Gadgets and Gizmos: A Formal Model of Simulation in the Gadget Framework for Motion Planning

by

Dylan Hendrickson

B.S., Massachusetts Institute of Technology (2019)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2021

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2021

Certified by .....
Erik D. Demaine
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Gadgets and Gizmos: A Formal Model of Simulation in the Gadget Framework for Motion Planning

by

Dylan Hendrickson

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2021, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science and Engineering

Abstract

Recent work has developed a theory of motion-planning gadgets, which are a useful tool for proving hardness for a variety of problems that can be thought of in terms of an agent navigating a dynamic environment. We introduce formal objects representing motion-planning gadgets, which we call *gizmos*, and ask which gizmos simulate each other—simulations between gizmos yield reductions between natural decision problems, so this has consequences for complexity.

We define several classes of gizmos, and prove that they are closed under simulation: gizmos with some property cannot simulate gizmos without it. For several of these classes, we also find a gizmo which can simulate every gizmo in the class; this is analogous to completeness for a complexity class. We consider gizmo simulation and prove unsimulability and universality in two restricted settings: planar simulation, where the simulation must embed in the plane without crossings, and input/output simulation, which is a model for fully deterministic settings. We mostly focus on simulations with finitely many gizmos and gizmos with finitely many states (called *regular*), but many of our results carry over to more exotic infinite gizmos.

Thesis Supervisor: Erik D. Demaine
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I'd like to thank my advisor, Erik Demaine, as well as Jayson Lynch, for introducing me to the study of gadgets and helping familiarize me with computer science research more generally. I'd like to also thank my various collaborators, including Joshua Ani, Josh Brunner, Lily Chung, Yevhenii Diomidov, and Linus Hamilton, for inspiring conversations and contributions about gadgets and many related topics. Yevhenii and Joshua in particular have contributed significantly to the recennt work on gadgets, including many results in this work; Yevhenii has convinced me multiple times that the core definitions were wrong and invented closed gizmo classes, and Joshua has provided the core ideas behind several universality proofs. Finally, I'd like to thank everyone, including those just mentioned, who has helped foster the inclusive and collaborative community in which I have developed this work.

Contents

Chapter 1

Introduction

1.1 History of gadgets

There is a long tradition of structuring reductions, typically those in hardness proofs, around 'gadgets.' Informally, a gadget in such a reduction is a local component corresponding to a portion of the input; the output instance is constructed by replacing each part of the input with the appropriate gadget. For instance, reductions from 3SAT often consist of 'clause gadgets' and 'variable gadgets' [Hol81, GLN14, JG79].

Szabó [Sza09] attributes the use of gadgets in reductions to Tutte [Tut54], who uses a local replacement to reduce finding f-factors of graphs to the special case of finding perfect matchings. This graph-theoretic type of gadget has been studied in detail: for instance, Loebl [Loe93] gives a characterization of exactly the degree constraints which can be enforced with gadgets in reductions from a generalization of finding f-factors to the problem of partitioning a graph into edges and designated triangles. Trevisan et al. [TSSW00] consider a more general class of gadgets in reductions between constraint satisfaction problems, and demonstrate a powerful way to understand the limits of these gadgets based on linear programming. They prove bounds on the efficiency of gadgets in hardness of approximation reductions, and use computer search to achieve many of these bounds.

Much of the work involving gadgets, and particularly the study of gadgets themselves, has focused on the kinds of gadgets used for reductions between constraint satisfaction problems. In particular, these gadgets are generally useful only for reductions between problems in NP. However, there are also plenty of hardness proofs for PSPACE and other classes which are based around gadgets in a similar way [ADGV15, DDHO03, FG87, HS04]. The general strategy for proving hardness has two parts: first build some gadgets in the problem of interest, and then use these gadgets to construct an instance of a known-hard problem.

In 2018, Demaine, Grosof, et al. [DGLR18] introduced a notion of 'gadget' for another broad class of problems: motion planning problems. In a motion planning problem, an 'agent' must navigate an environment which changes in response to the agent's actions. This kind of problem is too dynamic for constraint-satisfaction-style gadgets to apply, and often naturally lives in PSPACE. A motion-planning gadget

has some 'states,' some 'locations,' and some 'transitions,' which say that the agent is allowed to move from one location to another while switching the state from one to another.[1] Understanding these gadgets can simplify many hardness proofs, by preemptively performing the step of showing that navigating the gadgets is hard. All that needs to be done to prove a problem related to motion planning hard is the construction of a known-hard set of gadgets.

For instance, Demaine et al. [DDHO03] build four simple gadgets in a block-pushing problem, and then show that these gadgets suffice for NP-hardness through a fairly involved reduction from 3-coloring graphs. However, it turns out that one of the four gadgets alone is sufficient for NP-hardness [ABD$^+$20]. Holzer and Schwoon [HS04] build a gadget they call a 'catalyst chamber' and some auxiliary gadgets in the game Atomix, and use these to simulate finite automata. However, the catalyst chamber can easily be used to build a gadget called 'symmetric self-closing doors' or 'mismatched dicrumblers' (see Figure 2-5), which alone is PSPACE-complete [ABD$^+$20].[2] This example is also interesting because in Atomix, the player can make moves anywhere in the game; it does not have an 'agent' in the sense of motion-planinng gadgets. The reduction forces some amount of locality which allows Atomix to simulate a motion-planning agent.

Since Demaine, Grosof, et al.'s work in 2018 [DGLR18], there has been significant progress in understanding the complexity of motion-planning gadgets. Demaine, Hendrickson, and Lynch [DHL20] characterized the complexity of multiple decision problems for two large classes of gadgets. Ani, Demaine, et al. [ADHL20] considered a fully determinstic gadget model, and proved hardness of many gadgets within it. Ani, Bosboom, et al. [ABD$^+$20] showed additional gadgets PSPACE- and NP-complete, particularly in planar settings. Jayson Lynch's thesis [Lyn20] synthesizes many of these results and extends them in a handful of ways.

While the work on motion-planning gadgets has primarily focused on proving hardness, another natural question to ask is when gadgets can 'simulate' each other. Simulation is relevant even one is interested only in complexity, since a simulation between gadgets immediately gives a reduction between the corresponding decision problems. Several prior papers [ABD$^+$20, ADHL20, DHL20] have proven nontrivial results about gadget simulation, which we describe in Section 2.4.

1.2 Outline of this thesis

Gadget simulation is interesting enough to warrant its own investigation, independent of its original motivation in complexity theory. The goal of this work is to study the natural question:

Question 1.1. When is it possible to build some gadget out of copies of some other gadget?

[1] We describe motion-planning gadgets in more detail in Chapter 2

[2] This proof also applies to Ricochet Robots, which has independently been shown to be NP-hard [EK06], and perhaps other games with similar mechanics, such as Pete's Pike, which is also PSPACE-complete [Mey16].

Before we can begin to answer this question, we need to say what it means to build a gadget out of other gadgets; this is gadget simulation, which has not been defined carefully before. Our approach involves first defining gadgets in a completely new way.

In Chapter 2, we define gadgets and gadget-related terminology as they have been defined in past work: as state machines. The only new contribution in this chapter is the definition of a family of decision problems we call 'targeted (set) reconfiguration.'

In Chapter 3, we define the objects we will study, called 'gizmos.' Gizmos are a formal representation of gadgets which more easily yield a natural notion of simulation than the traditional state machines. As discussed in detail in Sections 3.1 and 3.5, gizmos are only applicable in the context of one-player targeted set reconfiguration, which includes one-player reachability as a special case. An important special class of gizmos, called 'prefix-closed' gizmos, capture one-player reachability. Another important special class, called 'regular' gizmos, capture gadgets with finitely many states.

In the remaining sections, we consider primarily regular prefix-closed gizmos. Our overarching goal is to determine when regular prefix-closed gizmos simulate other regular prefix-closed gizmos.

In Chapter 4, we prove several negative results. That is, we prove that various classes of gizmos are 'closed under simulation,' meaning gizmos in the class cannot simulate gizmos outside the class. The classes we consider include one infinite family of closed gizmo classes, called 'implication properties,' and a few other closed classes. Prefix-closed is an implication property, and thus prefix-closed gizmos are closed under simulation; this justifies studying specifically simulations between prefix-closed gizmos. Regular gizmos are also closed under simulation, provided we require simulations to use only finitely many gizmos. Several of our classes correspond to natural classes of gadgets which have been defined in past work [DGLR18, DHL20, Lyn20].

In Chapter 5, we prove several general positive results. These take the form of showing that a specific small set of (regular, prefix-closed) gizmos simulates every regular prefix-closed gizmo in some class.

In Chapter 6, we consider simulations which are allowed to have infinitely many gizmos, and nonregular gizmos, which are allowed to have infinitely many states. Several of our closure and universality results carry over to this situation.

Finally, in Chapter 7, we pose many remaining open problems and potential avenues to explore which are not covered by this thesis.

1.3 Summary of results

Here we collect and organize most of the results in this thesis, since they are spread across many sections.

Class	Def.	Closed under?		Universal gadget	Arb?
		Arb. sim.	Trav.		
Implication properties	4.5	✓4.6			
All (prefix-closed)	3.9		✓4.2	Door 5.8	✓6.3
$X \implies XX$ & $XY \implies YX$			✓5.17	Mutually closing diodes 5.19	
Unchanging	5.20		✓5.21	Diode 5.23	✓6.5
$X \implies X^{-1}$ unchanging				Wire 5.24	✓6.6
Reversible	6.7	✓6.8	✗3.26	2-toggle 5.10	✓6.10
Strictly bounded	4.11	✗4.17	✓4.12	Ordered dicrumblers 5.14	
Weakly bounded	4.14	✗4.17	✓4.15	Brittle door 5.15	
Balanced	4.18	✗4.20	✓4.21		

Figure 1-1: Table of gizmo classes. All classes listed are closed under finite simulation. "Def." points to the definition of each class. "Arb. sim." and "Trav." stand for arbitrary simulation and traversals. "Universal gadget" describes a set of gizmos—one representing each state of the gadget—which is universal for the regular prefix-closed gizmos in the class. "Arb?" indicates results that extend to arbitrary simulation and nonregular gizmos, which is only meaningful for classes closed under arbitrary simulation. We include references to the relevant results.

