Optimally Controlling a Random Population

Hugo Gimbert ✉
CNRS, LaBRI, Université de Bordeaux, France

Corto Mascle ✉
LaBRI, Université de Bordeaux, France and MPI-SWS, Kaiserslautern, Germany

Patrick Totzke ✉
University of Liverpool, UK

Abstract

The population control problem is a parameterized control problem where a population of agents has to be moved simultaneously into a target state. The decision problem asks whether this can be achieved for a *finite but arbitrarily large* population. We focus on the *random* version of this problem, where every agent is a copy of the same automaton and non-determinism on the global action chosen by the controller is resolved independently and uniformly at random. Controller seeks to almost-surely gather the agents in the target states. We show that the random population control problem is EXPTIME-complete.

2012 ACM Subject Classification Theory of computation → Formal languages and automata theory

Keywords and phrases Controller synthesis, Parameterized verification, Distance automata

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

An arbitrary number N of tokens are placed on the initial state of a non-deterministic finite automaton. They are updated in rounds, each resolving their non-deterministic choices: in every round, a controller selects a letter a and then every token independently selects and moves to an a-labelled successor state to the one it resides in. The goal for the controller is to eventually gather all tokens in an accepting state. The larger the number N, the harder it is to control the population of N tokens. Crucially, the decision question we study is *parameterized* in the number of tokens: given the automaton, decide whether controller can succeed for arbitrarily large population sizes N?

This approach follows a line of models for biological or chemical systems with large crowds of simple finite-state systems, like population protocols [1] or Petri nets [11]. Population control problems provide a formal framework for the design of strategies to control a large number of identical agents. This framework was introduced in [3], as a model for the synchronization of large populations of yeasts [18]. It fits into parameterized verification, a line of research that aims to verify distributed protocols over arbitrarily large networks [17, 7].

The population control problem has been studied both in the adversarial and stochastic settings, which differ in how agents resolve choices and what guarantees the controller is after. In the adversarial setting, an antagonistic environment resolves all agent's choices, trying to avoid synchronization. Bertrand et al. [2, 3] showed that this (adversarial) population control problem is decidable and EXPTIME-complete.

In the stochastic setting, all agent's choices are made uniformly at random and the controller aims to synchronize the agents almost-surely, with probability one. This *random population control* is known to be EXPTIME-hard [12] and decidable [4, 5]. Colcombet et al.'s decision procedure is based on two key ingredients. First, winning regions are downward-closed with respect to the natural product order on $\mathbb{N}^S$ and can therefore be finitely represented and manipulated as a union of ideals. The second key ingredient involves

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the *sequential flow problem*, which asks whether an unbounded number of tokens can be moved from one set of states to another while remaining within a specific safe region. They solve this problem by a reduction to the boundedness problem on distance automata [10]. The main limitation of this approach is that the termination of the resulting algorithm relies on well quasi-orders, resulting in a non-primitive recursive upper bound.

Our Contribution We show that the random population control problem is EXPTIME-complete (Theorem 10). The upper-bound proceeds in two steps, first arguing that it is sufficient for successful strategies to be based on a bounded abstraction of the winning regions (Theorem 7). The second step is to solve what we call the PATH PROBLEMin exponential time. Roughly speaking, this problem asks whether we can transfer, with positive probability, arbitrarily many tokens from an initial configuration to the set of final configurations, and all the while surely staying in a given set of safe configurations. To solve the PATH PROBLEMwe compute an appropriate semigroup, called the *flow semigroup*, which characterizes the existence of such unbounded paths (Theorem 27). To prove this characterization correct, we use two intermediate semigroups called the *cut semigroups*, one infinite and the other of doubly-exponential size (Theorem 22). We exhibit a duality between the flow semigroup and the symbolic cut semigroup which is a reflection of the max-flow min-cut duality (Lemma 25). This contribution also contains the original (unpublished) lower bound [12] (Theorem 29).

The following example shows an instance where Controller can win using the probabilistic behavior of the tokens.

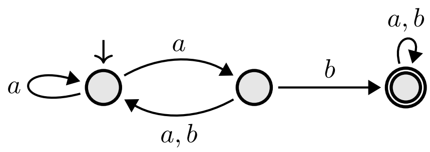


Figure 1 An automaton for which Controller wins against a random environment but not an adversarial one. The initial state is on the left and the target is the one on the right. Omitted, but implicitly present is a sink state, to which all states move on actions not shown.

▶ **Example 1.** The automaton in Figure 1 is a positive instance of the random population control problem. For every population size N, Controller can play the action a until all tokens are in the central state, which almost-surely happens eventually. She can then play b to send some of them to the target, while some are sent back to the first state. At that point, every token that was in the second state has a fixed probability to be in the target state. By repeating this procedure while there are tokens left in the first state, Controller almost-surely ends up putting all tokens in the target state. This shows that even though Controller wins in the stochastic setting, the expected time to synchronize all agents can be exponential in N, since just getting all tokens to the center state takes exponential expected time.

This paper uses hyperlinks. Occurrences of some notions are linked to their definition. The reader can click on words and symbols (or just hover over them on some PDF readers) to see the definition.

2 Preliminaries

We assume familiarity with automata theory [15] and Markov Decision Processes [14] and proceed to recall some necessary notation.

Markov decision processes. A *Markov decision process* (MDP) $\mathcal{M} = (S, \Sigma, \Delta)$ consists of a set S of states, Σ a set of actions, and a transition function $\Delta : S \times \Sigma \to \mathcal{D}ist(S)$, where $\mathcal{D}ist(S)$ denote the set of probability distributions over a set S.

We consider MDPs with almost-sure reachability objectives, given by a set $F \subseteq S$ of target states [1]. When F is reached the play is over and controller wins. If the play is infinite then controller loses. For such objectives it is well-known [13, Thm. B] that stationary (a.k.a. memoryless) strategies suffice to win. In this paper, a *strategy* is therefore a function $\sigma : S \to \mathcal{D}ist(\Sigma)$. A strategy might be partially defined, typically it is not defined on states from which there is no hope to win. If a state on which σ is not defined is reached, and if moreover this state is not final, then the play is over and controller loses. Once a strategy σ is fixed, the resulting process is a Markov chain, whose probability measure is denoted $\mathbb{P}_\sigma$ (see [14] for details). A strategy is *winning* from some state if it ensures that F is reached $\mathbb{P}_\sigma$-almost-surely. The *winning region* is the subset $W \subseteq S$ from which a winning strategy exists.

Random walks in winning regions. In an MDP with finitely many states, there is a canonical almost-surely winning strategy for Controller: play at random any action which guarantees to stay in the winning region. This is formalized using *arenas* and *safe random walks*, as follows.

Call an MDP *simple* if for all $s \in S$, the set of configurations reachable from s is finite. A *commit* is an element of $S \times \Sigma$. An *arena* is a set $W \subseteq S \times \Sigma$ of commits such that for all $(w, a) \in W$ and $s \in S$, if $\Delta(w, a)(s) > 0$ then there exists $b \in \Sigma$ such that $(s, b) \in W$. We occasionally abuse notations and write $s \in W$ instead of $\exists b \in \Sigma, (s, b) \in W$. A path *in* arena W is a path $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} \cdots \xrightarrow{a_k} s_k$ in $\mathcal{M}$ such that $s_j \in W$ for all j and $(s_{j-1}, a_j) \in W$ for all $j > 0$. The *safe random walk* in the arena W is the strategy σ defined on W such that $\sigma(w)$ is the uniform distribution on $\{a \in \Sigma \mid (w, a) \in W\}$. An arena is *winning* if for all $w \in W$ there is a path in W from w to F. This is closely linked with the notion of a winning region, as shown in the next lemma.

▶ **Lemma 2** (Winning arenas). *Given a simple MDP and reachability target F.*

1. *The union of two winning arenas is a winning arena and the winning region is the projection on S of the largest winning arena.*
2. *In a winning arena, the safe random walk is a winning strategy from every state.*

Random Populations. We consider *populations* of random agents, called *tokens*, described by an MDP $\mathcal{M} = (S, \Sigma, \Delta)$.

The n-fold product of $\mathcal{M}$ is the MDP where the controller selects at each step a global action $a \in \Sigma$ which applies simultaneously to all n tokens, whose states are independently updated with respect to the global action. Formally, $\mathcal{M}^{(n)} = (S^n, \Sigma, \Delta)$ is the MDP whose states, called *configurations* here, are n-dimensional vectors with components in S, and Δ is lifted to S^n in the natural way: $\Delta(\mathbf{q}, a)(\mathbf{p}) = \prod_{i=0}^{n-1} \Delta(\mathbf{q}(i), a)(\mathbf{p}(i))$ for all $\mathbf{q}, \mathbf{p} \in S^n$ and $a \in \Sigma$. The configuration of $M(\mathcal{A})^n$ where a state $i \in S$ is duplicated n times is denoted $i^{(n)}$. Equivalently, for convenience, we let T_∞ be a countably infinite set of tokens and configurations are elements in S^T where T is a finite subset of T_∞.

[1] Exact transition probabilities do not matter for these objectives, so one can assume w.l.o.g. that all probability distributions in Δ are uniform. The MDP can be equivalently represented as a non-deterministic automaton (NFA).

The present paper focuses on the following decision problem:

The RANDOM POPULATION CONTROL PROBLEM
Given: an MDP $\mathcal{M} = (S, \Sigma, \Delta)$, initial state $i \in S$ and target set $F \subseteq S$.
Question: Does the winning region of $\mathcal{M}^{(n)}$ with respect to the reachability objective $F^{(n)}$ always contain $i^{(n)}$, for arbitrarily large $n > 0$?

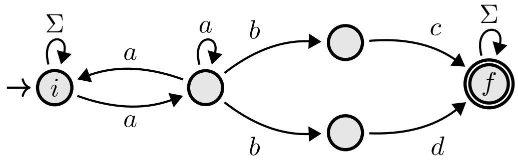


Figure 2 A positive instance of the RANDOM POPULATION CONTROL PROBLEM.

▶ **Example 3.** Controller has a winning strategy in the MDP associated with the automaton of Figure 2, where the target is $F = \{f\}$. Whenever a transition is not represented on the figure, it leads to a losing sink state, which Controller wants to avoid at all cost. Regardless of the size N of the population, she plays the action a until exactly one token is in the second state (which happens eventually with probability 1), and then plays b followed by either b or c (which ever is the safe move) to send that token to the final state. She then repeats that strategy until every token is at the target. This is the only way to progress: if she plays b while more than one token are in the second state, then some may be sent to different states, and she is stuck.

Symbolic configurations and the Path problem. Since the tokens are treated symmetrically, we sometimes abstract configurations by simply counting the number of tokens in each state. This offers a convenient way to represent sets of configurations using ideals.

We use the natural ordering on natural numbers, as well as its extension with a maximal element ω: $\overline{\mathbb{N}} = \{0 < 1 < 2 < 3 < \ldots < \omega\}$. We also use the product ordering on $\overline{\mathbb{N}}^S$: $\overline{x} \leq \overline{y}$ if and only if $\overline{x}(s) \leq \overline{y}(s)$ for all $s \in S$.

▶ **Definition 4** (Symbolic configurations and commits). *A symbolic configuration is an element of $\overline{\mathbb{N}}^S$. A symbolic commit is an element of $\overline{\mathbb{N}}^S \times A$. For a configuration $w \in S^T$ over a set of tokens T, write $|w| \in \mathbb{N}^S$ for the vector that counts tokens in each state: $|w|(s) = |\{t \in T \mid w(t) = s\}|$. The ideal $\overline{w}{\downarrow}$ is the set of all configurations v such that $|v| \leq \overline{w}$. For a set of configurations W and a symbolic configuration $\overline{w} \in \overline{\mathbb{N}}^S$ we say that $\overline{w}$* belongs to W*, denoted as $\overline{w} \in W$, if $\overline{w}{\downarrow} \subseteq W$.*

Another central decision problem in the paper is:

The PATH PROBLEM
Given: an MDP $\mathcal{M} = (S, \Sigma, \Delta)$, a finite set $\overline{W}$ of symbolic commits, an element $\overline{w}_0 \in \overline{W}$ and a subset $\overline{F} \subseteq \overline{W}$. **Question:** Does every configuration in $\overline{w}_0{\downarrow}$, admit a path to $\overline{F}{\downarrow}$ inside $\overline{W}{\downarrow}$?

The input in general contains constants that are arbitrarily large, but for our purpose we will only consider versions with small constants.

▶ **Definition 5.** *The* largest constant *of an instance of the* Path problem*is the largest finite coordinate which appears in one of the finite sets* $(\overline{w}_0, \overline{W}, \overline{F})$ *(by convention* 0 *in the rare cases where there is no such finite coordinate).*

3 Main results

In this section we present our main algorithmic result: the Random population control problem is EXPTIME-complete. This result relies on several intermediary results that are of independent interest, especially the algebraic arguments.

1. Whenever the answer to the Random population control problem is positive, there is a winning arena defined by abstract configuration using coefficients in $\{0, 1, \ldots, |S|, \omega\}$ (Theorem 7 and Section 4);
2. As a consequence, the Random population control problem can be solved with exponentially many calls to a subprocedure solving the Path problem with largest constant 1 (Lemma 9);
3. The Path problem with largest constant 1 reduces to the computation of a (large) finite semigroup, called the *symbolic cut semigroup* (Theorem 22, Section 5);
4. The Path problem with largest constant 1 reduces to the computation of a (small) finite semigroup, called the *flow semigroup*, and is therefore decidable in exponential time (Theorem 27, Section 6).
5. The Random population control problem is EXPTIME-hard (Theorem 29, Section 7).

Combining the first four results, we present in this section an exponential-time fix-point algorithm for the Random population control problem. The presentation of the algorithm and its correctness proof requires introducing the notion of K-definability.

▶ **Definition 6** (K-definability)**.** *A set of configurations is called* K-definable *if it is a union of* ideals *of the form* $\overline{w}{\downarrow}$ *with* $\overline{w} \in \{0, \ldots, K, \omega\}^S$.

In all that follows, denote W the arena of almost-surely winning configuration and commits, and for every $K \in \mathbb{N}$, denote $W_{0,\ldots,K,\omega}$ the maximal K-definable subset of W, i.e., the union of ideals $\overline{w}{\downarrow}$ included in W such that the finite coordinates of $\overline{w}$ are $\leq K$. Note that $\overline{F} \subseteq W_{0,\omega}$. The answer to the Random population control problem is positive if and only if $W_{0,\omega}$ contains $\mathbf{i}$.

We can now state one of the key ingredients of the EXPTIME upper bound.

▶ **Theorem 7** (Almost-surely winning with a few stray sheep)**.** *Let* W *be the* arena *of almost-surely winning configurations and commits. There exists a* sub-arena Y *of* W *such that:*

- Y *contains* $W_{0,\omega}$*; and*
- Y *is* $|S|$*-definable; and*
- Y *is a* winning arena *for reaching* $\overline{F}$.

The main arguments for proving Theorem 7 are exposed in Section 4 and the formal proof is available in appendix Section E.

▶ Remark. Theorem 7 implies that, if we can control arbitrarily many tokens then we can do so with an $|S|$-definable strategy, one that make the same choice in configurations that agree on the token multiplicites up to $|S|$. This has no immediate consequences for the shape of the winning region W. In fact, W may not be $|S|$-definable and its ideal representation may require doubly exponentially large constants (cf. Section B.2). Therefore, in general, Y

is strictly contained in W. Moreover, despite there existing memoryless and deterministic strategies from every configuration in the winning region [13], the $|S|$-definable winning strategies guaranteed by Theorem 7 may still require randomization (cf. Section B.1).

Theorem 7 suggests a simple dynamic programming algorithm to compute Y and decide the random population control problem, see Algorithm 1, which refines a candidate set V until the corresponding ideal is a winning arena.

Recall that we use the notation $u \in W$ for $\exists b \in \Sigma, (u, b) \in W$. For a set $V \subseteq \{0, \ldots, |S|, \omega\}^S \times \Sigma$ of symbolic commits let $V{\downarrow}$ denote the set of commits $(w, a) \in \mathbb{N}^S \times \Sigma$ with $w \leq v$ for some $(v, a) \in V$. Write $\bar{i}$ for the vector mapping i to ω and other states to 0.

Algorithm 1 Algorithm for the Random population control problem

```
1: V ← {0,...,|S|,ω}^S × Σ
2: repeat
3:     if ∃(v̄, a) ∈ V, w ∈ v̄↓, u ∉ V↓ s.t. Δ(w, a)(u) > 0 then
4:         V ← V \ {(v̄, a)}
5:     if ∃(v̄, a) ∈ V and w ∈ v̄↓ s.t. there is no path from w to F̄↓ in V↓ then
6:         V ← V \ {(v̄, a)}
7: until V does not change
8: return ∃a, (ī, a) ∈ V
```

Line 3 can be made effective in time exponential in $|S|$ (cf. Appendix F, Lemma 45).

