–324. Springer, 2004.
11. Philipp Rohde. *On Games and Logics over Dynamically Changing Structures*. PhD thesis, RWTH Aachen, 2005.
12. Johan van Benthem. An essay on sabotage and obstruction. In *Mechanizing Mathematical Reasoning, Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday*, volume 2605 of *Lecture Notes in Computer Science*, pages 268–276. Springer, 2005.
13. Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, Cambridge, UK, second edition, 2003.