Gadget	Implication properties							Reversible	Bnded.		Balanced	Univ.	Arb?
	$Y \Rightarrow XX^{-1}Y$	$XYZ \Rightarrow XYY^{-1}YZ$	$XX \Rightarrow X$	$YX \Rightarrow XY$	$XY \Rightarrow Y$	$XY \Rightarrow X, Y$	$X \Rightarrow X^{-1}$		Strictly	Weakly			
1-toggle 2-1	✓	✓						✓			✓		
Dicrumbler 2-2					✓				✓	✓	✓		
Door 2-3												5.8	✓
Self-closing door 2-4												5.9	✓
Mism. dicrumb.s 2-5												5.9	✓
2-toggle 2-6	✓	✓						✓				5.10	✓
Tripwire-lock 2-7	✓	✓						✓				5.11	✓
Locking 2-toggle 2-8	✓	✓						✓			✓		
Ord. dicrumb.s 2-9									✓	✓	✓	5.14	
Brittle door 2-10										✓		5.15	
Mut. cl. diodes 2-11			✓	✓	✓					✓		5.19	
Diode 2-12			✓	✓	✓	✓				✓		5.23	✓
Wire 2-13	✓	✓	✓	✓	✓	✓	✓	✓		✓		5.24	✓

Figure 1-2: Table of gadgets. A check indicates that every (prefix-closed) gizmo corresponding to a state of the gadget with full target set satisfies the column's property. Each column is closed under simulation: a gadget can only simulate gadgets with at least as many checks as it. "Univ." indicates that the gadget is universal for the class defined by all of its checks, and "Arb?" indicates the universality results which extend to arbitrary simulation and nonregular gizmos. All gizmos here are prefix-closed and regular.

Chapter 2

Traditional description of motion-planning gadgets

We now present motion-planning gadgets as they have been considered in previous work [AAD+20, ABD+20, ADHL20, DGLR18, DHL20, Lyn20]. In Chapter 3 we will introduce a new, more formal, description.

A gadget consists of some *states*, some *locations*, and some *transitions* of the form $(s, l) \to (s', l')$ where s and s' are states and l and l' are locations. Generally there are finitely many states and locations. The transition $(s, l) \to (s', l')$ is taken to mean that when the gadget is in state s, the agent can enter at l, leave at l', and put the gadget in state s'.

A gadget can be conveniently described by drawing a *state diagram*, which shows the transitions available in each state labeled with the state they go to. For instance, Figure 2-1 shows a state diagram for a gadget called the *1-toggle*, introduced by Demaine, Grosof, et al. [DGLR18]. The 1-toggle has two locations and two states, numbered 1 and 2. In each state, the gadget can be traversed in one direction to switch to the other state. More explicitly, if we call the locations A and B, the 1-toggle has transitions $(1, A) \to (2, B)$ and $(2, B) \to (1, A)$. This can be thought of as a directed tunnel which flips direction every time it is traversed.

Gadgets are combined into *systems* or *networks* by placing copies of them near each other and connecting the locations. Different prior papers [DGLR18, DHL20] have described the process of connecting locations in different but equivalent ways. We then imagine an agent, or sometimes multiple agents, moving around this network of gadgets. When an agent arrives at a gadget, it can make a transition through the gadget, changing its state. This is an abstraction for various problems involving agents

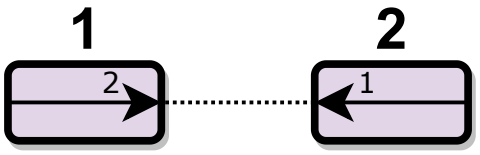

Figure 2-1: A state diagram for the 1-toggle.

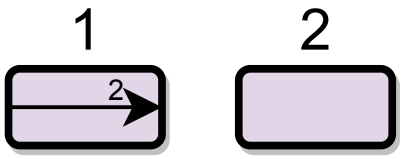

Figure 2-2: A state diagram for the directed crumbler.

navigating a dynamic environment; we ignore the details of the environment and only consider what motions are allowed and how they change the abstract state. Prior work on gadgets [AAD+20, ABD+20, ADHL20, DGLR18, DHL20, Lyn20, ADG+21], as well as related work which does not explicitly use the motion-planning gadget framework [ADGV15, Vig14, DVW16, DDHO03, HS04], includes many applications of problems which can be understood in this abstraction.

Often, particularly for applications in two dimensions, it is worthwhile to consider *planar* networks of gadgets. A network of gadgets is planar if it can be drawn in the plane without crossing; equivalently, when an appropriate graph constructed from the network is planar (see Demaine, Hendrickson, and Lynch [DHL20] or Lynch [Lyn20] for details).

2.1 Examples of gadgets

We will now meet a cast of gadgets, many of which were introduced in prior work [DGLR18, DHL20, ABD+20]. We have already seen the 1-toggle (Figure 2-1). Most of the gadgets here are *on tunnels*, meaning the locations can be partitioned into pairs $\{a_i, b_i\}_i$ called *tunnels* such that every transition is between a_i and b_i for some i.

When we draw gadgets elsewhere in this thesis, we will say which gadgets those shown are if it is at all ambiguous. For some gadgets, we use special notation or colors to indicate the gadget type and state.

The *directed crumbler* or *dicrumbler*, whose state diagram is shown in Figure 2-2, has 2 locations and 2 states. It has a directed tunnel, which 'crumbles' when used.

The *door*, whose state diagram is shown in Figure 2-3, has 6 locations and 2 states. There are three directed tunnels, called the *open*, *close*, and *door* tunnels from top to bottom. The open and close tunnels are always available, and the door is available in state 1. Traversing the open tunnel opens the door by switching to state 1, and traversing the close tunnel closes the door by switching to state 2.

The *self-closing door*, whose state diagram is shown in Figure 2-4, has 3 locations and 2 states. The bottom tunnel, called the *door*, is directed and available only in state 1, and switches to state 2 when traversed (hence 'self-closing'). The top location is the *button*, and it can be visited to reset the state to 1, reopening the door.

The *mismatched dicrumblers*, whose state diagram is shown in Figure 2-5, has 4 locations and 2 states. The agent must alternate which tunnels it takes, and each tunnel is directed. This gadget is composed of two dicrumblers, which are 'mismatched' in that they have opposite state. The mismatched dicrumblers is called the 'symmetric self-closing door' by Ani, Bosboom, et al. [ABD+20].

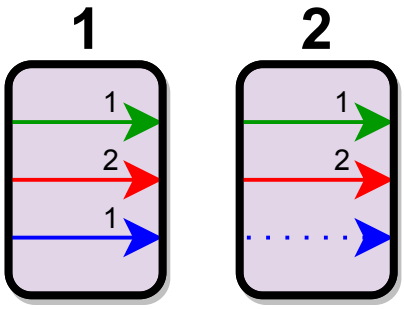

Figure 2-3: A state diagram for the door.

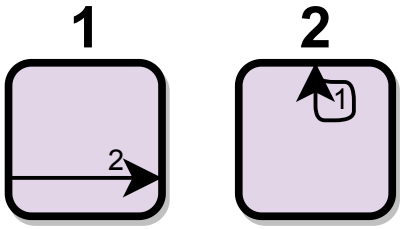

Figure 2-4: A state diagram for the self-closing door.

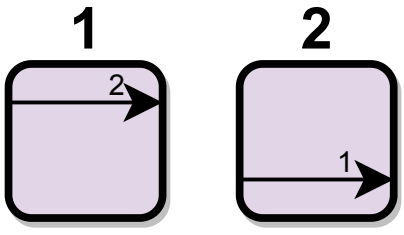

Figure 2-5: A state diagram for the mismatched dicrumblers.

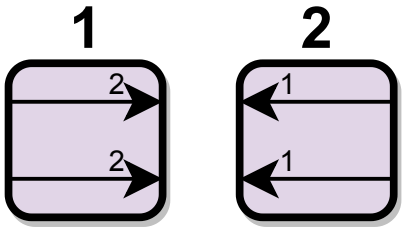

Figure 2-6: A state diagram for the 2-toggle.

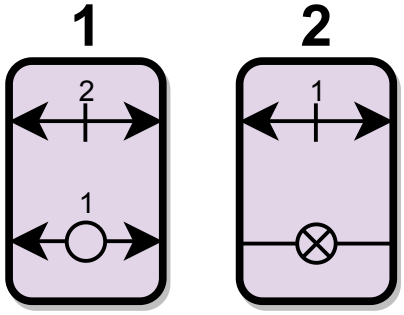

Figure 2-7: A state diagram for the tripwire-lock.

The *2-toggle*, whose state diagram is shown in Figure 2-6, has 4 locations and 2 states. It can be thought of as two 1-toggles sharing a state: traversing either 1-toggle flips them both. Explicitly, if the locations are labeled A through D, it has transitions $(1, A) \to (2, B)$, $(1, C) \to (2, D)$, $(2, B) \to (1, A)$, and $(2, D) \to (1, C)$. In the remaining examples we will not list the transitions explicitly, and trust the reader to be able to generate them from the state diagram.

The *tripwire-lock*, whose state diagram is shown in Figure 2-7, also has 4 locations and 2 states. The top tunnel, called the *tripwire*, is always traversable and toggles the state of the gadget whenever it is traversed. The bottom tunnel, called the *lock*, is traversable in both directions, or *open*, in state 1, but not traversable, or *closed*, in state 2. We will generally omit the arrows when drawing the tripwire-lock; the state is indicated by the presence or absence of an `X` on the lock.

The *locking 2-toggle*, whose state diagram is shown in Figure 2-8, has 4 locations and 2 states. It is similar to a 2-toggle, except that after traversing a tunnel, the agent must come back across the same tunnel—the other tunnel is closed until this happens.

The *ordered dicrumblers*, whose state diagram is shown in Figure 2-9, has 4 locations and 2 states. The top tunnel can be traversed, and then the bottom tunnel, and then there are no legal transitions. This is like the mismatched dicrumblers except that it can only be used twice.

The *brittle door*, whose state diagram is shown in Figure 2-10, has 6 locations and 3 states. It can be thought of as a door which is destroyed the first time the door is traversed. We use the same names for the tunnels as in the door.