▶ **Lemma 8.** *Algorithm 1 returns* True *if and only if the answer to the* Random population control problem *is positive.*

Proof. Let $V_0 \supsetneq V_1 \supsetneq \ldots \supsetneq V_n$ be the successive values of V throughout the execution, which terminates by monotonicity.

Suppose the algorithm returns *True* in line 8. Let $Y = V_n{\downarrow}$. As we exited the loop in line 7, no symbolic commit was removed in lines 4 and 6. Thus, Y is a winning arena for reaching F and, according to Lemma 2, the safe random walk in Y almost-surely reaches F.

For the other direction, suppose there is a winning strategy. By Theorem 7 there is a sub-arena Y of W satisfying the three conditions of Theorem 7. Let $X = \{(\overline{w}, a) \in \{0, \ldots, |S|, \omega\}^S \times \Sigma \mid (\overline{w}, a){\downarrow} \subseteq Y\}$. As Y is $|S|$-definable, we have $Y = X{\downarrow}$.

We show that the algorithm maintains the following invariant, defined for $i \in \{0 \ldots n\}$.

$$X \subseteq V_i. \tag{1}$$

This is clear for $i = 0$. Assume the invariant holds for some $i < n$: since $X \subseteq V_i$, we have $Y \subseteq V_i{\downarrow}$. Since $V_i \supsetneq V_{i+1}$, some symbolic commit $(\overline{v}, a)$ is removed from V_i to obtain V_{i+1}. As Y is a winning arena, for all $(y, a) \in Y$ every successor of y by a is in Y, and thus in $V_i{\downarrow}$. Also, for all $y \in Y$ there is a path in Y (and thus in $V_i{\downarrow}$) from y to $\overline{F}$. As a consequence, $(\overline{v}, a){\downarrow} \nsubseteq Y$, hence $(\overline{v}, a) \notin X$. Hence $X \subseteq V_i \setminus \{(\overline{v}, a)\} = V_{i+1}$.

Finally, the invariant holds for $i = n$. Thus, $Y \subseteq V_n{\downarrow}$ Furthermore we have $W_{0,\omega} \subseteq Y$, from Theorem 7. As there is a winning strategy, we also have $\bar{i}{\downarrow} \subseteq W_{0,\omega}$. Hence $\bar{i}{\downarrow} \subseteq Y$, and $\bar{i} \in X$. As a result, $\exists a, (\bar{i}, a) \in V_n$ and the algorithm therefore returns *True*. ◀

The condition in line 5 requires solving the Path problem. The following Lemma, (proved in Appendix F), shows that it suffices to solve instances whith largest constant 1.

▶ **Lemma 9.** *There is a computable reduction from the* Path problem *to the same problem with largest constant* 1*. The reduction increases the state space polynomially, when constants are encoded in unary.*

Section 6 provides an algorithm which solves the Path problem with largest constant 1 is time exponential in the number of states $|S|$ (Theorem 27), hence our main result:

▶ **Theorem 10.** *The* Random population control problem *is EXPTIME-complete.*

Proof. EXPTIME-hardness is Theorem 29. For the upper bound, we rely on the fact that both the Path problem with largest constant 1 and the Path problem with largest constant $|S|$ can be solved in exponential time in the number of states $|S|$. The former result is established in Section 6 (Theorem 27), while the latter is a consequence of Lemma 9. We have already established that Algorithm 1 is correct (Lemma 8). We argue that it takes at most exponential time.

- V has $(|S|+2)^{|S|}|\Sigma|$ elements at the start, and every iteration removes at least one.
- The first condition (line 3) can be checked simply by computing the set of successors of each symbolic commit in V. This can be straightforwardly done in exponential time in the number of states (cf. Appendix F, Lemma 45).
- Checking the second condition (line 5) is an instance of the Path problem with largest constant $\leq |S|$, which as discussed above, is solvable in exponential time in $|S|$.

Therefore, Algorithm 1 takes exponential time in the size of the input. As a result, the Random population control problem is EXPTIME-complete. ◀

4 Winning by counting up to $|S|$

4.1 Overview

The non-elementary complexity in the algorithm of [5] arises from the fact that their algorithm requires to describe a set of configurations defined by potentially very large integers. We show that a full description of the winning region is not required to solve the Random population control problem. Instead, we show that if a winning strategy exists then there is one that tracks only a few tokens and a set of unbounded states.

We will illustrate intuitions using a shepherd-sheep metaphor: the tokens are sheep, split between the *herd* and the *stray sheep* (or just *strays*). Intuitively, we can add as many sheep as we want in states occupied by the herd while staying in the winning region. By contrast, strays occupy bounded places. We describe a winning strategy based on two modes.

The first mode we call *"Funneling obedient sheep"*. Here, the tokens follow a predetermined *funnel path* that leads to a final configuration. On the funnel path, only a few sheep at a time may leave the herd, and none of those who do meet: they are expected to stay in different states. Formally, this means that the path stays within $W_{0,1,\omega}$. The existence of such a path is given by Lemma 14. Controller selects actions to stay on the funnel path, in the hope that all sheep follow it. This is not guaranteed but happens with positive and lower-bounded probability.

The second mode is called *"Gathering the herd"*, and is entered just after some non-obedient stray sheep have left the funnel path. As sheep outside the herd are alone in their state, there are at most $|S|$ of them. Controller's primary objective now is to gather all sheep back together into a herd, including the strays. Formally, the new objective is to reach $W_{0,\omega}$ while remaining in the winning region W. This must be possible because final configurations, which are reachable from everywhere in W, belong to $W_{0,\omega}$.

By Lemma 14 once more, Controller finds a new funnel path to $W_{0,\omega}$ and proceeds according to that. Again, the sheep might deviate from this path and produce more strays. We then select yet another funnel path to recover those and get to $W_{0,\omega}$ and so on. The issue with this approach is that Controller may continue to be unlucky and produce ever more strays. However, as we show in Lemma 15, Controller can play as described above while ensuring that strays introduced at different steps never meet before they are brought back to the herd. This is illustrated in Figure 3.

As at each step we produce at most $|S|$ stray sheep, we can have at most $|S|$ strays in each state. This strategy thus defines a sub-arena of the winning region in which we have a herd of sheep and at most $|S|$ strays in each state at all times, as stated in Theorem 7.

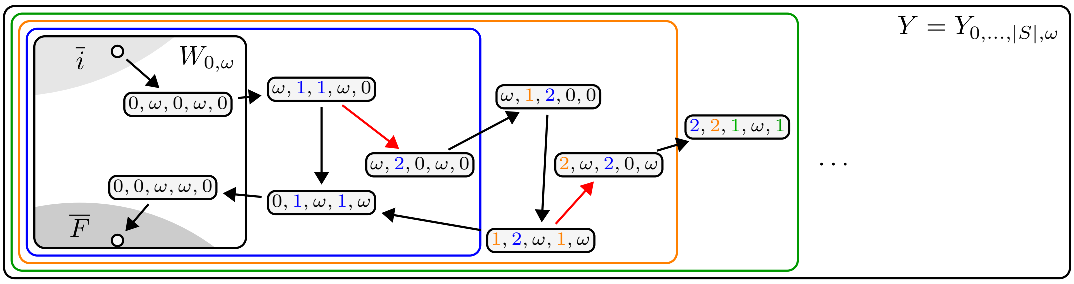


Figure 3 Controller tries to follow the black path from $\bar{i}$ to $\overline{F}$. Isolated tokens (in blue) may be spawned along that path. Lemma 14 guarantees that they are alone in their state at all times along the path. However, we may be unlucky and diverge from the path (red arrow), with those stray tokens potentially meeting. We then try again to reach $W_{0,\omega}$ by selecting a new black path, possibly spawning new isolated tokens (in orange). If we get unlucky again, we try to recover them using a new black path, and maybe spawning some new (green) isolated tokens. Lemma 15 will let us guarantee that tokens of different colors never meet before being brought back to an ω. This means that we have at most $|S|$ layers, since we cannot have more than $|S|$ isolated groups. We define Y as the union of all those layers: it is a winning arena.

4.2 Funneling obedient sheep

This is the first part of the proof of Theorem 7. Informally, in this section is exposed the way to guide a herd of *obedient* sheep to the objective.

Assume that we play in some arena W, from some initial position w_0. In the following statements, the set of tokens T_ω typically denotes the herd and T_f the strays.

In a configuration, we say that a set of states is an ω*-base* if an arbitrary amount of extra tokens could be placed on these states without exiting the arena W. This is formally defined as follows.

▶ **Definition 11** (ω-base and finite base)**.** *Fix an arena W and a configuration $w \in W$. A set of states S_ω is an ω-base of w in W if*

$$w[S_\omega * \omega]{\downarrow} \subseteq W \ .$$

with $w[S_\omega * \omega]$ the symbolic configuration obtained from $|w|$ by mapping states of S_ω to ω.

By extension, a set of tokens T_ω is an ω-base of w in W if the set of states occupied by those tokens in w is. Dually, a set of tokens T_f is a finite base of w in W iff its complement is an ω-base of w in W.

We extend naturally the notions of finite base and ω-base to commits.

The notions of ω-base and finite base will be crucial in the rest of this section. Note that a configuration w may have several ω-bases and finite bases: for instance, say we have two states s_1, s_2 and the winning region is $(\omega, 1)\downarrow \cup (1, \omega)\downarrow$, then the configuration $[t_1 \mapsto s_1, t_2 \mapsto s_2]$ has $\{t_1\}$ and $\{t_2\}$ as ω-bases, but not $\{t_1, t_2\}$.

Finally, we define the type of arena that we will deal with throughout the proofs.

▶ **Definition 12.** *A population arena is an arena that is a finite union of ideals.*

Sometimes we will need to keep track of a small set of tokens. Let T_f be a finite set of tokens. Given a vector $\overline{w} \in \mathbb{N}^S$ and a mapping $v_f : T_f \to S$, the ideal tracking T_f generated by $\overline{w}$ and v_f is written $v_f + \overline{w}\downarrow$ and defined as the set of configurations w such that:

- *for all $t \in T_f$, $w(t) = v_f(t)$*
- *for all $s \in S$, s contains at most $\overline{w}(s)$ tokens of $T_\infty \setminus T_f$*

We extend this notion naturally to commits, with the notation $(v_f + \overline{w}, a)\downarrow$. A population arena tracking T_f is an arena that is a finite union of ideals tracking T_f.

Those arenas are the ones that are closed under renaming and removing tokens, as the winning region naturally is. We sometimes need to keep track of a small set of tokens that should not be renamed or removed. This is why we introduce population arenas *tracking* T_f.

▶ **Definition 13.** *Let w be an arena. For every $K \in \mathbb{N}$, denote $W_{0,\dots,K,\omega,T_f}$ the union of all ideals tracking T_f of the form $v_f + \overline{w}\downarrow$ with $\overline{w} \in \{0, \dots, K, \omega\}^S$ that are included in W.*

▶ **Lemma 14** (Funneling the herd, except for a few loners)**.** *Let T_f be a finite set of tokens, F a finite union of ideals tracking T_f and W a population arena tracking T_f that is winning with respect to F. For every configuration w_0 in W_{0,ω,T_f} there exists a path in $W_{0,1,\omega,T_f}$ from w_0 to F.*

Sketch of proof. The full proof is presented in Appendix C. As W is a population arena tracking T_f, it is the union of finitely many ideals tracking T_f. Let B be the highest number used to define those ideals.

By definition, W_{0,ω,T_f} is also a finite union of ideals tracking T_f. Let $I = v_f + \overline{w}\downarrow$ with $\overline{w} \in \{0, \omega\}^S$ be one of them. Define $I[N]$ as a configuration obtained by taking v_f and adding N tokens on each state such that $\overline{w}(s) = \omega$. We only need to show that for all N we have a path from $I[N]$ to F in $W_{0,1,\omega,T_f}$. Let d be the number of states s such that $\overline{w}(s) = \omega$.

We have $I[B \cdot N] \in I \subseteq W$, hence there is a path from $I[B \cdot N]$ to F in W. Let n be its length. We interpret it as a directed weighted graph G whose vertices are $S \times \{1, \dots, n\}$. Its edges are so that there is an edge from (s, j) to $(s', j + 1)$ whenever there exists $t \in T \setminus T_f$ such that $w_j(t) = s$ and $w_{j+1}(t) = s'$.

We assign weights to all vertices of G according to the ideals of W the associated commits belong to. Those weights are in $\{0, \dots, B, \omega\}$. The trajectories of tokens outside of T_f in the path define a flow in G of capacity dBN. We replace every positive finite weight in G by 1 and use the max-flow min-cut theorem to show that the new graph has a flow $\geq dN$. This flow defines a path from $I[N]$ to F in $W_{0,1,\omega,T_f}$. ◀

4.3 The isolation lemma

Informally, the isolation lemma says that when a group of strays leaves the herd, the strays can be brought back in the herd without ever meeting any other strays outside their group. This will in turn let us bound the number of strays at all times.

We say that two tokens t_1, t_2 *meet* in a configuration w if they share the same state i.e. if $w(t_1) = w(t_2)$.

▶ **Lemma 15** (Isolation Lemma). *Fix a finite set of tokens T_f, a population arena W, and a configuration w_0 of W such that T_f is a finite base of w_0 in W. Denote Z the set of configurations in which a strict subset of T_f is a finite base.*

Assume that W is winning with respect to the set of configurations with a finite base of size $< |T_f|$.

Then there is a strategy σ in W which, when starting from w_0:

- *guarantees to reach Z almost-surely, and*
- *guarantees that the tokens from T_f never meet the other ones until Z is reached.*

Sketch of proof. Since W is a population arena, it is a finite union of ideals. Let B be the highest number used to define those ideals.

Let $M \in \mathbb{N}$. We define w_M as the configuration obtained from w_0 by putting M additional tokens for each token outside T_f in each state containing such tokens. Since T_f is a finite base of w_0, w_M is in W, so the safe random walk in W brings us in Z almost-surely.

We count the expected number of tokens outside T_f met by a token $t \in T_f$ before t is part of an ω-base. Before that happens, t can only meet B tokens at a time. Furthermore, since we are following a safe random walk in a population arena, all tokens are treated symmetrically. Hence every time t meets other tokens outside T_f, it has a probability at least $1/B$ of being the first one of them to become part of an ω-base. Furthermore there can be at most $B|S|$ tokens that are not part of an ω-base in a configuration. From this we infer an upper bound $B^3|S|$ on the expected number of tokens outside T_f met by t before it is part of an ω-base. Since all tokens outside T_f start with at least M other tokens in their state, and they are all treated symmetrically, we can show that each one has a probability $\leq B^3|S|/M$ of meeting t.

Furthermore, when t is saved, there can be at most $B|S|$ tokens in states containing at most B tokens. The probability that a token of T_ω is one of them is at most $B|S|/M$.

For all M, we can apply a randomized strategy from w_0 which simulates the safe random walk with the additional tokens of w_M. Therefore, we can make the probability that t meets a token outside T_f before a token is saved as close to 0 as we want while making the probability that there is a finite base without tokens of T_ω as close to 1 as we want.

Since the set of configurations reachable from W_0 is finite, we conclude that we actually have a strategy that makes those probabilities respectively 0 and 1, yielding the result. The proof is detailed in Appendix D. ◀

4.4 Gathering the sheep

The proof of Theorem 7 uses an induction on (the number of states minus) the number of isolated groups of strays in the arena. The induction step is done as follows:

- We try to follow a path to the target set F. We can choose this path so that we create at most $|S|$ isolated tokens, by Lemma 14.
- If we deviate from that path, we use Lemma 15 to define a sub-arena that lets us recover stray tokens while making sure that they don't meet any token from another group. We then apply the induction hypothesis to recover those stray tokens within an $|S|$-definable sub-arena.
- Once the stray tokens are recovered, we apply the first step again, until we successfully follow the path to the end.

We define the desired sub-arena Y as the union of those paths and sub-arenas. This proof is detailed in Appendix E.

5 An algebraic solution to the Path problem with largest constant 1

In this section the PATH PROBLEM with largest constant 1 is reformulated in the algebraic framework developed by Imre Simon for tackling the unboundedness problem of distance automata [16]. This allows to express the solution as a decidable property of a computable finite semigroup.

For the remainder of this section we fix an MDP $(S, A, p_{\mathcal{M}})$ and a population arena $\overline{W}$ with largest constant 1.

Borrowing notations of [16], consider the semirings:

$$\begin{aligned}
\mathcal{M} &= (\{0 < 1 < 2 < \ldots < \infty\}, \min, +), \text{ the tropical semiring} \\
\mathcal{T} &= (\{0 < 1 < 2 < \ldots < \omega < \infty\}, \min, +), \text{ the extended tropical semiring} \\
\mathcal{R} &= (\{0 < 1 < \omega < \infty\}, \min, \max), \text{ the minmax semiring} \\
\bar{\mathcal{R}} &= (\{0 < 1 < \omega < \infty\}, \max, \min), \text{ the maxmin semiring} .
\end{aligned}$$

where as usual, $+$ is subject to $\infty + x = x + \infty = \infty, \forall x \in \mathbb{N} \cup \{\omega\}$ and $\omega + n = n + \omega = \omega, \forall n \in \mathbb{N}$.

The central objects are *flows* and *cuts*.

▶ **Definition 16** (Flows, cuts and tropical cuts). *Matrices with coefficients in the maxmin semiring $\bar{\mathcal{R}}$ indexed by S^2 are called* flows. *Denote by $\mathcal{P}(S)$ the collection of subsets of S. Matrices with coefficients in the minmax semiring $\mathcal{R}$ (resp. $\mathcal{M}$) indexed by $\mathcal{P}(S)^2$ are called* cuts *(resp.* tropical cuts*).*

Flows which abstract the actions in the arena $\overline{W}$ are called *action flows*.

▶ **Definition 17** (Action flows). *Let*

$$\phi : \mathcal{M} \to \{0, 1, \omega\} \tag{2}$$

which stabilizes 0 and 1 and sends ∞ to ω, as well as any integer ≥ 2. The domain of a matrix $f \in \mathcal{M}^{S^2}$ with coefficients in the tropical semiring is the vector in $\{0, 1, \omega\}^S$ defined by: $\mathrm{dom}(f)(s) = \phi\left(\sum_{t \in S} f(s,t)\right)$. *An* action flow *$f$ is a flow whose coefficients belong to $\{0, 1, \infty\}$ (i.e. there is no ω-entry in f), such that there is an action $a \in A$ which satisfies two conditions:*

$$\begin{aligned}
&(\mathrm{dom}(f), a) \in \overline{W} \\
&\forall s, t \in S^2, f(s,t) \neq 0 \implies p_{\mathcal{M}}(s,a,t) > 0 \ .
\end{aligned}$$

The set of action flows is denoted by $\mathcal{F}$.

There is a duality between cuts and flows, whose basis is the next definition, and which is further developed in the next section.

▶ **Definition 18** (flow-to-cut and cut-to-flow). *For every flow $f \in \bar{\mathcal{R}}^{S^2}$ let $M(f) \in \mathcal{R}^{\mathcal{P}(S)^2}$ denote the cut defined as:*

$$M(f) = \left(\max_{\substack{s \in S_0 \\ t \in S \setminus T}} f(s,t) \right)_{S_0 \in \mathcal{P}(S), T \in \mathcal{P}(S)} , \tag{3}$$

with the convention $\max(\emptyset) = 0$. Conversely, let $M \in \mathcal{R}^{\mathcal{P}(S)^2}$ be a cut. The associated flow f_M is defined as

$$f_M = (M(\{s\}, S \setminus \{t\}))_{s \in S, t \in S} \ .$$

In [16], the decidability of the unboundedness problem for distance automata is tackled using two semigroups. In our specific context, both semigroups are generated by the cuts of action flows $\{M(f), f \in \mathcal{F}\}$. Since the coefficients of action flows belong to $\{0, 1, \infty\}$, they do live in the intersection of the minmax semiring $\mathcal{R}$ and the tropical semiring $\mathcal{M}$. Thus $\{M(f), f \in \mathcal{F}\}$ can be seen both as a family of cuts or a family of tropical cuts.

▶ **Definition 19** (tropical mincut semigroup $\mathcal{C}_{\mathcal{M}}$)**.** *The* tropical mincut semigroup*, denoted $\mathcal{C}_{\mathcal{M}}$, is the sub-semigroup of $\mathcal{M}^{\mathcal{P}(S)^2}$ generated by $\{M(f), f \in \mathcal{F}\}$.*

The tropical mincut semigroup $\mathcal{C}_{\mathcal{M}}$ can perform exact computations of minimal cuts in a graph with capacities (cf. Lemma 48). This is exactly the purpose of the distance automaton introduced in [4], which computes minimal cuts in single-source capacity graphs. This is no coincidence: the correspondance between distance automata and tropical semigroups developed in [16] produces exactly $\mathcal{C}_{\mathcal{M}}$ when applied to the distance automaton of [4]. This expressivity of the tropical mincut semigroup $\mathcal{C}_{\mathcal{M}}$ comes at the cost of being, in general, infinite. We also make use of a *finite* abstraction of $\mathcal{C}_{\mathcal{M}}$ provided by Imre Simon's framework [16], called the *symbolic mincut semigroup*, better suited for algorithmic purposes.

▶ **Definition 20** (symbolic mincut semigroup $\mathcal{C}_{R}$)**.** *The* symbolic mincut semigroup*, denoted $\mathcal{C}_{R}$, is the least sub-semigroup of $\mathcal{R}^{\mathcal{P}(S)^2}$ which contains $\{M(f), f \in \mathcal{F}\}$ and is stable by the iteration operation, defined as follows.*

Let $E = E^2$ be an idempotent symbolic cut of $\mathcal{R}^{\mathcal{P}(S)^2}$. A pair $(S_0, T) \in \mathcal{P}(S)^2$ such that $E(S_0, T) = 1$ is stable *in E if*

$$\exists R \in \mathcal{P}(S), E(R, R) = 0 \wedge \max\{E(S_0, R), E(R, T)\} = 1 \tag{4}$$

and unstable *otherwise. Then the iteration of E, denoted $E^\sharp$, is defined by:*

$$E^\sharp(S_0, T) = \begin{cases} E(S_0, T) & \text{if } E(S_0, T) \in \{0, \omega, \infty\} \\ 1 & \text{if } E(S_0, T) = 1 \text{ and } (S_0, T) \text{ is stable in } E \\ \omega & \text{if } E(S_0, T) = 1 \text{ and } (S_0, T) \text{ is unstable in } E. \end{cases}$$

The following lemma provides some more intuition about (un)stability.

▶ **Lemma 21** (Stability)**.** *Let $E = E^2$ be an idempotent symbolic cut of $\mathcal{R}^{\mathcal{P}(S)^2}$. Let $F = E$ be exactly the same matrix but considered as tropical, i.e. an element of $\mathcal{T}^{\mathcal{P}(S)^2}$. Let $(F^n)_{n \in \mathbb{N}}$ the sequence of powers of F, computed in the extended tropical semiring $\mathcal{T}$. Let $S_0, T \in \mathcal{P}(S)$ such that $E(S_0, T) = 1$. Then the following properties are equivalent:*

a) *(S_0, T) is unstable in E;*

b) *$(F^n(S_0, T))_{n \in \mathbb{N}}$ is a sequence of integers which converges to ∞.*

These tools provide an algebraic solution to the Path problem.

▶ **Theorem 22** (Solving the Path problem with cuts)**.** *Let $S_0 \in \mathcal{P}(S)$ be a non-empty set of states and $\overline{w}_0$ be the symbolic configuration with ω on coordinates in S_0 and 0 elsewhere. The following statements are equivalent:*

(i) *the answer to the Path problem in the arena $\overline{W}$ with initial configuration $\overline{w}_0$ and final states F is positive;*

(ii) *the subset of $\mathbb{N} \cup \{\infty\}$*

$$Z = \left\{ \min_{s \in S_0} M(\{s\}, S \setminus F), M \in \mathcal{C}_{\mathcal{M}} \right\} \tag{5}$$

contains ∞ or is infinite;

(iii) *the symbolic cut semigroup contains an element $M \in \mathcal{C}_R$ such that:*

$$\min_{s \in S_0} M(\{s\}, S \setminus F) \geq \omega \ .$$

Sketch of proof. The equivalence between (i) and (ii) relies on the idea introduced in [4]: exploit the max-flow min-cut duality in order to caracterise the existence of flows carrying an arbitrary amount of tokens as the unboundedness of minimal cuts computed by a distance automaton.

For the equivalence between (ii) and (iii), the case where $\infty \in Z$ is easy to tackle, this is the specific case where a single fixed path can carry an unbounded number of tokens. The case where Z is infinite is solved using a direct pick-up from Simon's toolbox: an application of Simon's tropical unboundedness theorem [16, Theorem 12] to our semigroups.

▶ **Theorem 23** (Simon's tropical unboundedness theorem applied to the cut semigroups)**.** *Let $S_0, T \in \mathcal{P}(S)$. The following statements are equivalent:*

(a) *the integer-valued coefficients of cuts at coordinates (S_0, T) are unbounded, i.e., $\{M(S_0, T) \mid M \in \mathcal{C}_{\mathcal{M}}\}$ is infinite;*

(b) *there is a cut $M \in \mathcal{C}_R$ whose coefficient at the coordinate (S_0, T) is ω, i.e., $\exists M \in \mathcal{C}_R, M(S_0, T) = \omega$.*

This shows the equivalence between (ii) and (iii) in case Z does not contain ∞. ◀

Theorem 22 above provides a doubly-exponential algorithm for the Path problem with largest constant 1: compute $\mathcal{C}_R$ and look for an element satisfying the condition (iii). Using a non-deterministic guess, the complexity can even be reduced to EXPSPACE, following a common argument on the maximal length of strictly decreasing chains of $\mathcal{J}$-classes (see [10] for more details). The next section provides a way to check condition (iii) more efficiently: instead of computing explicitely the (big) symbolic cut semigroup $\mathcal{C}_R$, a smaller semigroup, called the flow semigroup, is computed in EXPTIME.

6 An EXPTIME algorithm to solve the Path problem with largest constant 1

In this section we introduced yet another algebraic structure, called the *flow semigroup*, which is the key to obtain our EXPTIME complexity upper bound.

The symbolic mincut semigroup $\mathcal{C}_R$ is the central finite object in the proof of decidability of the Path problem with largest constant 1 (cf. Theorem 22). However $\mathcal{C}_R$ does contain a lot of redundant information. The flow semigroup is more compact. The two semigroups do run on different gears: whereas $\mathcal{C}_R$ is based on the *minmax* semiring $\mathcal{R}$, the flow semigroup $\mathcal{F}_R$ is based on the *maxmin* semiring $\bar{\mathcal{R}}$.

▶ **Definition 24** (flow semigroup)**.** *The* flow semigroup, *denoted $\mathcal{F}_R$, is the least sub-semigroup of $\bar{\mathcal{R}}^{S \times S}$ which contains the set $\mathcal{F}$ of action flows and is stable by the iteration operation, defined as follows. Let $e = e^2$ be an idempotent flow of $\mathcal{F}_R$. A pair $(s,t) \in S^2$ such that $e(s,t) = 1$ is* unstable *in e if there exists $s_0, t_0 \in S$ such that*

$$e(s, s_0) \geq \omega \wedge e(s_0, t_0) = 1 \wedge e(t_0, t) \geq \omega \ .$$

it is called stable *otherwise. Then the* iteration *of e, denoted $e^\sharp$, is defined by*

$$e^\sharp(s,t) = \begin{cases} e(s,t) & \text{if } e(s,t) \in \{0, \omega, \infty\} \\ 1 & \text{if } e(s,t) = 1 \text{ and } (s,t) \text{ is stable in } e \\ \omega & \text{if } e(s,t) = 1 \text{ and } (s,t) \text{ is unstable in } e. \end{cases}$$

There is a duality between flows and cuts, whose basis is Definition 18, and which is further developped in the next lemma.

▶ **Lemma 25** (duality between flows and cuts). *For every flow $f \in \bar{\mathcal{R}}^{S^2}$ and cut $M \in \mathcal{R}^{\mathcal{P}(S)^2}$, $f_{M(f)} = f$ and $M(f_M) = M$. The symbolic cut semigroup and the flow semigroup are dual from each other:*

$$\mathcal{F}_R = \{f_M, M \in \mathcal{C}_R\} \tag{6}$$

$$\mathcal{C}_R = \{M_f, f \in \mathcal{F}_R\} \ . \tag{7}$$

This duality is the key to solve the Path problem with largest constant 1 using the flow semigroup.

▶ **Theorem 26** (Solving the Path problem with largest constant 1 with flows). *Let $S_0 \in \mathcal{P}(S)$ a non empty set of states. Denote $\overline{w}_0$ the symbolic configuration with ω on coordinates of S_0 and 0 elsewhere. The following statements are equivalent:*

(i) *the answer to the Path problem with largest constant 1 in the arena $\overline{W}$ with initial configuration $\overline{w}_0$ and final states F is positive;*

(ii) *the flow semigroup $\mathcal{F}_R$ contains a flow f such that:*

$$\forall s \in S_0, \exists t \in F, f(s,t) \geq \omega \ .$$

Proof. According to Lemma 25, the condition (iii) of Theorem 22 is equivalent to the condition (ii) of the present theorem. ◀

Condition (ii) is pretty easy to check algorithmically, as a consequence:

▶ **Theorem 27.** *The Path problem with largest constant 1can be solved in exponential time in the number of states $|S|$.*

Proof. It is enough to compute the flow semigroup $\mathcal{F}_R$, according to Theorem 26. This computation performs basic algebraic operations on finite matrices in $\mathcal{F}_R$, a single multiplication or iteration operation is performed in polynomial time. There are at most $4^{|S|^2}$ such matrices, thus the computation of $\mathcal{F}_R$ can be performed in EXPTIME. ◀

7 Lower bound

An exponential time lower bound for the Random population control problem can be shown by reduction from countdown games, as follows.

▶ **Definition 28.** *A Countdown Game is given by a directed graph $\mathcal{G} = (V, E)$, where edges carry positive integer weights, $E \subseteq (V \times \mathbb{N}_{>0} \times V)$. For an initial pair $(v, c_0) \in V \times \mathbb{N}$ of a vertex and a number, two opposing players (Player 1 and 2) alternatingly determine a sequence of such pairs as follows. In each round, from (v, c), Player 1 picks a number $d \leq c$ such that E contains at least one edge (v, d, v'); then Player 2 picks one such edge and the game continues from $(v', c - d)$. Player 1 wins the game iff the play reaches a pair in $V \times \{0\}$.*

Determining the winner of a Countdown Game, where all constants are given in binary, is EXPTIME-complere [9]. We state the lower bound and a sketch of the construction. The full proof is detailed in Appendix A.

▶ **Theorem 29.** *The Random population control problem is EXPTIME-hard.*

Proof sketch. By reduction from solving Countdown games. First observe that the number of turns in a Countdown Game cannot exceed the initial value of the counter, as the initial counter value decreases at each turn. Thus, if Player 2 has a winning strategy, choosing actions at random yields a positive probability of applying that strategy, hence a positive probability of winning. Therefore, Player 1 wins the initial game if, and only if, she wins with probability one against a randomized adversary.

The main idea for our further construction is to require Controller, who impersonates Player 1, to move tokens one-by-one away from a waiting state, first into the control graph of the Countdown game, and ultimately into the target. To avoid a loss in the intermediate phase, she needs to win an instance of the game against a randomizing opponent. This is enforced using a combination of gadgets, including two binary counters that can effectively test for zero, be set to specific numbers, and that are set up so that they can decrement at the same rate. These are used to hold the global integral value of the game n, and an auxiliary counter holding the value d chosen by Player 2. Controller is compelled to reduce them both and can only continue once the auxiliary counter is exhausted. She can only afford to safely end the simulation of the game if the first counter holds value 0. As a result, Player 1 has a winning strategy for the two-player Countdown Game if, and only if, Controller can synchronize the n-fold product of the constructed MDP for all n. ◀

8 Conclusion

We showed that the Random population control problem is EXPTIME-complete. There are two main ingredients for the upper-bound. First, we establish that it is possible to win the population MDP while staying in a part of the winning region that has low descriptive complexity, in the sense of Theorem 7. This is the key to define an algorithm (Algorithm 1) to solve the Random population control problem using an exponential number of calls to an oracle solving the Path problem with largest constant 1. Second, we developed new algebraic tools that allow to solve the Path problem with largest constant 1 in time exponential in the number of states of the MDP (Theorem 27). The upper-bound is optimal: the Random population control problem is EXPTIME-hard (Theorem 29).

These results shed new light on parameterized control and pave the way to further positive results with more ambitious objectives. There is hope, for example, to use the toolset of the present paper to cope with a generalized version of the explorability problem, where infinite executions have to satisfy ω-regular conditions.

Further natural extensions of this work concerns qualitative (as opposed to almost-sure) guarantees, or identifying cases where synchronization can be fast. In practical applications, the requirement that *every token* should be synchronized *with probability 1* may be considered too strong and we may be after less strict, quantitative constraints. For instance, one may want to ensure that a lower-bounded proportion of tokens are synchronized, or with lower-bounded probability below 1.

Even in simple positive instances, the expected time to synchronize all tokens can be exponential in the number of tokens. It is currently open if one can decide the existence of a strategy that requires at most poly-logarithmic (resp. polynomial) time.