The *mutually closing diodes*, whose state diagram is shown in Figure 2-11, has 4

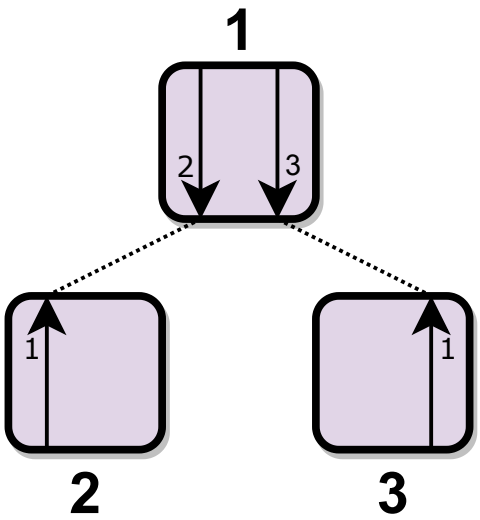

Figure 2-8: A state diagram for the locking 2-toggle.

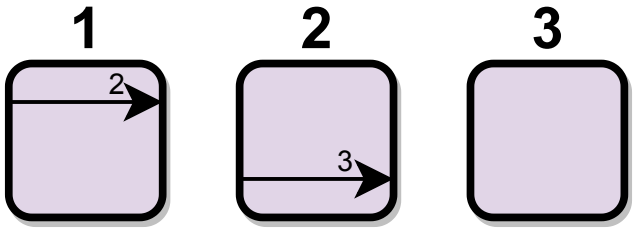

Figure 2-9: A state diagram for the ordered dicrumblers.

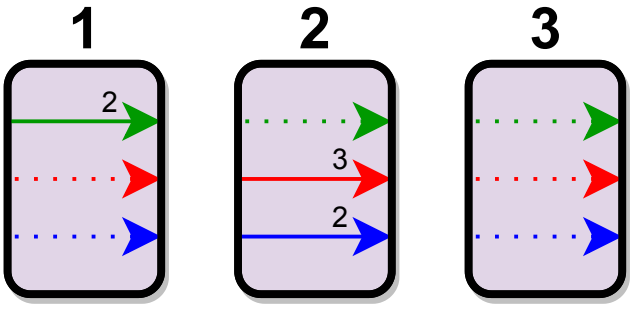

Figure 2-10: A state diagram for the brittle door.

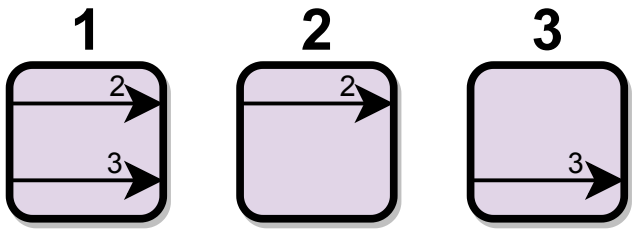

Figure 2-11: A state diagram for the mutually closing diodes.

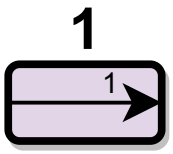

Figure 2-12: A state diagram for the diode.

locations and 3 states. Either directed tunnel can be traversed freely, but once one is traversed the other cannot be.

The *diode*, whose state diagram is shown in Figure 2-12, has 2 locations and 1 state. It can be traversed in only one direction.

The *wire*, whose state diagram is shown in Figure 2-13, has 2 locations and 1 state, and can be traversed freely. The wire is almost a trivial gadget, and can be constructed in almost every situation where gadgets are applied. It will be a useful example to understand our formalization of gadgets.

2.2 Gadget decision problems

We now introduce several decision problems related to gadgets which have been considered. These decision problems have been studied with the goal of finding reductions to problems outside of gadgets, so they are chosen to be convenient to reduce from. All of these decision problems are parameterized by set of gadgets S, which is usually finite and often a singleton.

The simplest decision problem, and the one our formalization is intended for, is *(one-player) reachability*. Reachability with S asks: given a network of gadgets from S and a specified *start location* and *target location*, can an agent get from the start location to the target location?

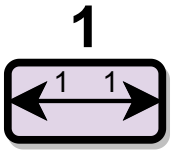

Figure 2-13: A state diagram for the wire.

Reachability is the natural gadget decision problem to reduce from for any problem involving trying to reach a location. This is the decision problem that has been the focus of all prior work on gadgets [AAD+20, ABD+20, ADHL20, DGLR18, DHL20, Lyn20].

Reconfiguration with S asks: given a network of gadgets from S, a start location, and a *target configuration* consisting of a specified state for each gadget in the network, can an agent at the start location put all gadgets simultaneously in their target states? Lynch [Lyn20] determines the complexity of reconfiguration for several gadgets.

We introduce a few variations on reconfiguration. *Targeted reconfiguration* with S asks: given a network of gadgets from S, a start location, a target location, and a target configuration, can an agent at the start location reach the target location and leave the network in the target configuration? This is essentially solving reachability and reconfiguration at the same time. For many gadgets, targeted reconfiguration is not significantly different from reconfiguration, and in particular most complexity results for reconfiguration (including all from Lynch [Lyn20]) apply equally to targeted reconfiguration.

Set reconfiguration and *targeted set reconfiguration* generalize reconfiguration and targeted reconfiguration. Instead of specifying a single state for each gadget in the network, the problem gives a *target set* of states for each gadget, and asks whether the agent can reach any configuration where each gadget is in a state in its target set. If each target set is a singleton, this is equivalent to the non-set decision problem. If each gadget's target set contains all states of the gadget, targeted set reconfiguration is equivalent to reachability.

2-player reachability with S asks: given a network of gadgets from S, a start location for each of White and Black, and a target location for each of White and Black, can White win the following game? Starting with White, the two players alternate making a single transition in the network. A player wins when they reach their target location. The complexity of 2-player reachability is studied by Demaine, Hendrickson, and Lynch [DHL20].

Team imperfect information reachability is similar to 2-player reachability, but there are multiple players on the same team, and players only know the states of gadgets they can reach. See Demaine, Hendrickson, and Lynch [DHL20] or Lynch [Lyn20] for a full definition.

It is also common to consider these decision problems when restricted to planar networks of gadgets. We abbreviate these restricted decision problems as e.g. *planar (one-player) reachability*.

2.3 Gadget properties

In this section, we define some properties of gadgets which have been studied in prior work.

A gadget is *reversible* if every transition can be undone; that is, whenever there is a transition $(s, l) \to (s', l')$, there is also a transition $(s', l') \to (s, l)$.

A gadget is *deterministic* if for every state s and location l, there is at most one

transition of the form $(s, l) \to (s', l')$. Reversible deterministic gadgets are studied in several previous papers [DGLR18, DHL20, Lyn20].

A gadget is *DAG* if the directed graph (called the *state graph*) whose vertices are states of the gadget and which has an edge $s \to s'$ for each transition $(s, l) \to (s', l')$ is acyclic. DAG gadgets are studied by Demaine, Hendrickson, and Lynch [DHL20], and multiple generalizations of DAG gadgets ('LDAG' and 'eventually static' gadgets) are studied by Lynch [Lyn20].

A gadget is *on tunnels* if its locations can be matched into tunnels where transitions never go between tunnels (this was defined in 2.1). Most of the gadgets in prior work have been tunnel gadgets, since they are often the easiest to work with.

A gadget is *monotonically opening* if its traversability never decreases: that is, if $(s, l) \to (\cdot, l')$ is legal and $(s, \cdot) \to (s', \cdot)$ is legal, then there is some legal transition $(s', l) \to (\cdot, l')$.[3]

A gadget is *monotonically closing* if its traversability never increases: that is, if $(s', l) \to (\cdot, l')$ is legal and $(s, \cdot) \to (s', \cdot)$ is legal, then there is some legal transition $(s, l) \to (\cdot, l')$.

A gadget is *unchanging* if it is both monotonically opening and monotonically closing; this typically makes for fairly boring gadgets. Lynch [Lyn20] introduced and studied monotonically opening, monotonically closing, and unchanging gadgets.

Once we have a formal notion of gadgets, we will introduce definitions analogous to several of these properties in Chapters 4 and 5. Our formalism will not have a notion of deterministic gadgets; in some sense nondeterministic gadgets will be converted to equivalent deterministic gadgets (see Section 3.5.3). Of the other properties listed, all but on tunnels and monotonically opening and closing are closed under simulation, so they are of interest in this thesis.

2.4 Gadget simulation

Consider how an agent can navigate the network of gadgets in Figure 2-14, consisting of tripwire-locks (Figure 2-7) and 1-toggles (Figure 2-1).

The agent cannot enter at H or T because the locks are closed. If it enters at A, all it can do (other than exiting at A with nothing changed) is cross the 1-toggle $E \to F$, go around the loop, and return across the same 1-toggle. This flips the state of all four tripwire-locks; now the agent can exit at H, but the locks at A and M are closed. All other nontrivial movements through this network are similar.

One can see that this network then behaves exactly like a 2-toggle (Figure 2-6). In the configuration shown, the agent can traverse $A \to H$ or $M \to T$, flipping all four tripwire-locks in either case. Then the agent can traverse $H \to A$ or $T \to M$, flipping the tripwire-locks back. These correspond to states 1 and 2 of the 2-toggle. Because of this equivalence, it is natural to say that the tripwire-lock and 1-toggle 'simulate' a 2-toggle. In fact, the tripwire-lock simulates a 1-toggle,[4] so the tripwire-lock alone

[3] We use $\cdot$ an arbitrary and irrelevant value.

[4] Connect one end of the tripwire to one end of the lock, to send the agent through both tunnels in series.