References

1. Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Comput.*, 18(4):235–253, 2006. URL: https://doi.org/10.1007/s00446-005-0138-3, doi:10.1007/S00446-005-0138-3.
2. Nathalie Bertrand, Miheer Dewaskar, Blaise Genest, and Hugo Gimbert. Controlling a population. In Roland Meyer and Uwe Nestmann, editors, *28th International Conference on Concurrency Theory, CONCUR 2017, September 5-8, 2017, Berlin, Germany*, volume 85 of *LIPIcs*, pages 12:1–12:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. URL: https://doi.org/10.4230/LIPIcs.CONCUR.2017.12, doi:10.4230/LIPICS.CONCUR.2017.12.
3. Nathalie Bertrand, Miheer Dewaskar, Blaise Genest, Hugo Gimbert, and Adwait Amit Godbole. Controlling a population. *Logical Methods in Computer Science*, Volume 15, Issue 3, July 2019. URL: https://lmcs.episciences.org/5647, doi:10.23638/LMCS-15(3:6)2019.
4. Thomas Colcombet, Nathanaël Fijalkow, and Pierre Ohlmann. Controlling a random population. In Jean Goubault-Larrecq and Barbara König, editors, *Foundations of Software Science and Computation Structures - 23rd International Conference, FOSSACS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25-30, 2020, Proceedings*, volume 12077 of *Lecture Notes in Computer Science*, pages 119–135. Springer, 2020. doi:10.1007/978-3-030-45231-5_7.
5. Thomas Colcombet, Nathanaël Fijalkow, and Pierre Ohlmann. Controlling a random population. *Log. Methods Comput. Sci.*, 17(4), 2021. URL: https://doi.org/10.46298/lmcs-17(4:12)2021, doi:10.46298/LMCS-17(4:12)2021.
6. Leonard Eugene Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *American Journal of Mathematics*, 35(4):413–422, 1913.
7. E. Allen Emerson and Kedar S. Namjoshi. Reasoning about rings. In Ron K. Cytron and Peter Lee, editors, *Conference Record of POPL'95: 22nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, San Francisco, California, USA, January 23-25, 1995*, pages 85–94. ACM Press, 1995. doi:10.1145/199448.199468.
8. Lester Randolph Ford and Delbert Ray Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8:399–404, 1956.
9. Marcin Jurdzinski, François Laroussinie, and Jeremy Sproston. Model checking probabilistic timed automata with one or two clocks. In Orna Grumberg and Michael Huth, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 13th International Conference, TACAS 2007, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2007 Braga, Portugal, March 24 - April 1, 2007, Proceedings*, volume 4424 of *Lecture Notes in Computer Science*, pages 170–184. Springer, 2007. doi:10.1007/978-3-540-71209-1_15.
10. Daniel Kirsten. Distance desert automata and the star height problem. *RAIRO Theor. Informatics Appl.*, 39(3):455–509, 2005. URL: https://doi.org/10.1051/ita:2005027, doi:10.1051/ITA:2005027.
11. Ina Koch. Petri nets–a mathematical formalism to analyze chemical reaction networks. *Molecular Informatics*, 29(12):838–843, 2010.
12. Corto Mascle, Mahsa Shirmohammadi, and Patrick Totzke. Controlling a random population is EXPTIME-hard. *CoRR*, abs/1909.06420, 2019. URL: http://arxiv.org/abs/1909.06420, arXiv:1909.06420.
13. Donald Ornstein. On the existence of stationary optimal strategies. *Proceedings of the American Mathematical Society*, 20(2):563–569, 1969. doi:10.2307/2035700.
14. Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994. doi:10.1002/9780470316887.
15. G. Rozenberg and A. Salomaa. *Handbook of Formal Languages*. Number v. 1-3 in Handbook of formal languages / G. Rozenberg; A. Salomaa. Springer, 1997. URL: https://books.google.co.uk/books?id=bpICQwAACAAJ.

16 Imre Simon. On semigroups of matrices over the tropical semiring. *Informatique Théorique et Applications*, 28(3-4):277–294, 1994.

17 A. Prasad Sistla and Steven M. German. Reasoning with many processes. In *Proceedings of the Symposium on Logic in Computer Science (LICS '87), Ithaca, New York, USA, June 22-25, 1987*, pages 138–152. IEEE Computer Society, 1987.

18 Jannis Uhlendorf, Agnès Miermont, Thierry Delaveau, Gilles Charvin, François Fages, Samuel Bottani, Pascal Hersen, and Gregory Batt. In silico control of biomolecular processes. *Computational Methods in Synthetic Biology*, pages 277–285, 2015.

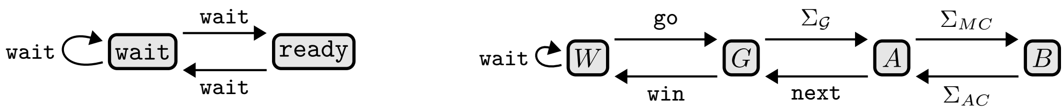


Figure 4 The waiting (on left) and the control gadgets (on right). Edges labelled by Σ_X are shorthand for several edges, one for each action in Σ_X. All but the depicted actions are daemonic.

A Lower Bound

In this section we show the lower bound for our main complexity result.

▶ **Theorem 29.** *The* Random population control problem *is EXPTIME-hard.*

A.1 Countdown Games

A *Countdown Game* is given by a directed graph $\mathcal{G} = (V, E)$, where edges carry positive integer weights, $E \subseteq (V \times \mathbb{N}_{>0} \times V)$. For an initial pair $(v, c_0) \in V \times \mathbb{N}$ of a vertex and a number, two opposing players (Player 1 and 2) alternatingly determine a sequence of such pairs as follows. In each round, from (v, c), Player 1 picks a number $d \leq c$ such that E contains at least one edge (v, d, v'); then Player 2 picks one such edge and the game continues from $(v', c - d)$. Player 1 wins the game iff the play reaches a pair in $V \times \{0\}$.

CountdownGame is the decision problem which asks if Player 1 has a strategy to win a given game for a given initial pair (v_0, c_0). All constants in the input are written in binary.

▶ **Proposition 30** (Thm. 4.5 in [9])**.** *CountdownGame is EXPTIME-complete.*

A.2 The Reduction

In order to reduce CountdownGame to Random population control problemwe first observe that the number of turns in a Countdown Game cannot exceed the initial value of the counter, as the initial counter value decreases at each turn. Thus, if Player 2 has a winning strategy, choosing actions at random yields a positive probability of applying that strategy, hence a positive probability of winning. Therefore Player 1 wins the initial game if, and only if, she wins with probability one against a randomized adversary.

The main idea for our further construction is to require Player 1 to move components one-by-one away from a waiting state, first into the control graph of the Countdown Game, and ultimately into the goal. To avoid a loss in the intermediate phase she needs to win an instance of that game against a randomizing opponent. This is enforced using a combination of gadgets, including two binary counters that can effectively test for zero, be set to specific numbers, and that are set up so that they can decrement at the same rate. As a result, Player 1 has a winning strategy for the two-player Countdown Game if, and only if, the controller can synchronize the n-fold product of the constructed MDP for all n.

For a given Countdown Game $\mathcal{G}$ with an initial pair (v_0, c_0) we construct an MDP $\mathcal{M}$ as follows. We write that action a *takes state* s to *successor* t to mean that $\delta(s, a)(t) > 0$.

A state s is *marked* in a configuration w if at least one token occupies it: $\exists t. w(t) > 0$. Whenever action a takes state s only back to itself we say that s *ignores* a. There are states `Heaven` (the target) and `Hell` which ignore all actions. For a given state s, an action a is *angelic* if it takes s only to `Heaven`, and *daemonic* if it takes s to `Hell`. An action a is *safe* in a configuration if it is not daemonic for any marked state (in any gadget).

Besides the special states `Heaven` and `Hell`, $\mathcal{M}$ contains several gadgets described below.

Waiting. The waiting gadget has two states `Wait` and `Ready` which react to the action `wait` as depicted in Figure 4 (left). Whenever a configuration marks one of these states, a strategy that continuously plays `wait` will almost-surely reach a configuration in which exactly one component marks `Ready`.

A special action `go` (to indicate successful isolation of one component) takes `Ready` to the initial state v_0 of the game $\mathcal{G}$. All other actions (in gadgets described below) are ignored. This is similar to what happens in Example 3.

Game. The game $\mathcal{G} = (G, E)$ is directly interpreted as MDP: For every edge $(s, d, s') \in E$ there is an action (s, d) which takes s to s' and which is daemonic for all states $s' \neq s$.

The action `win` is angelic for every state of G. All other actions are ignored.

Binary Counters. A (k-bit) Counter consists of states `(i:j)` for all $0 \leq i < k$ and $j \in \{0, 1\}$. For every bit i there is a decrement action `(deci)` which

- takes `(j:0)` only to `(j:1)` for all $0 \leq j < i$,
- takes `(i:1)` only to `(i:0)`,
- is daemonic for `(i:0)`, and
- is ignored by all `(j:l)`, for all $i < j$ and $l \in \{0, 1\}$.

We say that a configuration *holds* the number $c < 2^k$ in this counter if it marks those states that represent the binary expansion of c: for all $0 \leq i \leq k-1$, state `(i:j)` is marked iff the ith bit in the binary expansion of c is j. An action a *sets* the counter to number d if for all $0 \leq i < k$, it takes `(i:0)` to only `(i:j)` where $j \in \{0, 1\}$ is the ith bit in the binary expansion of d, and is daemonic for all `(i:1)` (to ensure that the counter can only be set if it holds 0).

Additionally, for every bit i the gadget has an error action `errori`, which is daemonic for `(i:0)` and `(i:1)`, and angelic for every other state (of $\mathcal{M}$). These actions can be used to quickly synchronize any configuration in which the counter is not correctly initialized, i.e., does not hold a number. See Figure 5 for a depiction of a 4-bit counter.

The MDP $\mathcal{M}$ will contain two distinct counter gadgets. A main counter MC has $\log_2(n_0)$ bits to hold possible counter values of the Countdown Game. An auxiliary counter AC has $\log_2(d_{\mathtt{max}})$ many bits to hold the largest edge weight $d_{\mathtt{max}}$ in $\mathcal{G}$. These have distinct sets of states and actions, so for clarity, we write $C.x$ to refer to state (or action) x in gadget C. We connect some new actions to these two counters as follows.

- The action `go` sets MC to n_0; this ensures that MC holds n_0 when starting to simulate $\mathcal{G}$.
- The action `win` is daemonic for every state $MC.$`(i:1)`. This enforces that the MC must hold 0 when a strategy claims Player 1 wins $\mathcal{G}$.
- Any action $(v, d) \in \Sigma_{\mathcal{G}}$ sets AC to d;
- The action `next` is daemonic for every state $AC.$`(i:1)`. This enforces that a strategy must first count down from d to 0 before it can simulate the next move in $\mathcal{G}$.

Control. The control gadget will enforce that a synchronizing strategy proposes actions in a proper order; see Figure 4. It consists of states W, G, A, B, and contains actions of all gadgets above (including `go`, `win`, `next`) and a new `error` action, which is angelic for all states except W, for which it is daemonic. All omitted edges in Figure 4 are daemonic.

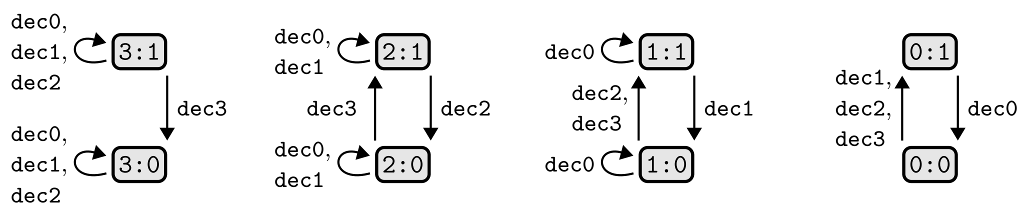


Figure 5 A (4-bit) Binary Counter. Not displayed are edges labelled by (deci) that make the respective actions daemonic for state (i:0), and error actions errori, which are daemonic for (i:0) and (i:1), for all bits $i \in \{0, 1, 2, 3\}$.

Start/End. To complete the construction of $\mathcal{M}$, we introduce an initial state `Init` and actions `start` and `end`. The action `start` takes `Init` to `Wait` (Waiting gadget), W (Control gadget), and all `(i:0)` states of counters AC and MC. It is daemonic for every other state.

The action `end` is daemonic for `Wait` and `Ready`, and angelic for every other state in $\mathcal{M}$.

▶ **Theorem 31.** *$\mathcal{M}^{(n)}$ is synchronizable for all $n \in \mathbb{N}$ iff Player 1 wins $\mathcal{G}$.*

Proof. Suppose Player 1 wins the game $\mathcal{G}$. Fix n. Recall that in $\mathcal{M}^{(n)}$ all components of the initial configuration mark `Init`. A synchronizing strategy proceeds as follows:

- Play `start` to initialize the Waiting and Control gadgets, and to set AC and MC to 0. If any of the gadgets is not correctly initialized afterwards, play the respective error action to win directly. For instance, if W is unmarked, play `error` to synchronize.
- Reduce the number of components marking `Wait` one by one until a configuration is reached in which `Wait` is not marked. Once this is true, play `end` to synchronize.
- To reduce the number of components marking `Wait`, isolate one of them, and move it to `Heaven` by simulating the Countdown Game:
  1. Play `wait` until only a single component marks `Ready`, then play `go`. This will mark v_0 in the game gadget and sets MC to n_0. Recall that (v_0, n_0) is the initial pair of $\mathcal{G}$.
  2. Simulate rounds of the game $\mathcal{G}$: assume state v in the game gadget is marked and the counter MC holds c, then let d be the the number Player 1 plays to win from the pair (v, c) in $\mathcal{G}$. Play (v, d). This action will set AC to d. Alternate between (safe) decrement actions in AC and AB until they hold 0 and $c - d$, respectively. Play `next`.
  3. The above simulation of rounds in $\mathcal{G}$ is repeated until both AC and AB hold 0, by assumption that Player 1 wins $\mathcal{G}$ this is possible. At this point it is safe to play `win`.

Conversely, assume that Player 1 cannot win $\mathcal{G}$. Suppose that after the (only possible) initial move `start`, all gadgets are correctly initialized. Clearly, for every n, this event has strictly positive probability. We argue that no strategy can synchronize such a configuration. Indeed, a successful strategy had to play a sequence in $\mathtt{wait}^* \cdot \mathtt{go}$ first, followed by actions in $(\Sigma_{\mathcal{G}} \cdot (\Sigma_{AC} \cdot \Sigma_{MC} \cdot \mathtt{next})^*)^*$, by construction of the control gadget. If after playing `go`, more than one component mark v_0, there is a non-zero chance that these will diverge, making subsequent actions in $\Sigma_{\mathcal{G}}$ unsafe. If exactly one component marks v_0 then the second sequence of actions (assuming all actions are safe) corresponds to a play of $\mathcal{G}$. This inevitably leads to a configuration in which counter MC holds 0 and the control enforces that the next action is in Σ_{MC}. But any such action will be daemonic for some state in MC and thus not be safe. We conclude that every strategy will lead to a configuration that at least one component marks `Hell` and thus cannot be synchronized. ◀

Theorem 29 follows immediately from Proposition 30 and Theorem 31.

B Hard Cases

Throughout this section, we construct an automata relying on the the language introduced in Appendix A.2. Specifically, a state s is *marked* in a configuration w if at least one token occupies it: $\exists t.w(t) > 0$. Whenever action an a takes state s only back to itself we say that s *ignores* a. There are states `Heaven` (the target) and `Hell` which ignore all actions. For a given state s, an action a is *angelic* if it takes s only to `Heaven`, and *daemonic* if it takes s to `Hell`. An action a is *safe* in a configuration if it is not daemonic for any marked state (in any gadget).

B.1 A Butterfly

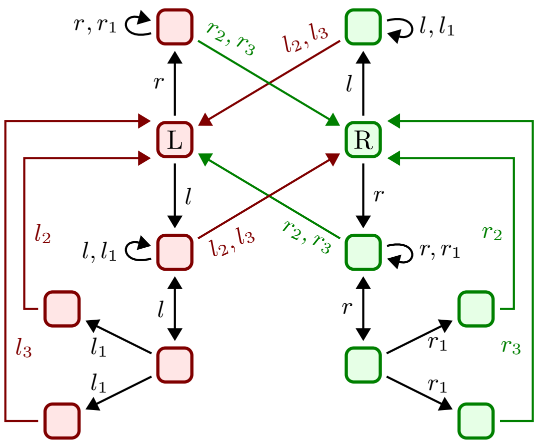


Figure 6 An automaton where Controller can synchronize any finite number of tokens but no deterministic and k-definable winning strategy exists.

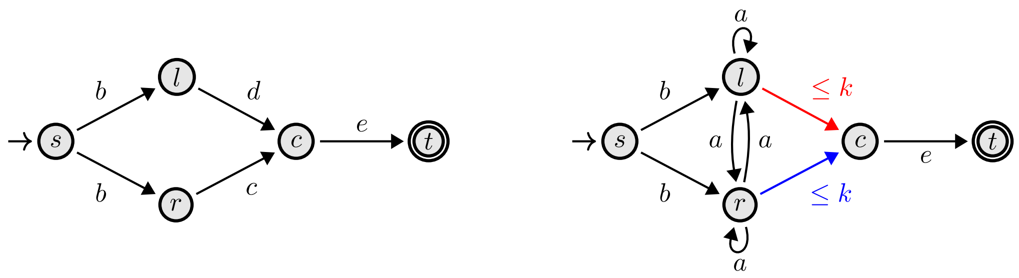
Consider now the example in Figure 6. We will set it up so that initially, all tokens are randomly distributed onto states L and R, and Controller must eventually place all of them on only on one side of the graph. To do so,

- add an initial action which moves tokens from an initial state to L and R and is daemonic everywhere else;
- add a fresh winning action that is angelic for all red states and daemonic for all green states; and
- add a fresh winning action that is angelic for all green states and daemonic for all red states.

Idea Each round starts with all tokens on L and R. Controller stepwise proposes a sequence of actions, either in $l^+l_1[l_2l_3]$ or $r^+r_1[r_2r_3]$. Notice that until one side is empty, these are the only safe sequences to play. At the end of each round, all tokens (except possibly one) will switch sides. Controller can chose to isolate one token and keep it on its side, thereby getting closer to her goal of moving everyone to one side. The relevant decisions to make are

1. whether to go left or right at the start of a round
2. when to stop playing l (or r, resp.)

Recall that a strategy is K-definable if it bases all decision only on a $\{0, 1, \omega\}$ abstraction. That is, it can be given as mapping from the finitely many symbolic configurations with constants in $\{0, 1, \ldots, \omega\}$, to distributions over actions.



(a) A bottleneck with capacity 1.

(b) A bottleneck with capacity $2k$.

Figure 7 Depiction of bottleneck gadgets using actions in $\Sigma = \{a, b, c, d, e\}$. Each starts in a state s and action b, and ends in a target state t with action e.

▶ **Lemma 32.** *The example is a positive instance of* Random population control problem*. However, for every $k \geq 0$, every deterministic and k-definable strategy is losing.*

Proof. Referring to the revevant decisions above, a winning strategy is to 1. always pick the smaller side and 2. play l (or r) until exactly one token is isolated.

Consider a deterministic and k-definable strategy. If for 2., a strategy does not separate exactly one token, then the next round will start in an equivalent position. For 1., by virtue of being determinisctic and k-definable, our strategy must either always play left or right, with probability one, because the start of a round will place ω's onto L and R and 0 elsewhere. After two rounds one necessarily ends in an equivalent configuration. ◀

B.2 A Chain of Bottlenecks

One central result of this work, Theorem 7, implies that a variant of this conjecture holds. Namely, for a positive instance of the population control problem with K many states, there exists a K-definable strategy: one that gives the same distribution over actions from all configurations that agree on token counts up to K. The example in Appendix B.1 shows that such strategies still need to randomize. Here, we show that for every K, one can construct a positive instance of the population control problem in which all $< K$-definable strategies are not winning. Our construction to show this uses the bottleneck gadgets above, in a way that prevents winning strategies that are 1-definable.

We present gadgets that we call *bottlenecks* with a fixed capacity k. These are designed so that Controller can "pass through" up to k many tokens, but not more, meaning that each such gadget is a negative instance of the random population control problem.

Bottlenecks Consider first the automaton in part Figure 7(a). One readily sees that it is a negative instance of the population control problem (both in antagonistic and stochastic settings). Controller can however safely move one token from start s to the target t.

Figure 7(b) shows how to construct bottlenecks of arbitrary finite capacity. To do this, we replace the the red (blue) edge by a bottleneck of capacity k so that 1) all its actions are distinct to, and ignored outside of, that red (blue) gadget. 2) the global actions a, b, e are daemonic for all but the last state of the red (blue) gadget. For instance, for $k = 2$, both red and blue edges are replaced by disjoint copies of the capacity-1 bottleneck in Figure 7(a).

▶ **Lemma 33.** *Controller can simultaneously move $2k$, but not more, tokens through a bottleneck of capacity $2k$.*

Proof. She can do so by playing b and then repeating actions a until exactly k tokens reside on states l and r, respectively. She can then use the actions in the red (blue) gadgets to move these tokens to the central state c. Only then is it safe to play the action e and move them all to the target. Note that, no action other than a is safe to play until tokens are equally split among states l, r. Indeed, the action a is daemonic outside of these two states. Therefore, once an action from the red or blue gadgets is played, further redistributing tokens betweek l and r is impossible. From that point onwards, the only way to move all tokens to the target t is to move them through the red (blue) gadgets. ◀

It is natural to conjecture that for any positive instance of the population control problem and $N \in \mathcal{N}$, Controller has a winning strategy that considers only which states currently host $0, 1$, or more tokens. After all, such strategies are sufficient to isolate and move a single token as in the canonical example in Figure 2 and if a gadget is traversable for k tokens then also, with the same strategy, for strictly fewer tokens.

A length-K chain of bottlenecks with capacity 1 We join K many copies of the gadget depicted in Figure 8 so that for all $1 \leq n < K - 1$, the right-most state of the n-th copy is the left-most state of the $(n+1)$-th copy. he initial state is q_1, the start of the first copy at level one, and the ultimate target state is q_{K+1}, the last state in the K-th copy. Notice that each member of the chain is an instance of Figure 2 followed by a bottleneck of capacity one (Figure 7(a)), and operates on its own alphabet Σ_n of actions. We further impose the following constraints.

1. In all states with index n, all actions in $\Sigma_{<n}$ can be ignored and all actions b_m, e_m with $m > n$ are daemonic.
2. In states s_n, action e_n is daemonic and in state c_n, action b_n is daemonic.
3. In states q_n, actions b_m, for $m \leq n$, lead back to q_1 (indicated by the blue arrow).

▷ Claim 34. To send even one token completely through the n-th gadget, Controller has to gather all tokens in state q_n *twice*: once upon entering and once upon exiting the bottleneck.

Notice that, when moving a large number of tokens from q_0 to q_{K+1}, she will reach at least one configuration with exactly one token on every state c_i for $0 \leq i < K$, at which point she choses action e_0. Indeed, to move a token from c_{K-1} to the target, she needs to play action e_{K-1}, which is blocked until all other tokens are moved up to the final gadget. This is done by untimately playing e_{K-2} many times, which is similarly blocked by the presence of lower-indexed tokens and so on.

A Leaky Chain We describe the input automaton, which has two disjoint parts. The first is the chain of K many bottlenecks of capacity 1 described just above.

The second part of the construction is a bottleneck of capacity K. This gadget is disjoint from, and its actions are ignored throughout, the chain of bottlenecks. The purpose of the capacity-K bottleneck is to recover up to K many tokens and move them back to the initial state q_1 in the chain. The final part in our construction is to add edges from all states c_n to the initial state of the capacity-K bottleneck. This happens on action e_0 and is indicated by the red arrow in Figure 8.

We summarize in the following lemma.

▶ **Lemma 35.** *In the automaton constructed above is a positive instance of the population control problem: For every $N \in \mathcal{N}$, Controller has a strategy to almost-surely move N tokens from q_1 to q_{K+1}.*

For $N \geq K$, every $(K-1)$-definable strategy is losing.

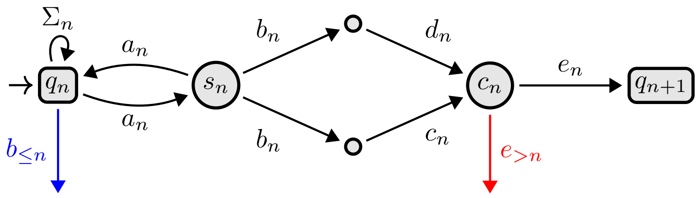


Figure 8 A member of the chain

Proof. She can win by combining strategies for the individual bottleneck gadgets: push one token into a chain i, have it wait in state c_i while all tokens on lower-indexed states move back to q_n, then move it out. This proceeds recursively and ultimately moves all tokens to q_{K+1}. On the way following the suggested strategy, she will visit configurations where all state c_i holds at most one token each, and where she next plays action e_1, potentially sending all of these $\leq K$ tokens to the recovery gadget. If at least one of them follow this route, she follows the strategy in the K-bottleneck to move these tokens back to s_1 and start over, which is possible by Lemma 33. This shows points 1 (the strategy outlined is winning) and 2, because Controller cannot avoid placing K tokens into the capacity-K bottleneck and by Lemma 33.

◀

C Proof of Lemma 14

Let us start by defining flows and cuts. Let $\overline{\mathbb{R}^+}$ be the set of non-negative real numbers, with an additional maximum element ω. The addition is naturally extended: $\omega + r = r + \omega$ for all $r \in \overline{\mathbb{R}^+}$.

Given a finite directed graph $G = (V, E)$, a capacity function $c : V \to \overline{\mathbb{R}^+}$, and two vertices src and tgt (a source and a target), we define a *flow* as follows. It is a function $f : E \to \overline{\mathbb{R}^+}$ such that for all $v \in V \setminus \{s, t\}$, we have

$$\sum_{(v_-, v) \in E} f(v_-, v) = \sum_{(v, v_+) \in E} f(v, v_+) \leq c(v).$$

The *value* of the flow is then defined as $\sum_{(src,v)\in E} f(src, v)$.

A *cut* is a set M of vertices such that every path from src to tgt goes through one of those vertices. Its *value* is $\sum_{v \in M} c(v)$.

The classic *max-flow min-cut theorem* states that the maximum value of a flow is equal to the minimal value of a cut [8].

We state it here with the capacities on the vertices, as it is convenient for the next proof. However, it is usually defined with capacities on the edges $c : E \to \overline{\mathbb{N}}$. The constraints on the flow is then $\sum_{(v_-,v)\in E} f(v_-, v) = \sum_{(v_-,v)\in E} f(v_-, v)$ for all $v \in V$ and $f(e) \leq c(e)$ for all $e \in E$. A cut is then defined as a set of edges, and the theorem is stated analogously.

For the proofs of the results of Sections 5 and 6 we will use that second convention.

Finally, another classic result is the *integer flow theorem*. It says that if all capacities in the graph are integers, then there is an integer maximal flow $f : E \to \overline{\mathbb{N}}$. This is a by-product of the Ford-Fulkerson algorithm [8].

▶ **Lemma 14** (Funneling the herd, except for a few loners). *Let T_f be a finite set of tokens, F a finite union of ideals tracking T_f and W a population arena tracking T_f that is winning* *with respect to F. For every configuration w_0 in W_{0,ω,T_f} there exists a path in $W_{0,1,\omega,T_f}$ from w_0 to F.*

Proof. As W is a population arena tracking T_f, it is the union of finitely many ideals tracking T_f. We call those the *ideal decomposition of* W. Let B be the highest number used to define those ideals.

By definition, W_{0,ω,T_f} is also a finite union of ideals tracking T_f. Let $I = v_f + \overline{w}\downarrow$ with $\overline{w} \in \{0, \omega\}^S$ be one of them. Let S_ω be the set of states s such that $\overline{w}(s) = \omega$. Define $I[N]$ as a configuration obtained by taking v_f and adding N tokens on each state of S_ω. We only need to show that for all N we have a path from $I[N]$ to F in $W_{0,1,\omega,T_f}$. Let d be the number of states s such that $\overline{w}(s) = \omega$. Let $N \in \mathbb{N}$, we show that such a path exists.

We have $I[B \cdot N] \in I \subseteq W$, hence there is a path $w_0 \xrightarrow{a_1} w_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} w_n$ with $w_0 = I[B \cdot N]$ and $w_n \in F$. Consider the following directed weighted graph G. Its vertices are $S \times \{1, \ldots, n\}$, plus a source *src* and a target *tgt*. Its edges are so that there is an edge from (s, j) to $(s', j+1)$ whenever there exists $t \in T \setminus T_f$ such that $w_j(t) = s$ and $w_{j+1}(t) = s'$. We add edges from *src* to $(s, 0)$ and from (s, n) to *tgt* for all $s \in S$.

We assign weights to all vertices of G as follows: for all $j \in \{0, \ldots, n-1\}$, we pick a maximal ideal tracking T_f $(v_j + \overline{w}_j, a_{j+1})\downarrow$ in the ideal decomposition of W such that (w_j, a_{j+1}) is in it. For $j = n$, we pick a maximal ideal tracking T_f $(v_n + \overline{w}_n)\downarrow$ in F in which w_n is. For all state s and index j, the vertex (s, j) is assigned weight $\overline{w}_j(s)$. Furthermore, for all $s \in S$, $(s, 0)$ has weight $B \cdot N$ if $\overline{w}(s) = \omega$ and 0 otherwise. Both *src* and *tgt* have weight ω.

The trajectories of tokens outside of T_f in the path $w_0 \xrightarrow{a_1} w_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} w_n$ naturally define a flow ϕ from *src* to *tgt* in G of capacity dBN: for each edge $e = ((s, j-1), (s', j))$ we define $\phi(e)$ as the number of tokens of $T \setminus T_f$ going from s to s' at the jth step. We also define $\phi((src, (s, 0))) = BN$ for all $s \in S_\omega$ and 0 otherwise. For all $s \in S$, $\phi(((s, n), tgt))$ is the number of tokens of $T \setminus T_f$ in s at the end of the path. This flow satisfies the capacity constraints of G by definition of the $\overline{w}_j$.

Let G^1 be the graph identical to G except that:

- the weight of $(s, 0)$ is N if $s \in S_\omega$ and 0 otherwise.
- every other positive finite weight has been replaced by 1

We claim that G^1 has an integer flow of capacity dN. Suppose the contrary, then by the max-flow min-cut theorem there is a cut in G^1 of weight $< dn$. Then, as the weight of a vertex in G is always at most B times the one in G^1, that same cut has weight $< dBN$ in G, contradicting the existence of a flow of value dBN. By contradiction, we obtain that G^1 has a flow of value dN. This value is optimal as $\{(s, 0) \mid s \in S\}$ is a cut of weight dN.

By the integer flow theorem, as all weights over G^1 are integers, it has an optimal integer flow. This integer flow defines a path from $I[N]$: at step j, the tokens of T_f move in the same way as in the previous path. The number of other tokens sent from state s to s' is given by the flow between $(s, j-1)$ and (s, j) in the graph. It stays in W as for all j, the jth commit is constrained to belong to the ideal $(v_{j-1} + w_{j-1}, a_j)\downarrow$ by the weights. In fact, as every positive finite coordinate has been replaced by 1, the jth commit belongs to a smaller ideal, with a finite base made of tokens of T_f and tokens that are alone in their state. Consequently, it belongs to $W_{0,1,\omega,T_f}$.

It ends in F as the weights of $(s, n)_{s \in S}$ constraint the final configuration to be in $v_n + \overline{w}_n\downarrow \subseteq F$.

This concludes the proof. ◀

D Proof of Lemma 15

▶ **Definition 36** (Meetings). *We say that two tokens $t_1, t_2 \in T$ meet in a configuration $w \in S^T$ if they are placed on the same state i.e. if $w(t_1) = w(t_2)$.*

▶ **Lemma 15** (Isolation Lemma). *Fix a finite set of tokens T_f, a population arena W, and a configuration w_0 of W such that T_f is a finite base of w_0 in W. Denote Z the set of configurations in which a strict subset of T_f is a finite base.*

Assume that W is winning with respect to the set of configurations with a finite base of size $< |T_f|$.

Then there is a strategy σ in W which, when starting from w_0:
- *guarantees to reach Z almost-surely, and*
- *guarantees that the tokens from T_f never meet the other ones until Z is reached.*

Proof. As W is a population arena, it is a finite union of ideals. As a consequence, there exists a bound B such that $W = W_{0,\dots,B,\omega}$.

Denote T_ω the set of tokens in w_0 which are not in T_f, and S_ω the states occupied in w_0 by tokens in T_ω. By hypothesis, S_ω is an ω-base of w_0 in W. Let $M \in \mathbb{N}$, and w_M the configuration obtained from w_0 by adding M tokens for each token in T_ω, on the state of that token. We write T_M for the set of tokens on S_ω in w_M, and call them *the herd tokens* of w_M.