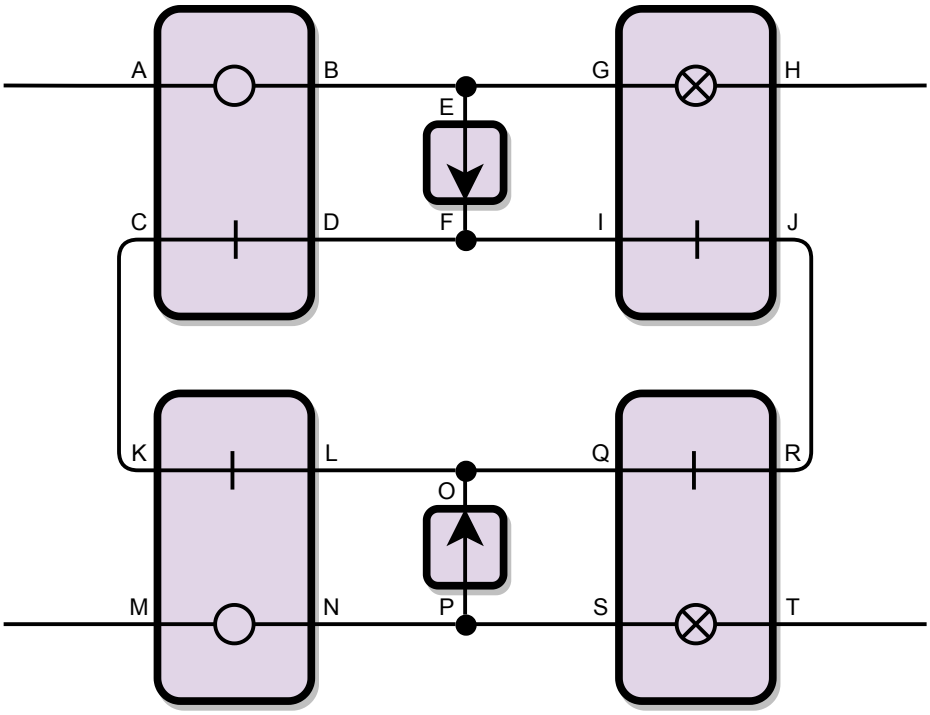

Figure 2-14: A simulation of a 2-toggle using tripwire-locks and 1-toggles, based on Figure 16 from Demaine, Grosof, et al. [DGLR18]. Gadgets' locations are labeled.

simulates a 2-toggle.

The goal of this thesis is to formalize and investigate this notion of simulation. In prior work [ABD+20, ADHL20, DGLR18, DHL20], gadget simulation has been considered, but never fully defined. We discuss in Chapter 3 some difficulties in defining simulation, justifying the decision not to do so. In any case, those papers were primarily interested in complexity, and all of their simulations have the properties related to reductions that they need. Our formalization will include all of the simulations in those papers, except for Ani, Demaine, et al. [ADHL20], which requires a different model of simulation.

Demaine, Grosof, et al. [DGLR18] shows that several 2-state 4-location reversible deterministic gadgets, including the 2-toggle and tripwire-lock, all simulate each other. This is true even in planar simulations, for every planar embedding of the gadgets in question. This thesis also showed that reversible gadgets can only simulate reversible gadgets (informally, without precisely defining reversible).[5]

Demaine, Hendrickson, and Lynch [DHL20] shows that every interacting[6] reversible deterministic gadget on tunnels simulates a locking 2-toggle. This is true even

[5]Demaine, Grosof, et al. [DGLR18] also prove that 'any system of gadgets composed of two deterministic reversible gadgets is deterministic and reversible.' This is not true in our formalization: their notion of 'system of gadgets' only allows locations to be connected in a (partial) matching, and connecting more than two locations requires an additional (nondeterministic) gadget called the 'branching hallway.' Our formalization allows connecting locations freely.

[6]Meaning some traversal on one tunnel changes the traversability of a different tunnel.

in planar simulation, and moreover every planar embedding of the locking 2-toggle can be simulated. They also prove the relatively simple fact that every nontrivial DAG gadget simulates either a directed or an undirected single-use tunnel—what we would call the dicrumbler or crumbler.

Ani, Demaine, et al. [ADHL20] shows that every so-called 'unbounded output-disjoint deterministic 2-state input/output gadget with multiple nontrivial inputs' simulates every so-called 'deterministic input/output gadget.' This was the first universality result in gadget simulation, meaning a result that some gadget simulates every gadget in some infinite class. This result uses a slightly different model of motion planning than the one we consider here—in particular, theirs is fully deterministic—and formalizing these simulations is beyond the scope of this thesis.

Finally, Ani, Bosboom, et al. [ABD$^+$20] shows that each of several gadgets, including the door, the mismatched dicrumblers, and the self-closing door, can simulate every gadget. This is true even in planar simulations, for every planar embedding except for one, called 'OTtocC.'

Both of these universality results from Ani, Demaine, et al. [ADHL20] and Ani, Bosboom, et al. [ABD$^+$20] rely on the assumption that the gadget being simulated has finitely many states. In our formalization, we call such gadgets *regular* (see Section 3.6). The latter result holds in our formalization when restricted to regular gadgets, and we will present a more formal proof in Section 5.2.

Chapter 3

Formalization of gadget simulation

3.1 Multiple notions of simulation

In different contexts, different understandings of 'simulation' are appropriate. For instance, consider a gadget with an optional opening; that is, an agent walking through the gadget can choose whether to open some other tunnel or leave it closed. In one-player reachability, the player has no reason to choose not to open the tunnel, so we can assume they always open it. For one-player reachability, it is reasonable to consider this as a simple simulation of a similar gadget where opening is not optional; more generally, we can ignore any ways the player may make things worse for themselves. On the other hand, for multiplayer decision problems, it may well be advantageous to choose not to open a tunnel, since one's opponent may benefit from the tunnel being open. Thus in multiplayer contexts we would not consider the optional opening gadget to simulate the forced opening gadget.

In this thesis, our focus is primarily on simulations in the context of one-player reachability, and our formal notion of gadgets is designed for this context. Our formalization actually captures simulations in the more general context of one-player targeted set reconfiguration. To leave open the option of introducing alternative formalizations for other contexts, and since our formal objects do not exactly correspond to gadgets,[7] we call these formal objects *gizmos*.

It is likely that our definition can be adapted to apply to situations such as multiplayer games and untargeted reconfiguration with relatively small changes. For multiplayer, we likely need to consider an appropriate notion of 'quantified traversal sequences' instead of traversal sequences. For reconfiguration, a difficulty arises where the agent may end inside a simulation of a gadget, though we do not allow the agent to end inside a gadget; targeted reconfiguration avoids this by specifying the location the agent must end. Studying simulation in these contexts is beyond the scope of this thesis.

[7] They correspond more closely, but not exactly, to states of gadgets.

3.2 Desirable properties

In order to motivate our definition of gizmos, we discuss some properties we would like them to have. Some of these considerations are similar to those faced by Bosboom in designing a representation of gadgets for an automated search for simulations [Bos20].

3.2.1 Simulations beget reductions

Gizmos are intended to represent gadgets in the context of one-player reachability problems. The question of whether a player can navigate a network of gadgets to reach a target location will become, in the language of gizmos, the question of whether the gizmo constructed by a particular simulation (corresponding to the gadget network) has a particular traversal (from the start location to the target location).

We have not yet defined gizmos and simulation, but we would like to use the following definition of targeted set reconfiguration. It turns out that the machinery of simulations gives a concise way to define this decision problem. A 'gizmo on $\{s, t\}$' will mean a gizmo with two locations s and t.

Definition 3.1. For a finite[8] set S of gizmos, *targeted set reconfiguration with S* is the following decision problem: Given a simulation of a gizmo on $\{s, t\}$ from S, does the simulated gizmo allow the traversal $s \to t$?

The input to targeted set reconfiguration with S is given by specifying which of the finitely many gizmos from S each gizmo in the simulation is a copy of; this takes constant space per gizmo for fixed S.

Each gizmo will encode the legal transitions, the initial state, and the target set of states for the gadget it represents. Reachability is simply targeted set reconfiguration where each target set is the entire set of states. This is a special case of the decision problem where we allow only gizmos with target set, which we call *prefix-closed* for reasons which will become clear.

The motivation for studying simulations is as a tool for finding reductions. Thus we demand our definition satisfy the following lemma.

Lemma 3.2. *Let S and S' be finite sets of gizmos, and suppose S simulates each gizmo in S'. Then there is a logarithmic-space reduction from targeted set reconfiguration with S' to targeted set reconfiguration with S.*

If all of the gizmos in S and S' are prefix-closed, then this is a reduction between reachability problems.

Note that the reduction goes the opposite direction of the simulations. Intuitively, the reduction is to replace each gizmo in S' with its simulation using gizmos from S.

If one were to define formal objects analogous to gizmos to study other decision problems related to gadgets (e.g. multiplayer games, untargeted reconfiguration), the formal objects should satisfy an analog of Lemma 3.2 for the decision problem in question.

[8] One can consider targeted set reconfiguration with an infinite set of gizmos, but it requires a way to describe arbitrary gizmos in the set.

3.2.2 Ignore irrelevant differences

One guiding principle in designing our definition is that we would like to ignore differences between gadgets which are irrelevant in the context of one-player reachability. That is, if replacing an instance of G with an instance of G' never changes the answer to a reachability problem, we would like to consider G and G' to be equivalent. If one simulates G with some gadgets, we would like to say that they have also simulated G', at least for the purposes of one-player reachability. This notion of equivalence is obviously different when one considers other decision problems. Ultimately, gadgets which are equivalent in this sense will be represented by the same gizmo.

One-player incentives. One type of irrelevant difference is optional opening, as discussed in Section 3.1, and more generally 'one-player incentives': whenever a gadget has a transtion which is never required to solve a reachability problem, we can ignore it. The gadget is equivalent to a version of the gadget without that transition.

Self-loops. In the self-closing door (Figure 2-4), the button to set the state to 1 is only available in state 2. Consider a modified version of this gadget where we add a transition that allows the player to press the button while in state 1, which does not change the state. Perhaps our gadget came from an application which includes an actual button an agent can press, and they can press the button regardless of the door's openness. We should consider this gadget equivalent to the original self-closing door, since the new self-loop transition is useless.

This can be thought of as an instance of one-player incentives, but it is also something more specific: adding 'self-loop' transitions of the form $(s, l) \to (s, l)$ does not meaningfully change a gadget. In particular, if we can simulate a gadget, except that our simulation also allows the agent to enter, do nothing, and exit at the same location, we want to consider this a valid simulation. To keep gizmo equality simple, it will be most convenient to assume our gadgets always have all self-loops, and this will be built into the definition of gizmos.

Transitivity. Consider the gadget whose state diagram is shown in Figure 3-1. From state 1, the agent can enter at A, move to B switching to state 2, and then immediately move to C switching to state 3. Consider also a modified version of this gadget where we add a transition $(1, A) \to (3, C)$, allowing the path described to occur in a single transition. This gadget is equivalent, since the same transition could be achieved through a combination of two transitions. Similarly we could introduce without meaningfully changing the gadget the transitions $(2, B) \to (3, D)$ and $(1, A) \to (3, D)$.

Similarly to for self-loops, it will be most convenient to assume that all of our gadgets already have these transitions, and this will be built into our definition of gizmos. We call this property 'transitive closure': if there are transitions $(s_1, l_1) \to (s_2, l_2) \to (s_3, l_3)$, then there is also $(s_1, l_1) \to (s_3, l_3)$.