We say a token $t \in T_f \cup T_M$ is *saved* in a configuration if it is in an ω-base of that configuration. By definition of B, while a token $t \in T_f \cup T_M$ is *not* saved then it shares its state with at most $B-1$ other tokens. Denote Z' the set of configurations where one of the tokens in T_f is saved. Then $Z \subseteq Z'$. Note that this inclusion may be strict. Indeed, it might be the case that there is a finite base not containing t but containing some tokens of T_ω which have "left the herd".

Let σ be a safe random walk in W, which guarantees to reach Z almost-surely from everywhere in W, and thus in particular to reach Z' almost-surely.

Let t be one of the stray sheep of T_f. When t meets one or more tokens of T_M, we call this event a *meeting*. Denote V the random variable counting the number of different herd tokens met by t before reaching Z'.

We show that the expected value of V is finite and upper-bounded independently of M. To do so, we argue that the probability that V is above $B \cdot |S| \cdot \ell$ decreases exponentially with ℓ.

▷ Claim 37. The probability that $V \geq B \cdot |S| \cdot \ell$ is at most $(1 - 1/B)^\ell$.

Proof. First, we show that, every time t meets some herd tokens of w_M, there is probability at most $1 - 1/B$ that one of the tokens met by t is saved before t is. Until Z' is reached, no more than $B-1$ other tokens can share the same state than t.

Since W is a population arena, the random walk σ only depends on the number of tokens on every state, not their exact identity. In particular, the random walk σ does not make a difference between t and the herd tokens meeting t. After a meeting, all of them thus have the same probability measure on their possible future trajectories (and btw this measure only depends on the current number of tokens in each state). Since σ guarantees to almost-surely reach Z someday, all of the herd tokens met by t will almost-surely be saved someday, thus t as well. By symmetry, all of them have the same probability to be among the first to be saved, thus the probability that none of the $\leq B-1$ herd tokens is saved before t is at least $1/B$.

Now, look at those finite pathes where t meets at least $B|S|$ different herd tokens, and still t is not saved. By definition of B, among those $B|S|$ herd tokens, at least 1 has been saved. Thus, at every moment the probability that t meets $B|S|$ different herd tokens in the future is at most $1 - 1/B$, hence

$$\mathbb{P}_{\sigma,w_M}(V \geq B \cdot |S| \cdot \ell) \leq (1 - 1/B)^\ell \ . \tag{8}$$

This completes the proof of Claim 37. ◁

Consequently, we can bound the expected value of V independently of M.

$$\mathbb{E}_{\sigma,w_M}[V] \leq \sum_{\ell\in\mathbb{N}} B \cdot |S| \cdot (\ell + 1) \cdot (1 - 1/B)^\ell = B^3 \cdot |S| \ .$$

Let h be a herd token, we can then bound its probability to meet t before reaching Z'. As h starts in the same state as at least M other herd tokens, and σ treats all tokens symmetrically, all those tokens have the same probability p of meeting t. As a result, the expected number of them which meet t is $\geq M \cdot p$. On the other hand, this number is bounded by the expected total number of meetings between t and herd tokens. We obtain $Mp \leq B^3 \cdot |S|$ and thus $p \leq \frac{B^3|S|}{M}$. As a result, the probability that a herd token h meets t before t is saved converges to 0 as M grows.

Recall that T_ω denotes the set of tokens in S_ω in w_0, and T_M the set of tokens in S_ω in w_M. The safe random walk σ_M from w_M can be projected onto a strategy from w_0 by simulating the additional tokens in $T_M \setminus T_\omega$. The resulting strategy is randomized. According to (8), the probability that a token in T_ω meets a token in T_f before Z' occurs can be made arbitrarily small, by increasing the number M of tokens being simulated, while guaranteeing at the same time that Z' occurs almost-surely.

Since the set of configurations reachable from w_0 is finite, and the corresponding condition is a reachability condition under safety constraint, the probability can be turned to 0: there is a memoryless strategy that achieves an optimal probability (i.e. 1) of reaching Z' [13] without any meeting between T_f and other tokens. Hence there is a strategy to reach Z' almost-surely, while keeping the tokens in T_f isolated. We have shown that there exists a strategy to reach Z' almost-surely while making sure that no token of T_f meets a token outside T_f.

We have not quite proven our goal: when starting the play in w_0 we can keep the tokens in T_f isolated and reach almost-surely Z', but what about reaching Z? The inclusion $Z \subseteq Z'$ might be strict because when one of the stray sheep t in T_f is saved, it might be that other sheep from the herd prevent a strict subset of $T_f \setminus \{t\}$ to be a finite base.

Let T'_f the set of tokens that share their state with $\leq B$ other tokens when t is saved. By definition, there are at most $B|S|$ tokens in T'_f. Call the tokens of $T_M \cap T'_f$ the *blockers*. Then the expected number of blockers is $\leq B|S|$. But then, by symmetry, the probability that a given token from T_ω becomes a blocker is less than $B|S|/M$. Furthermore, by definition of B, the non-blockers must form an ω-base of the configuration reached when t is saved. Thus, by increasing M and following σ_M, we can have a probability arbitrarily close to 1 that the configuration when we reach Z' has a finite base with no tokens of T_ω.

Using the same argument as before, we obtain that from w_0 we can make the probability that a token in T_ω is a blocker as close to 0 as we want, while ensuring reaching Z' and keeping the tokens in T_f isolated.

Again, since the set of configurations reachable from w_0 in W is a finite MDP, we can make this probability 0, in which case reaching Z' is equivalent to reaching Z. We have shown that there exists a strategy to reach Z almost surely while making sure that tokens of T_f never meet tokens outside T_f before reaching Z. ◀

E Proof of Theorem 7

▶ **Theorem 7** (Almost-surely winning with a few stray sheep). *Let W be the arena of almost-surely winning configurations and commits. There exists a sub-arena Y of W such that:*

- *Y contains $W_{0,\omega}$; and*
- *Y is $|S|$-definable; and*
- *Y is a winning arena for reaching $\overline{F}$.*

In this section we combine the funneling lemma (Lemma 14) and the isolation lemma (Lemma 15) in order to almost-surely gather the stray sheep back in the herd, while keeping the total number of stray sheep below some bound.

In order to articulate those lemmas together, we need the following definitions. Throughout the proofs, we track specific subsets of tokens. However, we sometimes need to re-anonymize some of them: for instance, to apply Lemma 15, we need W to be a population arena, which does not track any specific token. This is why we introduce the following closure operation, which closes an arena under renaming and removing tokens (apart from a set T_f), making it a population arena (tracking T_f).

▶ **Definition 38.** *Let W be a set of configurations and commits and T_f a finite set of tokens. We define $\overline{W}^{T_f}$ as its closure under renaming and deleting tokens outside T_f. When $T_f = \emptyset$ we simply write $\overline{W}$.*

Formally, for all configuration $w \in W$ we define $\phi_{T_f}(w)$ as the pair $(v_f, \overline{w}) \in S^T \times \mathbb{N}^S$, where v_f is the projection of w on T_f and $\overline{w}$ counts the number of other tokens in each state. Then we define $\overline{W}^{T_f} = \bigcup_{(v_f,\overline{w}) \in \phi_{T_f}(W)} (v_f, \overline{w})\downarrow$.

▶ **Lemma 39.** *Let T_f be a finite set of tokens, let F be a union of ideals tracking T_f. Let W be a winning arena with respect to F. Then $\overline{W}^{T_f}$ is a population arena tracking T_f, and is winning with respect to F.*

Proof. The fact that $\overline{W}^{T_f}$ is an arena is straightforward. Dickson's lemma implies that the set $\bigcup_{(v_f,\overline{w}) \in \phi_{T_f}(W)} (v_f, \overline{w})\downarrow$ is in fact a finite union of ideals tracking T_f [6]. There exists a finite set $V \subseteq \overline{\mathbb{N}}^S$ such that $\overline{W}^{T_f} = \bigcup_{(v_f,\overline{w}) \in V} (v_f, \overline{w})\downarrow$.

Consequently, $\overline{W}^{T_f}$ is a population arena tracking T_f. It is winning as a consequence of the fact that F is a union of ideals tracking T_f, and is thus closed under renaming and deleting tokens outside T_f.

◀

We will now tackle the induction on the number of groups of stray sheep that lets us prove Theorem 7. In all that follows we call a set of tokens *isolated* in a configuration w if every state contains either only tokens of that set or no token of that set.

▶ Remark 40. We can assume without loss of generality that Σ contains a letter $\square$ that labels a loop on each state of $\mathcal{A}$, and no other transition. Adding that letter clearly does not affect the answer to the Random population control problem.

▶ **Lemma 41** (Induction). *Let $T_1, \ldots, T_d$ be disjoint non-empty sets of tokens of size at most $|S|$. Let $T_f = \bigcup_{i=1}^{d} T_i$. Let I a set of initial configurations containing at least one token of each T_i. Let F be a finite union of ideals tracking T_f of the form $v_f + \overline{w}\downarrow$ with $\overline{w} \in \{0, \omega\}^S$.*

Let W a population arena tracking T_f such that:

- *$I \subseteq W$, and for all $w \in I$, T_f is a finite base of w in W, and*
- *for all $w \in W \setminus F$, for all s and T_i, either $w^{-1}(s) \subseteq T_i$ or $w^{-1}(s) \cap T_i = \emptyset$,and*
- *W is winning with respect to reaching F*

Then there exists a winning sub-arena Y of W such that $I \subseteq Y$ and $Y = Y_{0,\ldots,|S|,\omega,T_f}$.

Proof. We proceed by induction on $|S| - d$.

Suppose $|S| = d$. By hypothesis on W, the tokens of $T_1, \ldots, T_d$ occupy disjoint sets of states in every configuration of $W \setminus F$. Furthermore, there is at least one token of each set and no other token can be on the states they occupy. Hence for all $w \in W \setminus F$, the only tokens in w are the ones of $T_1, \ldots, T_d$, and they each occupy one state. As a consequence, we have $W = W_{0,\ldots,|S|,\omega,T_f}$. Hence setting $Y = W$ yields the result.

Now suppose $|S| > d$. By Lemma 14, for all $w_0 \in W_{0,\omega,T_f}$ there is a path in W from w_0 to F along which every commit is in $W_{0,1,\omega,T_f}$.

Let V be the set of configurations $v \in W$ such that there is a non-empty set T^v of at most $|S|$ tokens such that:

- $T_f \cup T^v$ is a finite base of v, and
- all T_i and T^v are isolated in v.

For each $v \in V$, we choose such a set T_v of minimal size. We also define F^v the set of configurations of W with a strict subset of $T_f \cup T^v$ as a finite base.

Note that as all T_i are isolated in every configuration of $W \setminus F$, they must be contained in every finite base of every configuration of $W \setminus F$. As a consequence, every configuration of F^v is either in F or has a finite base of the form $T_f \cup T'$ with $T' \subsetneq T^v$.

▷ Claim 42. For every commit (w, a) in $W_{0,1,\omega,T_f}$, every successor v of (w, a) is either in F, in W_{0,ω,T_f} or in V.

Proof. First of all, as $W_{0,1,\omega,T_f} \subseteq W$ and W is an arena, $v \in W$.

Suppose $v \notin F$, then $T_1, \ldots, T_d$ are isolated in v. Let IT be the set of tokens that are not in any T_i and that were alone in their state in w. Let T^v be the set of tokens of IT whose state in v only contains tokens of IT. Note that by definition of IT, we have $|T^v| \leq |IT| \leq |S|$.

It remains to show that $T_f \cup T^v$ is a finite base of v in W. As $(w, a) \in W_{0,1,\omega,T_f}$, we know that $T'_f = IT \cup \bigcup_{i=1}^{d} T_i$ is a finite base of (w, a). Let S_ω be the set of states containing tokens outside T'_f in w. It is an ω-base of (w, a).

Let S'_ω be the set of states reachable from S_ω by playing a. As S_ω is an ω-base of (w, a), we can safely play a from any configuration obtained from w by adding tokens in states of S_ω. As we may then get arbitrarily many tokens in each state of S'_ω, it is an ω-base of v in W. Every token outside T'_f must end up in S'_ω in v. Consequently, T'_f is a finite base of v in W.

Furthermore, since $v \notin F$, tokens of T_f cannot meet other tokens, hence states containing tokens of IT either contain only those or are in S'_ω.

If T_f is a finite base of v then v is in W_{0,ω,T_f}. Otherwise, T_f is not a finite base of v in W and thus $T^v \neq \emptyset$. Then, as $T_f \cup T^v$ is a finite base of v in W, and all T_i and T^v are isolated in v, we have $v \in V$. ◁

Now, for all $v \in V$, we define an arena Y^v that allows us to bring back some tokens of T^v in the herd while keeping all T_i and T^v isolated.

▷ Claim 43. For all $v \in V$, there is a sub-arena Y^v of W such that:

- $v \in Y^v$

- $Y^v = Y^v_{0,\dots,|S|,\omega,T_f \cup T^v}$
- a safe random walk in Y^v almost-surely reaches F^v
- $T_1, \dots, T_d, T^v$ are isolated in every configuration of $Y^v \setminus F^v$

Proof. Observe that F^v is a finite union of ideals tracking $T_f \cup T^v$. As F only contains configurations of which T_f is a finite base, $F \subseteq F^v$. Furthermore, F^v is a subset of the set F' of configurations with a finite base of size $< |T_f \cup T^v|$. Hence W is also winning for F'.

Since F' is a finite union of ideals, we can apply Lemma 39 with $T_f = \emptyset$ to conclude that $\overline{W}$ is a population arena and is winning with respect to F'.

By Lemma 15, there is a strategy to almost-surely reach F^v from w while remaining in W and ensuring that $T_f \cup T^v$ stays isolated while we have not reached F^v.

As we stay in W, we furthermore maintain the fact that the groups of tokens $T_1, \dots, T_d$ are isolated. As a consequence, $T_1, , \dots, T_d, T^v$ and the rest of the tokens occupy disjoint sets of states at all times. We can therefore define R the set of commits reachable from v while following this strategy. It is clearly an arena. It is a winning arena with respect to reaching F'. By Lemma 39, so is $\overline{W'}^{T_f \cup T^v}$. Furthermore, since $R \subseteq W$ and W is a population arena tracking T_f, we have $\overline{R}^{T_f \cup T^v} \subseteq \overline{W}^{T_f \cup T^v} = W$.

We apply the induction hypothesis to get a winning sub-arena Y^v of R (and of W) such that $\{v\} \subseteq Y^v$ and $Y^v = Y^v_{0,\dots,|S|,\omega,T_f \cup T^v}$. ◁

Define $X = W_{0,1,\omega,T_f} \cup \bigcup_{v \in V} Y^v \cup \{(w, \Box) \mid w \in F\}$. Note that as $v \in Y^v$ for all v, $V \subseteq X$. The letter $\Box$ is given by Remark 40. It is used here to integrate F in Y without conflicts in definitions (an arena being a set of commits and not configurations).

▷ Claim 44. X is a winning arena with respect to F.

Proof. We start by showing that X is an arena. Let $(x, a) \in X$. If $a = \Box$ then the only successor is x itself. If $(x, a) \in Y^v$ for some $v \in V$, as Y^v is an arena, the successors are all in Y^v. If $(x, a) \in W_{0,1,\omega,T_f}$, then by Claim 42 every successor v is in V, and thus in X.

We now show that for all $x \in X$, there is a path from x to F.

First, we now show that for all $v \in V$, there is a path from every configuration of Y^v to F, by strong induction on $|T_v|$. Let $v \in V$, and $x \in Y^v$. As Y^v is winning with respect to F^v, there is a path in Y^v from y to some configuration $z \in F^v$.

- If $z \in F$ then we are done
- If $z \notin F$ then $T_1, \dots, T_d$ are isolated in z.
  - If $z \in W_{0,\omega,T_f}$ then $z \in P$ and there is a path from z to F in P
  - If $z \notin W_{0,\omega,T_f}$ then $z \in V$. As $z \in F^v$, $|T^z| < |T^v|$. We can apply the induction hypothesis to get a path from z to F.

In both cases we have a path from x to z and from z to F, thus from x to F.

Then, observe that every configuration in $W_{0,1,\omega,T_f}$ is either in F, in V or in W_{0,ω,T_f}. In the first case, we are done. In the second case we have already shown the existence of the path to F, and in the third case we simply apply the fact that $P \subseteq W_{0,1,\omega,T_f} \subseteq Y$. ◁

Clearly $I \subseteq W_{0,\omega,T_f} \subseteq P \subseteq Y$.

In order to satisfy the condition that $Y = Y_{0,\dots,|S|,\omega,T_f}$, we close Y under renaming and removing tokens outside T_f and downward-closure.

By Lemma 39, we obtain a population arena tracking T_f that is winning for F. Clearly $W_{0,1,\omega,T_f}$ is closed under those operations. For all v, as $Y^v = Y^v_{0,\dots,|S|,\omega,T_f \cup T^v}$, and T^v has at most $|S|$ elements and is isolated in all of $Y^v \setminus F^v$, the set $\overline{Y^v}^{T_f}$ is a union of ideals tracking T_f with largest constant $|S|$.

As a result, so is Y. Hence $Y = Y_{0,\dots,|S|,\omega,T_f}$. ◀

Proof of Theorem 7. Since W is the winning region, it is the maximal winning arena. As a consequence, by Lemma 39, we have $W = \overline{W}$, hence W is a population arena. We can then straightforwardly apply Lemma 41 with $d = 0$ (and $T_f = \emptyset$). ◀

F Reduction to the Path problem with upper-bound 1

▶ **Lemma 9.** *There is a computable reduction from the* Path problem *to the same problem with largest constant* 1*. The reduction increases the state space polynomially, when constants are encoded in unary.*

Proof of Lemma 9. We prove that there exists an algorithm which solves the Path problem in time $\leq 2^{\mathcal{O}(|S|)}$ whenever the largest constant is $\leq |S|$. Let $(\mathcal{M}, I, W, F)$ an instance. We create a new instance $(\mathcal{M}', \overline{w}'_0, \overline{W}', \overline{F}')$ whose finite constants are at most 1, whose size is polynomial in the size of $(\mathcal{M}, \overline{w}_0, \overline{W}, \overline{F})$, and for which the answer to the Path problem is the same as for $(\mathcal{M}, \overline{w}_0, \overline{W}, \overline{F})$.

The definition of $(\mathcal{M}', \overline{w}'_0, \overline{W}', \overline{F}')$ is as follows. Let K be the sum of the finite coordinates in $\overline{w}_0$ (at most $|S|^2$). Let T_0 be a set of K tokens. The MDP $\mathcal{M}'$ consists in $K+1$ disjoint copies of $\mathcal{M}$. The initial configuration w'_0 is obtained as follows. On the K first copies of $\mathcal{M}$, we place exactly 1 token of T_0 on one of the state, so that the projection matches the finite coordinates of w_0. On the $K+1$th copy, we place the ω like in w_0. The final configurations F' are those whose sum over the different copies is in F. From every anonymous configuration w' of $\mathcal{M}'$ one can obtain a configuration $w = \phi(w')$ in $\mathcal{M}$ by summing up the coordinates in the different copies. Fix an abstract configuration $\overline{w}$ in $\mathcal{M}$ we denote $\psi(\overline{w})$ all the abstract configurations $\overline{w}'$ of $\mathcal{M}'$ such that:

- the coordinates of $\overline{w}'$ on the K first copies are either 0 or 1; and
- the coordinates of $\overline{w}'$ on the $K+1$th copy are in $\{0, 1, \omega\}$; and
- $\phi(\overline{w}'\downarrow) \subseteq \overline{w}\downarrow$.

Remark that there are at most $|S|^K \cdot 3^{|S|}$ such configurations. We define $W' = \psi(W)$ and $F' = \psi(F)$. According to Lemma 14, the answer to the Path problem is the same for $(\mathcal{M}', \{w'_0\}, W', F')$ and $(\mathcal{M}, \{w_0\}, W, F)$. ◀

▶ **Lemma 45.** *Given a finite set $V \subseteq \{0, \dots, |S|, \omega\}$ of symbolic commits with constants below $|S|$ and $(\overline{v}, a) \in V$, one can check, in exponential time in $|S|$, if there exist a commit $(w, a) \in (\overline{v}, a)$ and a configuration $u \notin V\downarrow$ with $\Delta(w, a)(u) > 0$.*

Proof. Notice that if such configurations w and u exists, then already a small one. Indeed, let u_- a configuration obtained as follows: in every state $s \in S$, if u has more than $|S|+1$ tokens in s, remove every token from s except $|S|+1$ of them. Remove the same tokens from w to obtain a configuration w_-.

As constants in V do not exceed $|S|$, and as $u \notin V\downarrow$, u_- is not in $V\downarrow$. As $w \in V\downarrow$, we have $w_- \in V\downarrow$

Observe that u_- and w_- contains at most $|S|^2 + |S|$ tokens. We can enumerate triples w, a, u (up to renaming tokens) with at most $|S|^2 + |S|$ tokens in w and u, and check that they satisfy the conditions, in time $|S|^{\mathcal{O}(|S|^2)}$. ◀

G Multi-source flow

We extend the notion of flow (defined in Section C) in a graph $G = (V, E)$ with capacities $c : E \to \overline{\mathbb{R}^+}$ to sets of sources and targets $Src, Tgt \subseteq V$. Note that we put capacities on the edges here.

The constraints are adapted naturally: a flow $f : E \to \overline{\mathbb{R}^+}$ must be such that for all $v \in V \setminus (Src \cup Tgt)$, we have $\sum_{(v_-,v)\in E} f(v_-, v) = \sum_{(v_-,v)\in E} f(v_-, v)$ and for all $e \in E$, $f(e) \le c(e)$. The *value* of the flow is defined as $\min_{src\in Src} \sum_{(src,v)\in E} f(src, v)$.

In the following sections we will need the following observation. It says that if we have a weighted graph with d sources, if we can have a flow of N from any individual source, we can have a flow of N/d from all sources simultaneously.

As a consequence, if we can have arbitrarily large flows from each of the sources, then there are arbitrarily large flows for the whole set of sources.

▶ **Lemma 46.** *Let $G = (V, E)$ be a graph and $c : E \to \overline{\mathbb{N}}$ an integer capacity function. Let $Src, Tgt \subseteq V$ be sets of sources and targets, and $N \in \mathbb{N}$. If for all $s \in Src$ there is a flow from s to Tgt of value at least N, then there is an integer flow $f : E \to$ from Src to Tgt of value $\ge N/|Src|$.*

Proof. Let $d = |Src|$. For convenience we will assume that d divides N. Add a node s' to G, with outgoing edges to each $s \in Src$ with capacity N/d. We distinguish two cases.

If a maximal flow f from s' to Tgt has value $\ge N$ then as $\{(s', s) \mid s \in Src\}$ is a cut, all those edges must be mapped to N/d by f (and f must have value N). Therefore the projection of that flow on G is a flow from Src to Tgt whose output on each $s \in Src$ is N/d. We conclude by the integer flow theorem.

If the maximal flow from s' to Tgt is $< N$ then there is a minimal cut C and some $s \in Src$ such that $(s', s) \notin C$. As a consequence, $C \cap E$ must be a cut from s to Tgt, of value $< N$. However, there is a flow of value N from s to Tgt, contradicting the max-flow min-cut theorem. ◀

H Proof of Lemma 21

The proof of Lemma 21 is provided for the curious reader, this lemma is not used in the proof of other results thus the rest of the paper is independent of Lemma 21.

Proof of Lemma 21. By definition of the extended tropical semiring, for every $n \in \mathbb{N}$

$$F^n(S_0, T) = \min \left\{ \sum_{i\in 0\ldots n} F(S_i, S_{i+1}) \mid S_0, S_1, \ldots, S_{n+1} = T \in \mathcal{P}(S) \right\} . \tag{9}$$

Assume (a) holds. Let $K \in \mathbb{N}$ and $n \ge K \cdot 2^{|S|} + 1$. Let $S_0, S_1, \ldots, S_n = T \in \mathcal{P}(S)$ which minimizes $x_n = F^n(S_0, T) = \sum_{i\in 0\ldots n-1} F(S_i, S_{i+1})$. Since $E^n(S_0, T) = E(S_0, T) = 1$ then $x_n \le n$. Since $n \ge K \cdot |S|$, there is a state R appearing K times in the sequence at indices $k_0 < k_1 < \ldots < k_K$. Let $z = \max\{E(S_0, R), E(R, T)\} \in \{0, 1, \omega, \infty\}$. Then

$$z = 1 \tag{10}$$

because $(E(S_0, T) = 1$ implies $z \ge 1$ and $z < \omega$ since $x_n \le n$. Let $y = E(R, R)$.

$$y = 0 \iff E(R, R) = 0 \iff \forall k, F^k(R, R) = 0 \ . \tag{11}$$

By definition of unstability, $y \geq 1$. Then (b) holds because $x_n \geq \sum_{i=1\dots K} F^{k_i - k_{i-1}}(R,R) \geq K$, and, by choice, x_n minimizes $F^n(S_0, T)$.

Conversely assume (b) holds. Let R such that $\max\{E(S_0,R), E(R,T)\} = 1$. then $F^n(S_0,T) \leq F(S_0,R) + F^{n-2}(R,R) + F(R,T)$. We conclude with (11) that $E(R,R) \neq 0$. Thus (a) holds. ◀

I Algebraic solution with flows: proof of Theorem 22

Proof of Theorem 22. There are two steps, starting with the equivalence between (i) and (ii), followed by the equivalence between (ii) and (iii).

The equivalence between (i) **and** (ii) is proved using sequences of action flows, called *pipelines*.

▶ **Definition 47** (pipelines and pathes through them). *A pipeline is a sequence $(f_1,\dots,f_\ell)$ of length $\ell \geq 1$ of action flows, i.e. a finite non-empty word in $\mathcal{F}^+$. A path $w_0,\dots,w_\ell \in S^X$ of a finite set of tokens X is a* path through *$(f_1,\dots,f_\ell)$ if the number of tokens travelling along edges respect the flow constraints, i.e. for every states $s,t \in S$ and index $k \in 1\dots\ell$*

$$| \{x \in X \mid (w_{k-1}(x), w_k(x)) = (s,t)\} | \leq f_k(s,t) \ .$$

The next lemma establishes upper and lower bounds on the capacity of a pipeline. This proof technique was introduced in [4] to show decidability of the Path problem.

▶ **Lemma 48** (max-flow min-cut in a pipeline). *Denote*

$$\psi : \mathcal{F}^+ \to \mathcal{C}_{\mathcal{M}}$$

the morphism which computes the product of elements of a pipeline in the tropical cut semigroup. Let $P = (f_1,\dots,f_\ell) \in \mathcal{F}^+$ a pipeline. For every subset $S_0 \subseteq S$, denote

$$B_P(S_0,F) = \psi(P)(S_0, S \setminus F) \in \mathbb{N} \cup \{\infty\} \ .$$

Let n be an integer and X_n a finite set of tokens placed on the initial configuration $w_0^{(n)}$ such that there are $\overline{w}_0[n](s)$ tokens on state s. A path of X_n through P is said to be final *if it carries all tokens to F, i.e. in the final configuration w_ℓ, $\forall x \in X_n, w_\ell(x) \in F$. Define the* capacity of P*, denoted $C_P(S_0,F)$, with value in $\mathbb{N} \cup \{\infty\}$, as*

$$C_P(S_0,F) = \sup\left\{n \in \mathbb{N} \mid \text{ there is a final path from } w_0^{(n)} \text{ through } (f_1,\dots,f_\ell)\right\} \ .$$

Then

$$\frac{1}{|S|}\min_{s \in S_0} B_P(\{s\},F) \leq C_P(S_0,F) \leq \max_{s \in S_0} B_P(\{s\},F) \ . \tag{12}$$

Proof. Given a sequence $1 \leq i_1 < i_2 < \ldots < i_k \leq \ell$, denote $Q(i_1 < i_2 < \ldots < i_k)$ the pipeline obtained from P by erasing all 1-edges of flows at indices $i_1 < i_2 < \ldots < i_k$. Say that this sequence *disconnects a set of states $S_0 \in \mathcal{P}(S)$ from F* if there is no path through the pipeline $Q(i_1 < i_2 < \ldots < i_k)$ which can carry a *single* token from one of the states in S_0 to F. The cost of the disconnection of S_0 from F in P, denoted $D_P(S_0,F)$ is the minimal length k of a disconnecting sequence, or ∞ if there is no such sequence.

▶ Remark 49. Note that $D_P(S_0,F) = \infty$ iff removing all 1-labelled edges will not disconnect S_0 from F, i.e. there exists $s \in S_0$ and an $\{\infty\}$-labelled path in P connecting s to F. In that case, an easy induction shows that no sequence of cuts $\{s\} = S_0, S_1, \dots, S_\ell = S \setminus F$ can isolate $\{s\}$ from F without cutting through an ∞ edge of the path, thus $B_P(S_0, S \setminus F) = \infty$.

Let $1 \leq i_1 < i_2 < \ldots < i_k \leq \ell$ be a disconnecting sequence of minimal length. The difference between P and $Q(i_1 < i_2 < \ldots < i_k)$ is that at least k and at most $k \cdot |S|$ edges of capacity 1 have been deleted from P to obtain $Q(i_1 < i_2 < \ldots < i_k)$. According to the minflow/maxcut theorem for single source capacity graphs, for every state $s \in S_0$,

$$D_P(\{s\}, F) \leq C_P(\{s\}, F) \leq D_P(\{s\}, F) \cdot |S|$$

We show:

$$\frac{1}{|S|} \min_{s \in S_0} D_P(\{s\}, F) \leq C_P(S_0, F) \leq \max_{s \in S_0} D_P(\{s\}, F) \ . \tag{13}$$

The right inequality in (13) is pretty obvious, because $\forall s \in S, C(S_0, F) \leq C(\{s\}, F)$, indeed if for some n there is a final path from $w_0^{(n)}$ through $(f_1, \ldots, f_\ell)$ then for every state $s \in S_0$, we can delete from this path tokens which are not in s initially and get a final path through $(f_1, \ldots, f_\ell)$ starting from the initial configuration $s[n]$ where n tokens are in s.

The left inequality in (13) relies on the next lemma.

▶ **Lemma 50** (maximum concurrent flow). *Let $N \in \mathbb{N}$. For every state $s \in S_0$, denote $s[N]$ the initial configuration where N tokens are placed on s. Assume that for every $s \in S_0$, there is a final path from $s[N]$ through P. Then there is a final path from $\overline{w}_0[\lfloor N/|S_0| \rfloor]$ through P.*

Proof. This a direct application from Lemma 46. ◀

We prove the following relation between B_P and the cost of disconnection, for every pipeline P and sets $S_0, T \in S_D$,

$$D_P(S_0, T) = B_P(S_0, T) \ . \tag{14}$$

The informal reason is that $\mathcal{C}_{\mathcal{M}}$ this is the semigroup associated to the distance automaton used in [4] to compute the equivalent of disconnecting sequences.

We start the proof of (14) with the easiest direction:

$$D_P(S_0, F) \geq B_P(S_0, S \setminus F) \ . \tag{15}$$

If $D_P(S_0, F) = \ell$ then there is a pipeline $P = (f_1, \ldots, f_\ell)$ in which deleting all 1-edges disconnects S_0 from F. Define by induction

$$S_i = \{t \exists s \in S_{i-1}, f_i(s, t) = \infty\}, 1 \leq i \leq \ell \ .$$

Then

$$B_P(S_0, F) = (M(f_1) \cdot \ldots \cdot M(f_\ell))(S_0, S \setminus F) \leq \sum_{i \in 1 \ldots \ell} M(f_i)(S_{i-1}, S_i) \leq \ell = D_P(S_0, F) \ ,$$

since every $M(f_i)$ has coefficients in $\{0, 1, \infty\}$ and S_i is chosen so that $M(f_i)(S_{i-1}, S_i) < \infty$.

The base case is when the length of P is 1, i.e. $P = (f_1)$. Then $D_P(S_0, T) \in \{0, 1\}$ and according to (15) also $B_P(S_0, T) \in \{0, 1\}$. The case $D_P(S_0, T) = 0$ happens when deletion is not necessary for disconnecting S_0 from F, i.e. when f_1 is 0 on every edge connecting S_0 to F, in which case $D_{(f_1)}(S_0, T) = 0 = \max_{s \in S_0, t \in S \setminus F} f_1(s, t) = B_P(S_0, F)$. The case $D_P(S_0, T) = 1$ arises when deletion is necessary for disconnecting S_0 from F, i.e. $\exists (s, t) \in S_0 \times T, f_1(s, t) \geq 1$, which is equivalent to $B_P(S_0, F) \geq 1$. For the induction step, take $P_\ell = (f_1, \ldots, f_\ell)$ and $P_{\ell+1} = (f_1, \ldots, f_\ell, f_{\ell+1})$. Assume we disconnect S_0 from T in $P_{\ell+1}$ with a sequence $u = i_1 < i_2 < \ldots < i_k$ of minimal cost. Denote u' the sequence with is equal to u if $\ell+1$ does not appear in the disconnecting sequence, i.e. $i_k < \ell+1$ and otherwise u' is equal to u minus the last element $i_k = \ell+1$ removed. Denote R' the set of states *not* reachable from S_0 in the pipeline $Q(u')$. Since u is optimal for disconnecting F from S_0 in $P_{\ell+1}$, then:

- u' is optimal for disconnecting S_0 from $S \setminus R'$ in P_ℓ,
- u erases $\ell+1$, (i.e. $i_k = \ell+1$) iff R is not already disconnected from T in the pipeline $(f_{\ell+1})$.