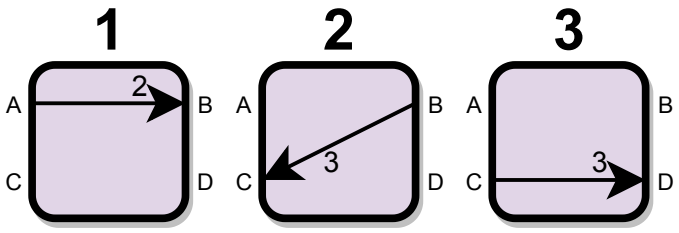

Figure 3-1: A state diagram for a gadget demonstrating transitive closure, with locations labeled A—D.

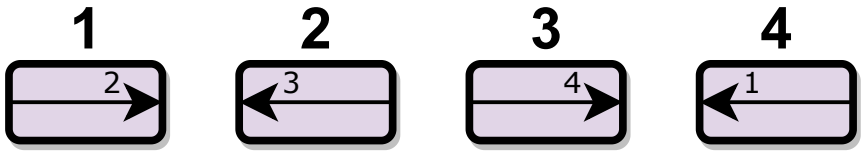

Figure 3-2: A state diagram for the doubly covered 1-toggle.

3.2.3 Reversible is closed

As shown by Demaine, Grosof, et al. [DGLR18], reversible gadgets are closed under simulations. We would like this to be the case in our formalization.

Consider the *doubly covered 1-toggle*, whose state diagram is shown in Figure 3-2. The doubly covered 1-toggle is much like a 1-toggle, except that it only returns to its initial state after being traversed twice in each direction.

The doubly covered 1-toggle is equivalent to the (ordinary) 1-toggle for the purposes of reachability. This is analogous to (in fact, a special case of) a regular language which is recognized by multiple DFAs. The legal movements are the same, but are represented by states of gadgets differently.

We would like to say that, in the context of reachability, the 1-toggle simulates the doubly covered 1-toggle. However, the doubly covered 1-toggle is not reversible as defined above: it contains the transition $(1, A) \to (2, B)$ but not $(2, B) \to (1, A)$.[9] It seems, then, that reversible gadgets are not closed under simulation. Either our definition of reversibility is wrong (and the doubly covered 1-toggle is actually reversible), or our notion of simulation is wrong (and the 1-toggle actually cannot simulate the doubly covered 1-toggle).

Our stance will be that the 1-toggle can simulate the doubly covered 1-toggle, and the doubly covered 1-toggle is reversible. In fact, the doubly covered 1-toggle will be represented by precisely the same gizmo as the 1-toggle.[10]

To resolve this difficulty using the description of gadgets in terms of states—

[9]The transitive closure of the doubly covered 1-toggle does contain $(2, B) \to (1, A)$; one can easily construct more complicated examples where this would not be true.

[10]This is true specifically for the gizmos representing this gadgets with full target set. The gizmo representing the doubly covered 1-toggle with a different target set, such as the set containing only state 1, does not represent the 1-toggle with any target set, and in fact cannot be simulated by 1-toggles and should not be considered reversible.

essentially finite automata—one would have to perform some kind of NFA minimization in order to determine whether two gadgets are equivalent. This is the approach taken by Bosboom [Bos20]. Our definition of gizmos avoids this by directly saying which motions are legal, instead of going through finite automata. This is analogous (actually equivalent; see Section 3.6) to considering languages instead of the NFAs that recognize them.

3.3 The definition

3.3.1 Gizmos

The idea is for a gizmo to be described by the set of legal sequences of traversal that can be made. In the following L and L' are sets, representing the set of locations of a gadget.

Definition 3.3. A *traversal* on L is a pair of elements $a, b \in L$, written $a \to b$. We write $\mathcal{T}(L) = L \times L$ for the set of traversals on L.

Definition 3.4. A *traversal sequence* on L is a sequence of traversals on L. If X and Y are traversal sequences on L, we write their concatenation XY. We write explicit traversal sequences using brackets, such as $[a \to b, c \to d]$. We denote the set of traversal sequences on L by $\mathcal{T}(L)^*$.

Definition 3.5. A function $f : L \to L'$ induces functions $f_\circ : \mathcal{T}(L) \to \mathcal{T}(L')$ and $f_\circ^* : \mathcal{T}(L)^* \to \mathcal{T}(L')^*$, by

$$f_\circ : a \to b \mapsto f(a) \to f(b)$$
$$f_\circ^* : [t_1, \ldots, t_n] \mapsto [f_\circ(t_1), \ldots, f_\circ(t_n)].$$

Clearly $\cdot_\circ$ and $\cdot_\circ^*$ are (covariant) functors.[11]

Definition 3.6. A *gizmo* on a set L is a set of traversal sequences $G \subset \mathcal{T}(L)^*$ such that (for all $X, Y \in \mathcal{T}(L)^*$, $a, b, c \in L$)

1. G is transitively closed: if $X[a \to b, b \to c]Y \in G$, then $X[a \to c]Y \in G$.

2. G is closed under insertion of self-loops: if $XY \in G$, then $X[a \to a]Y \in G$.

If G is a gizmo on L, we write $\operatorname{locs} G = L$.

These conditions are clearly suggested by the discussion in Section 3.2.2. It is interesting that these two properties are all that is needed for a set of traversal sequences to make sense as the legal actions an agent could take that leave a gadget in a state in its target set.

[11] That is, $(f \circ g)_\circ = f_\circ \circ g_\circ$, and similarly for $\cdot_\circ^*$.

Definition 3.7. Two gizmos G and G' are *isomorphic*, and we write $G \cong G'$, if there is a bijection $f : \operatorname{locs} G \to \operatorname{locs} G'$ such that for all $X \in \mathcal{T}(\operatorname{locs} G)^*$, $X \in G$ if and only if $f^*_\circ(X) \in G'$. Such a bijection is called an *isomorphism*.

Isomorphic gizmos are the same except that their locations are labeled differently; all interesting properties of gizmos are preserved by isomorphism.

Definition 3.8. Let G be a gizmo on L, and $X \in \mathcal{T}(L)^*$. Then G *after* X is the set of traversal sequences

$$G[X] = \{Y \mid XY \in G\} \subset \mathcal{T}(L)^* .$$

G after X represents the state a gadget changes to after traversing X; see Proposition 3.16.

Definition 3.9. A gizmo G is *prefix-closed* if for every $XY \in G$, also $X \in G$.

An agent navigating a network of gadgets changes the gizmos corresponding to those gadgets with every traversal: moving from a to b replaces a gizmo G with $G[[a \to b]]$. The empty sequence $[\,] \in G[X]$, or equivalently $X \in G$, means that after traversing the sequence X, the gizmo G is 'happy': if this is true for every gizmo and the agent is at the target location, the targeted set reconfiguration problem is solved. A prefix-closed gizmo G is one which never goes from being 'sad' to being 'happy'—formally, for any sequence X, if $[\,] \notin G[X]$ then $G[X] = \emptyset$, and the empty gizmo will never be satisfied. One can think of the states of a gizmo G as the nonempty reachable gizmos $G[X]$ (an empty gizmo representing something illegal having happened), so prefix-closed gizmos represent gadgets with every reachable state in the target set. We will show in Section 4.1 that prefix-closed gizmos are closed under simulation, which is expected: if we do not care what state gadgets end in, we should not care what state the gadgets they simulate end in either.

Let us translate some gadgets into this formalism. The legal traversal sequences for state 1 of the 1-toggle (Figure 2-1), with locations A and B, include $[\,]$, $[A \to B]$, $[A \to B, B \to A]$, and $[A \to B, A \to A, B \to B, B \to A, B \to B]$. Note in particular that we can freely insert $A \to A$ and $B \to B$ because of condition 3 in Definition 3.6. We can precisely describe the gizmo corresponding to this state with target set containing both states, which is the set of all legal traversal sequences, with a regular expression, using A, B, $\to$, and $\leftarrow$ as shorthand for $A \to A$, $B \to B$, $A \to B$, and $B \to A$, respectively:[12]

$$((A|B)^*{\to}(A|B)^*{\leftarrow})^*(A|B)^*({\to}|\varepsilon)(A|B)^*.$$

Describing the gizmo explicitly in this way quickly becomes cumbersome. It is more convenient to describe it as the minimal prefix-closed gizmo containing $(\to\leftarrow)^*$, and let closure under prefixes and insertion of self-loops handle the rest. There is always a unique minimal gizmo and minimal prefix-closed gizmo on a set of locations containing some set of traversal sequences: it is the set of sequences obtainable from

[12] See Section 3.6 for more on the connection to regular languages.

sequences in the given set by inserting self-loops, applying transitive contractions, and, for prefix-closed, taking prefixes.

There are two other nontrivial gizmos[13] corresponding to state 1 of the 1-toggle, with different target sets. If the target set contains just state 1, we have the minimal gizmo containing $(\rightarrow\leftarrow)^*$; unlike the gizmo for the full target set, this does not contain $[\rightarrow]$. If the target set contains just state 2, we have the minimal gizmo containing $(\rightarrow\leftarrow)^*\rightarrow$.

The gizmos for state 2 of the 1-toggle are different; for instance the gizmo for the full target set includes $[B \to A]$ but not $[A \to B]$. However, this is equivalent to the gizmo for state 1 with full target set if we turn the 1-toggle around; formally, these gizmos are isomorphic by swapping A and B.

State 1 of the 2-toggle (Figure 2-6) with full target set is the minimal prefix-closed gizmo containing $((A \to B \mid C \to D)(B \to A \mid D \to C))^*$, if we label locations appropriately. State 2 of the 2-toggle (also with full target set) is isomorphic to state 1 under $A \leftrightarrow B$, $C \leftrightarrow D$.

Unless otherwise specified, when we define a gizmo from a gadget and a state, we mean for the target set to be the entire set of states, and in particular the gizmo is prefix-closed.

For one more example, state 1 of the tripwire-lock (Figure 2-7) is the minimal prefix-closed gizmo containing

$$((A \to B \mid B \to A)(A \to B \mid B \to A) \mid C \to D \mid D \to C)^*.$$

State 2 of the tripwire-lock is not isomorphic to state 1;[14] it is the minimal prefix-closed gizmo containing

$$((A \to B \mid B \to A)(C \to D \mid D \to C)^*(A \to B \mid B \to A))^*.$$

3.3.2 Simulations

We now move on to defining simulations between gizmos. We think of gadget simulations as being done in three steps: first, put some gadgets next to each other. Second, connect some locations. Finally, designate which locations are to be 'ports' of the simulation accessible from outside. We first define our notion for putting gadgets next to each other.