Thus,

$$D_{P_{\ell+1}}(S_0, T) = D_{P_\ell}(S_0, S \setminus R') + \max_{r \in R', t \in T} f_{\ell+1}(r,t) \ ,$$

and by optimality again,

$$D_{P_{\ell+1}}(S_0, T) = \min_{R \in \mathcal{P}(S)} \left(D_{P_\ell}(S_0, S \setminus R) + \max_{r \in R, t \in T} f_{\ell+1}(r,t) \right) \ . \tag{16}$$

By definition of the product in the tropical mincut semigroup,

$$\begin{aligned} B_{P_{\ell+1}}(S_0, T) &= \min_{R \in \mathcal{P}(S)} \left(B_{P_\ell}(S_0, S \setminus R) + M(f_{\ell+1})(R, S \setminus T) \right) \\ &= \min_{R \in \mathcal{P}(S)} \left(B_{P_\ell}(S_0, S \setminus R) + \max_{r \in R, t \in T} f_{\ell+1}(r,t) \right) \ . \end{aligned} \tag{17}$$

Combining (16), (17) and the induction hypothesis implies (14), and completes the proof of the induction step.

Combining (14) and (13) provides (12), which terminates the proof of the lemma. ◀

The equivalence between (i) and (ii) is established on the basis of Lemma 48. Assume (i) holds, let N be a large integers, X a set of N tokens and let $\overline{w}_0, a_1, \overline{w}_1, a_2, \ldots, a_\ell, \overline{w}_\ell$ a path in $\overline{W}$ from $\overline{w}_0[N]$ to a final configuration $\overline{w}_\ell$. Make use of the round-up function ϕ as defined in (2). For $i \in 1 \ldots n$, let $f_i \in \{0, 1, \infty\}^{S^2}$ which counts the number of tokens per edge, and round it up using ϕ, defined by:

$$f_i(s,t) = \phi(|\{x \in S \mid (\overline{w}_{i-1}(x), \overline{w}_i(x)) = (s,t)\}|) \ .$$

Since the largest constant in $\overline{W}$ is 1 then $\mathrm{dom}(f_i) \in \overline{W}$ and the flow f_i is an action flow in $\overline{W}$. Thus $\overline{w}_0, \overline{w}_1, \ldots, \overline{w}_\ell$ is a path through the pipeline $P = (f_1, \ldots, f_\ell)$. According to Lemma 48, and since the image of ψ is in the tropical cut semigroup $\mathcal{C}_\mathcal{M}$, the condition (ii) is satisfied. Conversely, assume (ii) holds. Since action flows form the generator basis of $\mathcal{C}_\mathcal{M}$, then according to Lemma 48, the capacity of pipelines is unbounded, and according to the max-flow min-cut theorem, then (i) holds.

The equivalence between (ii) and (iii). Elements of Z belong to the tropical semiring $\mathcal{M}$ thus $Z \subseteq \mathbb{N} \cup \{\infty\}$. There are two cases.

Assume first $\infty \in Z$. Then by definition of the tropical mincut semigroup, there exists a sequence of action flows $f_1, f_2, \ldots, f_n \in \mathcal{F}$ such that the product $M_0 = M(f_1) \cdot M(f_2) \cdot \ldots \cdot M(f_n)$ in $\mathcal{C}_\mathcal{M}$ satisfies $\forall s \in S_0, M_0(\{s_0\}, S \setminus F) = \infty$. Then by duality, $\forall s \in S_0, \exists f \in F, (f_1 \cdot f_2 \cdot \ldots \cdot f_n)(s_0, f) = \infty$. Denote $M = M(f_1) \cdot M(f_2) \cdot \ldots \cdot M(f_n) \in \mathcal{C}_R$. Then $\min_{s \in S_0} M(\{s_0\}, S \setminus F) = \infty$. Conversely, if there exists $M \in \mathcal{C}_R$ such that $\min_{s \in S_0} M(\{s_0\}, S \setminus F) = \infty$, then the corresponding sequence of generators produce an element in $\mathcal{C}_\mathcal{M}$ and $\infty \in Z$.

Apply the main result in [16] to our semigroups.

▶ **Theorem 51.** *[16, Theorem 12] Let $S_0, T \in \mathcal{P}(S)$. The following statements are equivalent:*
(a) *the integer-valued coefficients of cuts at coordinates (S_0, T) are unbounded, i.e., $\{M(S_0, T) \mid M \in \mathcal{C}_{\mathcal{M}}\}$ is infinite;*
(b) *there is a cut $M \in \mathcal{C}_R$ whose coefficient at the coordinate (S_0, T) is ω, i.e., $\exists M \in \mathcal{C}_R, M(S_0, T) = \omega$.*

This shows the equivalence between (ii) and (iii) in case Z does not contain ∞. ◀

J Duality: proof of Lemma 25

The core of the proof is as follows.

▶ **Lemma 52** (dual pairs and their properties)**.** *For every flow $f \in \bar{\mathcal{R}}^{S^2}$ and cut $M \in \mathcal{R}^{\mathcal{P}(S)^2}$,*

$$f_{M(f)} = f \tag{18}$$

$$M(f_M) = M \tag{19}$$

A pair (f, M) which satisfy $f = f_M$ and $M = M(f)$ is called a dual pair*. Every dual pair (f, M) satisfies several properties.*

- monotonicity*: for $S_0, S_1, T_0, T_1 \in \mathcal{P}(S)$,*

$$(S_0 \subseteq S_1 \wedge T_0 \subseteq T_1) \implies (M(S_0, T_1) \leq M(S_1, T_0)) \ . \tag{20}$$

- atomicity*: for $S_0, T \in \mathcal{P}(S)$,*

$$M(S_0, T) = \max_{s \in S_0} M(\{s\}, T) = \max_{t \in S \setminus T} M(S_0, S \setminus \{t\}) = \max_{\substack{s \in S_0 \\ t \in S \setminus T}} M(\{s\}, S \setminus \{t\}) \ . \tag{21}$$

- compatibility *with semigroup operations: for every dual pairs $(f_1, M_1), (f_2, M_2), e, E)$, such that $E = E^2$ and $e = e^2$,*

$$M(f_1 \cdot f_2) = M_1 \cdot M_2 \tag{22}$$

$$f_{M_1 \cdot M_2} = f_1 \cdot f_2 \tag{23}$$

$$M\left(e^{\sharp}\right) = M(e)^{\sharp} \tag{24}$$

$$f_{E^{\sharp}} = f_E^{\sharp} \ . \tag{25}$$

Proof. Equality (18) holds because for every $f \in \mathcal{R}^{S^2}$ and $s, t \in S$, $f_{M_f}(s, t) = M(f)(\{s\}, S \setminus \{t\}) = f(s, t)$. Equality (19) holds because for every $M \in \mathcal{R}^{\mathcal{P}(S)^2}$ and $S_0, T \in \mathcal{P}(S)$,

$$\begin{aligned} M(f_M)(S_0, T) &= \max_{\substack{s \in S_0 \\ t \in S \setminus T}} f_M(s, t) = \max_{\substack{s \in S_0 \\ t \in S \setminus T}} M(\{s\}, S \setminus \{t\})) \\ &= \max_{\substack{s \in S_0 \\ t \in S \setminus T}} M(\{s\}, S \setminus \{t\})) = M(S_0, T) \end{aligned}$$

Monotonicity follows from the definition of $M(f)$ (Definition 3). Atomicity (21) holds since $M(S_0, T) = \max_{s \in S_0, t \in S \setminus T} f(s, t) = \max_{s \in S_0, t \in S \setminus T} M(s, t)$.

Compatibility with operations, starting with (22). Let $s_0, s_2 \in S$. Denote $x = (f_1 \cdot f_2)(s_0, s_2)$ and let $r \in S$ such that $x = \min(f_1(s_0, r), f_2(r, s_2))$. For every $S_1 \in \mathcal{P}(S)$, either $r \notin S_1$, in which case $M_1(\{s_0\}, S_1) \geq x$ or $r \in S_1$, in which case $M_2(S_1, S \setminus \{s_2\}) \geq x$. Thus

$$(M_1 \cdot M_2)(\{s_0\}, \{s_2\}) \geq x \ . \tag{26}$$

Let $S_1 = \{s_1 \in S \mid f_1(s_0, s_1) > x\}$. Then

$$M_1(\{s_0\}, S_1) = \max_{s \in S \setminus S_1} f_1(s_0, s) \leq x \ . \tag{27}$$

For every $s_1 \in S_1$, $f_1(s_0, s_1) \leq x$ because $x = (f_1 \cdot f_2)(s_0, s_2) \geq \min(f_1(s_0, s_1), f_2(s_1, s_2))$. Thus

$$M_2(S_1, S \setminus \{s_2\}) \leq x \ . \tag{28}$$

Combining (27) and (28) gives

$$(M_1 \cdot M_2)(\{s_0\}, S \setminus \{s_2\}) \leq x \ . \tag{29}$$

hence according to (26),

$$(M_1 \cdot M_2)(\{s_0\}, S \setminus \{s_2\}) = x = (f_1 \cdot f_2)(s_0, s_2) \ .$$

This proves (22) for atomic coordinates, and the general case of (22) follows by atomicity. Combined with (18), this gives (23).

Towards (24), let (e, E) be a dual pair. According to the compatibility with multiplication, $e = e^2 \iff E = E^2$. Let $S_0, T \in \mathcal{P}(S)^2$. We show

$$E^\sharp(S_0, T) = M(e^\sharp(S_0, T)) \ . \tag{30}$$

Start with the (easy) case $E(S_0, T) \neq 1$. Then by definition of the iteration $E^\sharp(S_0, T) = E(S_0, T)$. By atomicity of E, $E(S_0, T) = \max_{s \in S_0, t \in S \setminus T} e(s, t)$. There are two subcases. The first subcase is $E(S_0, T) = 0$ in which case (30) holds because $\forall s \in S_0, t \in S \setminus T, e(s, t) = 0$ thus $\forall s \in S_0, t \in S \setminus T, e^\sharp(s, t) = 0$ and $M(e^\sharp(S_0, T)) = \max_{s \in S_0, t \in S \setminus T} e^\sharp(s, t) \leq \max_{s \in S_0, t \in S \setminus T} e^\sharp(s, t) = 0$. The second subcase is $E(S_0, T) = \omega$ in which case (30) holds as well because $E(S_0, T) = \max_{s \in S_0, t \in S \setminus T, e(s,t) \geq \omega} e(s, t) = \max_{s \in S_0, t \in S \setminus T, e^\sharp(s,t) \geq \omega} e^\sharp(s, t) = E^\sharp(S_0, T)$.

The proof of (30) in case $E(S_0, T) = 1$ is the following lemma.

▶ **Lemma 53.** *Let $S_0, T \in \mathcal{P}(S)$ such that $E(S_0, T) = 1$. The following properties are equivalent.*

a) *$E^\sharp(S_0, T) = 1$;*

b) *all edges $(s, t), s \in S_0, t \in S \setminus T$ such that $e(s, t) = 1$ are stable in e.*

Proof. Assume that (a) holds. Let $R \in \mathcal{P}(S)$ which witnesses the stability of R i.e. $E(R, R) = 0$ and $\max(E(S_0, R), E(R, T)) = 1$. Let $s \in S_0, t \in S \setminus T$ such that $e(s, t) = 1$. We show by case analysis that

$$e^\sharp(s, t) = 1 \ , \tag{31}$$

i.e. (s, t) is stable in e. By contradiction, assume (s, t) is unstable in e i.e. there exists s_0, t_0 such that $e(s, s_0) \geq \omega \wedge e(s_0, t_0) = 1 \wedge e(t_0, t) \geq \omega$. Since $s \in S_0 \wedge e(s, s_0) \geq \omega \wedge E(S_0, R) \leq 1$ then $s_0 \in R$. Since $s_0 \in R \wedge e(s_0, t_0) = 1 \wedge E(R, R) = 0$ then $t_0 \in R$. Since $t_0 \in R \wedge e(t_0, t) \geq \omega \wedge E(R, T) \leq 1$ then $t \in T$, a contradiction with the initial choice of $t \in S \setminus T$. That shows (31). Hence (b) holds.

Conversely, assume that (b) holds. We show that

$$R = \{t_0 \in S, \exists s_0 \in S, e(s, s_0) \geq \omega \wedge (s_0 = t_0 \vee e(s_0, t_0) \geq 1)\}$$

witnesses the stability of (S_0, T) in E.

We prove first that

$$E(R, R) = 0 \ . \tag{32}$$

The reason is R is closed by successors through nonzero edges of E. Let $t_0 \in R$ and $t_1 \in S$ such that $e(t_0, t_1) \geq 1$. By definition of R, there exists $s_0 \in S$ such that $e(s, s_0) \geq \omega$ and $(s_0 = t_0 \vee e(s_0, t_0) \geq 1)$. In the subcase $s_0 = t_0$ then $e(s_0, t_1) = e(t_0, t_1) \geq 1$. In the subcase $e(s_0, t_0) \geq 1$ then $e(s_0, t_1) \geq \min(e(s_0, t_0), e(t_0, t_1)) \geq 1$ In both subcases s_0 witnesses that $t_1 \in R$. This completes the proof of (32).

Moreover

$$E(S_0, R) \leq 1 \tag{33}$$

because R contains all ω-successors and ∞-successors of S_0 through e.

Finally show by contradiction

$$E(R, T) \leq 1 \ . \tag{34}$$

A contrario, assume $E(R, T) \geq \omega$. By atomicity, there exists $t_0 \in R$ and $t \in S \setminus T$ such that $e(t_0, t) \geq \omega$. By definition of R, there exists $s_0 \in S$ such that $e(s, s_0) \geq \omega$ and $s_0 = t_0 \vee e(s_0, t_0) \geq 1$. A step towards the contradiction is to show that

$$e(s, t) = 1 \wedge s_0 \neq t_0 \wedge e(s_0, t_0) = 1 \ . \tag{35}$$

First, $e(s, t) \leq E(S_0, T) \leq 1$, by atomicity of E. There are two subcases, depending on which clause does hold in the disjunction $(s_0 = t_0 \vee e(s_0, t_0) \geq 1)$. The subcase $s_0 = t_0$ cannot hold because we would have $e(s, t) \geq \min(e(s, s_0), e(s_0 = t_0, t)) \geq \omega$, a contradiction with $e(s, t) \leq 1$, supra. So the other subcase $e(s_0, t_0) \geq 1$ holds, and $1 \geq e(s, t) \geq \min(e(s, s_0), e(s_0, t_0), e(t_0, t)) \geq \min(\omega, e(s_0, t_0), \omega) \geq 1$. That completes the proof of (35).

Finally, we have $e(s, s_0) \geq \omega$ and $e(s_0, t_0) = 1$ and $e(t_0, t) \geq \omega$, which contradicts the stability of (s, t) in e. This completes the proof of (34) by contradiction.

Combining (34) with (32) gives $\max(E(S_0, R), E(R, T)) \leq 1$, and this is actually an equality since $\max(E(S_0, R), E(R, T)) \geq E(S_0, T) \geq e(s, t) = 1$. Combined with (32), this witnesses the stability of (S_0, T) in E. ◀

Now we get to complete the proof of (30), remember we have already processed the easy case $E(S_0, T) \neq 1$ thus we can assume $E(S_0, T) = 1$. By definition of the iteration, $E^\sharp(S_0, T) \in \{1, \omega\}$. The first case is $E^\sharp(S_0, T) = 1$. According to Lemma 53, $M(e^\sharp)(S_0, T) = \max_{s \in S_0, t \in S \setminus T} e^\sharp(s_0, t) = \max_{s \in S_0, t \in S \setminus T} e(s_0, t) = E(S_0, T) = E^\sharp(S_0, T)$. The second case is $E^\sharp(S_0, T) = \omega$. According to Lemma 53, there is $s \in S_0$ and $t \in S \setminus T$ such that $e^\sharp(s, t) = \omega$. Thus $M(e^\sharp)(S_0, T) \geq \omega$. Since $M(e^\sharp)(S_0, T) \leq \max\{\omega, M(e)(S_0, T)\}$ then $M(e^\sharp)(S_0, T) = \omega$. We have processes both possible cases.

Finally, (30) has been proved in all cases, hence (24) holds. To get (25), set $e = f_E$, so that $E = M(e)$ and apply (24) to get $f_{E^\sharp} = f_{M(e)^\sharp} = f_{M(e^\sharp)} = e^\sharp = f_E^\sharp$. ◀

Proof of Lemma 25. The generators of $\mathcal{F}_R$ and $\mathcal{C}_R$ are duals of other. Both the inclusions $\mathcal{F}_R \subseteq \mathcal{C}_R$ and $\mathcal{C}_R \subseteq \mathcal{F}_R$ follow from the compatibility with respect to product and iteration (22)–(25), and the inductive definition of both semigroups with respect to those two operations. ◀