Definition 3.10. Let I be an index set, and suppose G_i is a gizmo for each $i \in I$. The *tensor product* of G_i, denoted $\bigotimes_{i\in I} G_i$, is the set of traversal sequences X on $\bigsqcup_{i\in I} \operatorname{locs} G_i$[15] such that

[13] And the empty gizmo, where the target set is empty.

[14] You can see this by noting that state 1 contains eight length-1 traversal sequences ($A \leftrightarrow B$, $C \leftrightarrow D$, and 4 self-loops) while state 2 contains only six (the same ones except for $C \leftrightarrow D$).

[15] $\bigsqcup$ denotes disjoint union: formally, $\bigsqcup_{i\in I} S_i = \{(i, x) \mid i \in I, x \in S_i\}$, which is isomorphic to $\bigcup_{i\in I} S_i$ when S_i are disjoint. For convenience, we generally treat $(i, x) \in \bigsqcup_{i\in I} S_i$ and $x \in S_i$ as equal.

1. For each traversal $a \to b$ in X, both locations are on the same multiplicand: for some i, we have $a, b \in \operatorname{locs} G_i$.

2. For each $i \in I$, the restriction $X|_{\operatorname{locs} G_i}$, i.e the sequence of traversals in X which only use locations in $\operatorname{locs} G_i$, is in G_i.

We say that X is an *interleaving* of $X|_{\operatorname{locs} G_i}$.

The tensor product represents putting the gadgets G_i next to each other, and allowing the agent to make traversals on them in any order. It may be easier to think about the situation when $|I| = 2$, where we have

$$G \otimes H = \{X_1 Y_1 \cdots X_n Y_n \mid X_1 \cdots X_n \in G, Y_1 \cdots Y_n \in H\}.$$

While we could use this formula to define tensor products, using the unbiased definition above allows us to simplify proofs that would otherwise use induction on the number of gizmos, and to have the option of discussing simulations involving infinitely many gizmos.

Next, we define a notion for connecting locations of a gadget.

Definition 3.11. Let G be a gizmo, and let $\sim$ be an equivalence relation on $\operatorname{locs} G$. The *quotient* $G/\!\sim$ is the transitive closure of the set $\{\pi_{\sim\circ}^*(X) \mid X \in G\}$ of traversal sequences on $\operatorname{locs}(G)/\!\sim$, where $\pi_\sim : \operatorname{locs} G \to \operatorname{locs} G/\!\sim$ is the projection to equivalence classes induced by $\sim$.

By *transitive closure*, we mean the smallest superset which is transitively closed as defined in Definition 3.6.

The quotient $G/\!\sim$ represents the gadget formed by connecting locations of G which are related by $\sim$. We need to take transitive closure because connected locations may allow traversal sequences to be contracted: if $X[a \to b, c \to d]Y \in G$ and $b \sim c$, the transitive closure induces a new traversal sequence $X[a \to d]Y$ in $G/\!\sim$.

We can describe the quotient in a slightly different way, which will be more convenient for many proofs.

Definition 3.12. Let X and Y be traversal sequences on L. We say that Y is a *transitive contraction* of X, and write $X \rightsquigarrow Y$, if we can obtain Y from X by a sequence of local replacements of the form $[a \to b, b \to c] \mapsto [a \to c]$. Formally, $\rightsquigarrow$ is the minimal reflexive transitive relation with $X[a \to b, b \to c]Y \rightsquigarrow X[a \to c]Y$.

Clearly the transitive closure of a set S of traversal sequences is

$$\{Y \mid X \in S, X \rightsquigarrow Y\}.$$

Thus we have another description of the quotient:

Proposition 3.13. *Suppose G is a gizmo and $\sim$ is an equivalence relation on* $\operatorname{locs} G$. *Then for any traversal sequence X, $X \in G/\!\sim$ if and only if $\pi_{\sim\circ}^*(Y) \rightsquigarrow X$ for some $Y \in G$.*

Finally, we define a notion for specifying accessible ports of a gadget.

Definition 3.14. Let G be a gizmo, and let $L \subset \operatorname{locs} G$. The *subgizmo* $G|_L$ is $G \cap \mathcal{T}(L)^*$.

The subgizmo $G|_L$ represents the gadget formed by ignoring all the locations in G except for those in L. Think of L as indicating the 'ports' of a simulation, and only those locations are accessible to the outside world.

Definition 3.15. Let S be a set of gizmos, and let G be a gizmo. Then S *arbitrarily simulates* G if there is a collection of gizmos $G_i \in S$ indexed by a set I, an equivalence relation $\sim$ on $\bigsqcup_{i \in I} \operatorname{locs} G_i$, and a set $L \subset \bigsqcup_{i \in I} \operatorname{locs} G_i \Big/ \sim$, such that

$$G \cong \bigotimes_{i \in I} G_i \Big/ \sim \Big|_L .$$

Such a combination $(I, \{G_i\}_{i \in I}, \sim, L)$ is called an *arbitrary simulation* of G from S. We say that G is *arbitrarily simulable* from S if such a simulation exists, and *arbitrarily unsimulable* from S otherwise.

A *finite simulation* is one where I is finite, and we say *finitely simulable*, etc. We also use *simulation*, *simulable*, etc. to refer to this finite version.

When $S = \{G'\}$ is a singleton, we will often say simply that G' simulates G.

Note that our definition of simulation allows for simulations which use multiple different gizmos. Usually when one is interested in a finding a simulation, S is a finite set, and often a singleton. Since gizmos represent individual states of a gadget, often S is taken to contain a gizmo for each such state.

To illustrate our definition of simulation, we will translate the simulation in Figure 2-14 into the formalism. The (prefix-closed) gizmos we are using are

$$S = \{\text{1-toggle}, \text{open tripwire-lock}, \text{closed tripwire-lock}\}$$

(recall that the two states of the 1-toggle are isomorphic, so there is no reason to specify which one). The index set I has 6 elements, and there are two G_is which are each element of S. The tensor product is a gizmo (see Proposition 3.17) with 20 locations labeled A through T and 64 states (since each of the 6 components has two states; see Section 3.6 for the correct notion of 'state').

Next we take a quotient to connect locations. In this case, we have $B \sim E \sim G$, $D \sim F \sim I$, $C \sim K$, $J \sim R$, $L \sim O \sim Q$, and $N \sim P \sim S$. We use a letter to denote its equivalence class. This gizmo (see Proposition 3.18) has 9 locations, corresponding to the connected regions outside of gadgets in the network, and still has 64 states. It includes traversal sequences such as $[A \to E]$, $[E \to F]$, $[A \to E, E \to F]$, $[A \to F]$ (because of transitivity), and $[A \to H, T \to M]$, but not $[A \to M]$ or $[F \to C, P \to O, E \to A]$.

Finally, we take the subset $L = \{A, H, M, T\}$ of locations representing the 'ports' of the simulation. The resulting subgizmo (see Proposition 3.19) includes e.g. $[A \to$

$H, T \to M]$ but not $[A \to M]$. It can be verified that this gizmo is isomorphic to the 2-toggle.

With our definition of simulation in hand, we can refine Question 1.1 into a more well-defined question:

Question 3.1. For a set S of gizmos and a gizmo G, when does S simulate G?

Since we are primarily interested in reachability, which corresponds to prefix-closed gizmos, our goal is to answer this special case:

Question 3.2. For a set S of prefix-closed gizmos and a prefix-closed gizmo G, when does S simulate G?

3.4 Basic properties

In this section we prove several basic properties of gizmos which one expects should hold if our definitions are reasonable. Most of them are straightforward to prove.

Proposition 3.16. *Let G be a gizmo on L, and $X, Y \in \mathcal{T}(L)$.*

1. *$G[X]$ is a gizmo.*
2. *$G[X][Y] = G[XY]$*
3. *If G is prefix-closed, then $G[X]$ is prefix-closed.*

Proof. These are all straightforward.

1. We must show $G[X] = \{Y \mid XY \in G\}$ is transitive and has self-loops. For transitivity,

$$\begin{aligned} Y[a \to b, b \to c]Z \in G[X] &\iff XY[a \to b, b \to c]Z \in G \\ &\implies XY[a \to c]Z \in G \iff Y[a \to c]Z \in G[X]. \end{aligned}$$

   For self-loops,

$$\begin{aligned} YZ \in G[X] &\iff XYZ \in G \\ &\implies XY[a \to a]Z \in G \iff Y[a \to a]Z \in G[X]. \end{aligned}$$

2. $Z \in G[X][Y] \iff YZ \in G[X] \iff XYZ \in G \iff Z \in G[XY]$.
3. We have $YZ \in G \implies Y \in G$, so

$$YZ \in G[X] \iff XYZ \in G \implies XY \in G \iff Y \in G[X].$$

□

Proposition 3.17. *Suppose G_i is a gizmo for every $i \in I$. Then $\bigotimes_{i \in I} G_i$ is a gizmo.*

Proof. For transitivity, suppose $X[a \to b, b \to c]Y \in \bigotimes_{i \in I} G_i$. Then for some $i, i' \in I$, we have $a, b \in \operatorname{locs} G_i$ and $b, c \in \operatorname{locs} G_{i'}$. So $i = i'$ and $\operatorname{locs} G_i$ contains all three locations. Thus each traversal in $X[a \to c]Y$ is on a single multiplicand. The restriction $(X[a \to b, b \to c]Y)|_{\operatorname{locs} G_i} = X_{\operatorname{locs} G_i}[a \to b, b \to c]Y_{\operatorname{locs} G_i} \in G_i$, so $X_{\operatorname{locs} G_i}[a \to c]Y_{\operatorname{locs} G_i} = (X[a \to c]Y)|_{\operatorname{locs} G_i} \in G_i$. The other restrictions do not change when we contract $[a \to b, b \to c]$. Hence $X[a \to c]Y \in \bigotimes_{i \in I} G_i$.

For self-loops, suppose $XY \in \bigotimes_{i \in I} G_i$ and $a \in \operatorname{locs} G_i$. Then $X_{\operatorname{locs} G_i} Y_{\operatorname{locs} G_i} \in G_i$, so $X_{\operatorname{locs} G_i}[a \to a]Y_{\operatorname{locs} G_i} \in G_i$, and other restrictions do not change upon inserting $a \to a$. The only new traversal $a \to a$ is on $\operatorname{locs} G_i$, so $X[a \to a]Y \in \bigotimes_{i \in I} G_i$. □

Proposition 3.18. *Let G be a gizmo and $\sim$ be an equivalence relation on $\operatorname{locs} G$. Then $G/\!\sim$ is a gizmo.*

Proof. Transitivity is immediate from the definition of $G/\!\sim$. For self-loops, suppose $XY \in G/\!\sim$ and $[a] \in \operatorname{locs} G/\!\sim$. Then XY is a transitive contraction of $\pi_{\sim\circ}^*(Z)$ for some $Z \in G$. We can decompose Z as $Z = X'Y'$ where $\pi_{\sim\circ}^*(X') \rightsquigarrow X$ and $\pi_{\sim\circ}^*(Y') \rightsquigarrow Y$. Since G is a gizmo, $X'[a \to a]Y' \in G$. But $\pi_{\sim\circ}^*(X'[a \to a]Y') \rightsquigarrow X[[a] \to [a]]Y$, so we have $X[[a] \to [a]]Y \in G/\!\sim$.**[should I prove the facts about $\rightsquigarrow$ I use?]** □ xxx

Proposition 3.19. *Let G be a gizmo and $L \subset \operatorname{locs} G$. Then $G|_L$ is a gizmo.*

Proof. If $X[a \to b, b \to c]Y \in G|_L$, then $X[a \to b, b \to c]Y \in G$, so $X[a \to c]Y \in G$. But all the locations in $X[a \to c]Y$ are in L, so $X[a \to c]Y \in G|_L$. Next, if $XY \in G|_L$ and $a \in L$, then $X[a \to a]Y \in G$ and again all locations are in L, so $X[a \to a]Y \in G|_L$. □

Propositions 3.17 through 3.19 are crucial properties of our definitions of tensor product, quotient, and subgizmo.

Proposition 3.20. *Let I be an index set, G_i and G be gizmos, $\sim$ be an equivalence relation on $\operatorname{locs} G$, $L \subset \operatorname{locs} G$, and X be a traversal sequence on the appropriate set. Then the 'after X' operation interacts with the other operations we defined in the following ways:*

1. *If $\left(\bigotimes_{i \in I} G_i\right)[X]$ is nonempty, then $\left(\bigotimes_{i \in I} G_i\right)[X] = \bigotimes_{i \in I} (G_i[X|_{\operatorname{locs} G_i}])$.*
2. $(G/\!\sim)[X] = \bigcup_{\substack{\widetilde{X} \in \mathcal{T}(\operatorname{locs} G)^* \\ \pi_{\sim\circ}^*(\widetilde{X}) \rightsquigarrow X}} G[\widetilde{X}]/\!\sim$
3. $(G|_L)[X] = G[X]|_L$.

Proof. Let Y be a traversal sequence on the appropriate set.

1. Each traversal in X must be on a single multiplicand. If this is not also true of Y, then Y is in neither gizmo. Otherwise,

$$\begin{aligned} Y \in \left(\bigotimes_{i\in I} G_i\right)[X] &\iff XY \in \bigotimes_{i\in I} G_i \\ &\iff \forall i \in I : (XY)|_{\operatorname{locs} G_i} \in G_i \\ &\iff \forall i \in I : Y|_{\operatorname{locs} G_i} \in G_i[X|_{\operatorname{locs} G_i}] \\ &\iff Y \in \bigotimes_{i\in I}(G_i[X|_{\operatorname{locs} G_i}]). \end{aligned}$$

2. The union comes from the fact that there may be many traversal sequences in G corresponding to X, and $(G/\!\sim)[X]$ is the superposition of all states reachable by such sequences. Formally:

$$\begin{aligned} Y \in (G/\!\sim)[X] &\iff XY \in G/\!\sim \\ &\iff \exists \widetilde{XY} \in G : \pi^*_{\sim\circ}(\widetilde{XY}) \rightsquigarrow XY \\ &\iff \exists \widetilde{X}, \widetilde{Y} : \widetilde{X}\widetilde{Y} \in G \;\&\; \pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X \;\&\; \pi^*_{\sim\circ}(\widetilde{Y}) \rightsquigarrow Y \\ &\iff \exists \widetilde{X} : \pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X \;\&\; Y \in G[\widetilde{X}]/\!\sim \\ &\iff Y \in \bigcup_{\substack{\widetilde{X}\in\mathcal{T}(\operatorname{locs} G)^* \\ \pi^*_{\sim\circ}(\widetilde{X})\rightsquigarrow X}} G[\widetilde{X}]/\!\sim. \end{aligned}$$

3. A sequence $Y \in \mathcal{T}(L)^*$ is in each gizmo if and only if $XY \in G$, since both X and Y are on L.

□

Proposition 3.21. *Let I and $\{H_i\}_{i\in I}$ be index sets; G, $\{G_i\}_{i\in I}$, and $\{G_{ih}\}_{i\in I, h\in H_i}$ be gizmos; $\sim$, $\sim'$, and $\sim_i$ be equivalence relations on appropriate sets; and L, L', and L_i be appropriate sets. Tensor product, quotient, and subgizmo obey the following identities:*

1. $\displaystyle\bigotimes_{i\in I}\bigotimes_{h\in H_i} G_{ih} \cong \bigotimes_{(i,h)\in\bigsqcup_{i\in I} H_i} G_{ih}$
2. $G/\!\sim/\!\sim' \;\cong G/(\sim' \circ \sim)$
3. $G|_L|_{L'} = G|_{L'}$
4. $\displaystyle\bigotimes_{i\in I} G_i/\!\sim_i \;\cong \bigotimes_{i\in I} G_i \Big/ \bigsqcup_{i\in I} \sim_i$
5. $\displaystyle\bigotimes_{i\in I} G_i|_{L_i} = \bigotimes_{i\in I} G_i\Big|_{\bigsqcup_{i\in I} L_i}$

Here the composition $(\sim' \circ \sim)$ of equivalence relations is the equivalence relation where a $(\sim' \circ \sim)$ b whenever $[a]_\sim \sim' [b]_\sim$, where $[x]_\sim$ is the $\sim$-equivalence class of x; equivalently up to isomorphism, $\pi_{(\sim' \circ \sim)} = \pi_{\sim'} \circ \pi_\sim$. The disjoint union $\bigsqcup_{i \in I} \sim_i$ of equivalence relations $\sim_i$ on A_i is the natural equivalence relation on $\bigsqcup_{i \in I} A_i$.

For tensor products of only two gizmos, identities 1, 4, and 5 can be written

$$\begin{aligned} G_1 \otimes (G_2 \otimes G_3) &\cong (G_1 \otimes G_2) \otimes G_3 \\ G/\!\sim \otimes G/\!\sim' &\cong (G \otimes G')/(\sim \sqcup \sim') \\ G|_j \otimes G|_{j'} &\cong (G \otimes G')|_{j \sqcup j'}. \end{aligned}$$

Proof. The isomorphisms are the obvious bijections between location sets.

1. Whether a traversal has both locations on the same G_{ih} is clearly equivalent for both sides. Let X be a traversal sequence on $\bigsqcup_{i \in I, h \in H_i} \operatorname{locs} G_{ih}$ in which every traversal is on a single gizmo. Then

$$\begin{aligned} X \in \bigotimes_{i \in I} \bigotimes_{h \in H_i} G_{ih} &\iff \forall i \in I : X\Big|_{\bigsqcup_{h \in H_i} \operatorname{locs} G_{ih}} \in \bigotimes_{h \in H_i} G_{ih} \\ &\iff \forall i \in I, h \in H_i : \left(X \Big|_{\bigsqcup_{h \in H_i} \operatorname{locs} G_{ih}} \right) \Big|_{\operatorname{locs} G_{ih}} \in G_{ih} \\ &\iff \forall i \in I, h \in H_i : X|_{\operatorname{locs} G_{ih}} \in G_{ih} \\ &\iff X \in \bigotimes_{(i,h) \in \bigsqcup_{i \in I} H_i} G_{ih}. \end{aligned}$$

2. Suppose $X \in G/\!\sim/\!\sim'$. Then $\pi_{\sim'\circ}^*(Y) \rightsquigarrow X$ for some $Y \in G/\!\sim$. But then $\pi_{\sim\circ}^*(Z) \rightsquigarrow Y$ for some $Z \in G$. So we have $\pi_{\sim'\circ}^*(\pi_{\sim\circ}^*(Z)) \rightsquigarrow \pi_{\sim'\circ}^*(Y) \rightsquigarrow X$. Under the isomorphism between $\operatorname{locs} G/\!\sim/\!\sim'$ and $\operatorname{locs} G/(\sim' \circ \sim)$, we have $\pi_{(\sim'\circ\sim)})Z) \rightsquigarrow X$, so $X \in G/(\sim' \circ \sim)$.

   Conversely, suppose $X \in G/(\sim' \circ \sim)$. Then $\pi_{(\sim'\circ\sim)}(Z) \rightsquigarrow X$ for some $Z \in G$. Then we can take $Y = \pi_{\sim\circ}^*(Z) \in G/\!\sim$, and up to the same isomorphism $\pi_{\sim'\circ}^*(Y) \rightsquigarrow X$, so $X \in G/\!\sim/\!\sim'$.

3. For $G|_L|_{L'}$ to be defined, $L' \subset L$, so $G|_L|_{L'} = G \cap \mathcal{T}(L)^* \cap \mathcal{T}(L')^* = G \cap \mathcal{T}(L')^* = G|_{L'}$.

4. Whether both locations of a traversal are on the same multiplicand is clearly equivalent on both sides, since $\sim_i$ is confined to $\operatorname{locs} G_i$.

   Take $X \in \bigotimes_{i \in I} G_i/\!\sim_i$, so $X|_{\operatorname{locs} G_i/\sim_i} \in G_i/\!\sim_i$. But then $\pi_{\sim_i\circ}^*(Y_i) \rightsquigarrow X|_{\operatorname{locs} G_i/\sim_i}$ for some $Y_i \in G_i$. Let $\widetilde{X} \in \mathcal{T}\left(\bigsqcup_{i \in I} \operatorname{locs} G_i\right)^*$ be the interleaving of Y_i cor-

responding to X (thought of as an interleaving of $X|_{\operatorname{locs} G_i/\sim_i}$). [**this fact about $\rightsquigarrow$ is less obvious**] Then $\widetilde{X} \in \bigotimes_{i\in I} G_i$, and $\pi_{\bigsqcup_{i\in I}\sim_i}{}^{*}_{\circ}\left(\widetilde{X}\right) \rightsquigarrow X$, so $X \in \bigotimes_{i\in I} G_i \Big/ \left(\bigsqcup_{i\in I} \sim_i\right)$.

Conversely, take $X \in \bigotimes_{i\in I} G_i \Big/ \left(\bigsqcup_{i\in I} \sim_i\right)$, so we have $\pi_{\bigsqcup_{i\in I}\sim_i}{}^{*}_{\circ}\left(\widetilde{X}\right) \rightsquigarrow X$ for some $\widetilde{X} \in \bigotimes_{i\in I} G_i$. Each contraction occurs between traversals on the same multiplicand, so they can all be performed on each $\widetilde{X}|_{\operatorname{locs} G_i}$ separately. That is, $\pi_{\sim_i}{}^{*}_{\circ}(\widetilde{X}|_{\operatorname{locs} G_i}) \rightsquigarrow X|_{\operatorname{locs} G_i/\sim_i}$, so $X|_{\operatorname{locs} G_i/\sim_i} \in G_i/\sim_i$, and hence $X \in \bigotimes_{i\in I} G_i/\sim_i$.

5. Once again, it is clearly equivalent on both sides whether the locations of a traversal come from the same gizmo. Conditioned on this being true for a traversal sequence $X \in \mathcal{T}\left(\bigsqcup_{i\in I} L_i\right)^*$,

$$\begin{aligned} X \in \bigotimes_{i\in I} G_i|_{L_i} &\iff \forall i \in I : X|_{L_i} \in G_i|_{L_i} \\ &\iff \forall i \in I : X|_{\operatorname{locs} G_i} \in G_i|_{L_i} \\ &\iff \forall i \in I : X|_{\operatorname{locs} G_i} \in G_i \\ &\iff X \in \bigotimes_{i\in I} G_i \\ &\iff X \in \bigotimes_{i\in I} G_i\Big|_{\bigsqcup_{i\in I} L_i}. \end{aligned}$$

Note that we use the fact that X is on $\bigsqcup_{i\in I} L_i$.

□

Our final identity is a bit more complicated.

Proposition 3.22. *Let G be a gizmo, $L \subset \operatorname{locs} G$, and $\sim$ be an equivalence relation on L. Then*

$$G|_L/\sim \;=\; G/(\sim \cup =)|_{L/\sim},$$

where $(\sim \cup =)$ is the extension of $\sim$ to an equivalence relation on $\operatorname{locs} G$, meaning $a \;(\sim \cup =)\; b$ if $a \sim b$ or $a = b$.

Proof. Suppose $X \in G|_L/\sim$. Then $\pi_{\sim_\circ}^*(Y) \rightsquigarrow X$ for some $Y \in G|_L$. Then we have $Y \in G$, and $\pi_{\sim_\circ}^*(Y) = \pi_{(\sim\cup=)_\circ}^*(Y)$, so $X \in G/(\sim \cup =)$. Since X is on $L/\sim$, also $X \in G/(\sim \cup =)|_{L/\sim}$.

Now suppose $X \in G/(\sim \cup =)|_{L/\sim}$, and think of X as a traversal sequence on $\operatorname{locs} G/(\sim \cup =)$. Then $\pi_{(\sim\cup=)_\circ}^*(Y) \rightsquigarrow X$ for some $Y \in G$. Suppose a traversal in

Y has a location a not in L. Then this location must be contracted away in the conversion to X, so the appropriate location b in an adjacent traversal in Y must be in the same equivalence class under $(\sim \cup =)$ as a. But the only nontrivial relations are between elements of L, so $a = b$ and this transitive contraction can be performed in Y before taking the quotient (since G is transitive). Hence, by taking Y to have minimal length, we can assume Y is on L. Thus $Y \in G|_L$, and X is a transitive contraction of $\pi_{\sim\circ}^*(Y)$, so $X \in G|_L/\sim$. □

Simulations are transitive in the natural sense.

Lemma 3.23. *Let S and S' be sets of gizmos, and let G be a gizmo. Suppose S finitely (resp. arbitrarily) simulates each gizmo in S', and S' finitely (resp. arbitrarily) simulates G. Then S finitely (resp. arbitrarily) simulates G.*

Proof. Since S' simulates G, we have $G \cong \bigotimes_{i\in I} G_i/\sim|_L$ with $G_i \in S'$, $\sim$ an equivalence relation on $\bigsqcup_{i\in I} \operatorname{locs} G_i$, and $L \subset \bigsqcup_{i\in I} \operatorname{locs} G_i \Big/ \sim$. For each $i \in I$, since S simulates G_i, we have $G_i \cong \bigotimes_{h\in H_i} G_{ih}/\sim_i|_{L_i}$ with $G_{ih} \in S$. For sanity, assume $L_i = \operatorname{locs} G_i$ and these isomorphisms are the identity; otherwise we would need to adjust $\sim$ and L according to the isomorphisms. We proceed algebraically using Propositions 3.21 and 3.22:

$$
\begin{aligned}
G &\cong \bigotimes_{i\in I} G_i \Big/ \sim \Big|_L \\
&\cong \bigotimes_{i\in I} \left(\bigotimes_{h\in H_i} G_{ih}/\sim_i|_{L_i} \right) \Big/ \sim \Big|_L \\
&\cong \bigotimes_{i\in I} \left(\bigotimes_{h\in H_i} G_{ih}/\sim_i \right) \Big|_{\bigsqcup_{i\in I} L_i} \Big/ \sim \Big|_L \\
&\cong \bigotimes_{i\in I} \left(\bigotimes_{h\in H_i} G_{ih}/\sim_i \right) \Big/ (\sim \cup =) \Big|_{\bigsqcup_{i\in I} L_i \Big/ \sim} \Big|_L \\
&\cong \bigotimes_{i\in I} \left(\bigotimes_{h\in H_i} G_{ih}/\sim_i \right) \Big/ (\sim \cup =) \Big|_L \\
&\cong \bigotimes_{i\in I} \bigotimes_{h\in H_i} G_{ih} \Big/ \bigsqcup_{i\in I} \sim_i \Big/ (\sim \cup =) \Big|_L \\
&\cong \bigotimes_{(i,h)\in \bigsqcup_{i\in I} H_i} G_{ih} \Big/ \bigsqcup_{i\in I} \sim_i \Big/ (\sim \cup =) \Big|_L
\end{aligned}
$$

$$\cong \bigotimes_{(i,h)\in\bigsqcup I} G_{ih} \Big/ \left((\sim \cup =) \circ \bigsqcup_{i\in I} \sim_i \right) \Bigg|_L .$$

For the finite simulation case, if I and H_i are all finite, then so is $\bigsqcup I = \{(i,h) \mid i \in I, h \in H_i\}$. □

Corollary 3.24. *Let S be a set of gizmos. For both finite and arbitrary simulation, with I finite in the case of finite simulation:*

1. *If S simulates G_i for $i \in I$, then S simulates $\bigotimes_{i\in I} G_i$.*

2. *If S simulates G, then S simulates $G/\!\sim$.*

3. *If S simulates G, then S simulates $G|_L$.*

Proof. These follow from Lemma 3.23 by making some components of simulations trivial:

1. $\{G_i\}_{i\in I}$ simulates $\bigotimes_{i\in I} G_i$, by taking $\sim$ to be equality and $L = \bigcup_{i\in I} \operatorname{locs} G_i$.

2. G simulates $G/\!\sim$, by taking $I = \{0\}$, $G_0 = G$, and $L = \operatorname{locs} G/\!\sim$.

3. G simulates $G|_L$, by taking $I = \{0\}$, $G_0 = G$, and $\sim$ to be equality.

Here we use the fact that G, $\bigotimes_{i\in\{1\}} G$, $G/\!=$, and $G|_{\operatorname{locs} G}$ are all isomorphic, and tensor product, quotient, and subgizmo preserve isomorphism. □

Lemma 3.2—that simulations yield logarithmic-space reductions between targeted set reconfiguration problems—now follows easily.

Proof of Lemma 3.2. We have finite sets S and S' of gizmos, a simulation of each gizmo in S' from S, and a simulation of an unknown gizmo on $\{s,t\}$ from S'; we wish to construct a simulation of the same gizmo from S.[16]

This is exactly the simulation provided by Lemma 3.23. It must be noted that this simulation can be constructed in logarithmic space, since S and S' are finite and thus the simulations of the gizmos in S' from S have constant size. □

Definition 3.25. A set S of gizmos is *closed under (finite) simulation* if whenever S (finitely) simulates a gizmo G, $G \in S$.

S is *closed under abitrary simulation* if whenever S arbitrarily simulates a gizmo G, $G \in S$.

[16]The lemma only requires that the gizmo we simulate has the same opinion about the legality of $[s \to t]$, but it is no harder to construct exactly the same gizmo.

3.5 Consequences of our definition

In this section, we discuss some of the less intuitive consequences of our definitions of gizmo and simulation. Some of these consequences seem to inevitably result from formalizing gadgets, and some are the result of choices we made when crafting our definition.

3.5.1 Specific to one-player targeted set reconfiguration

As discussed in Section 3.1, different notion of simulation are relevant in different contexts. Gizmos themselves only make sense in the context of one-player targeted set reconfiguration. For situations such as multiplayer games or (untargeted) reconfiguration, one would need a new type of object distinct from gizmos. We are specifically interested in the more narrow context of one-player reachability, which prefix-closed gizmos are appropriate for.

3.5.2 Reductions not from simulations

We have defined simulation such that a simulation between gizmos yields a reduction between the corresponding targeted set reconfiguration problems (Lemma 3.2). However, there can be reductions between these problems which do not come from simulations. From a certain point of view, this is obvious: for instance, reachability with the 2-toggle (Figure 2-6) is PSPACE-complete [DGLR18], so there is a reduction from reachability with the door (Figure 2-3) to reachability with the 2-toggle. But the 2-toggle is reversible and thus cannot simulate the door (Theorem 4.6).

However, there are also reductions between reachability problems which have the same flavor as simulations between gadgets, but are not quite simulations. For instance, consider the notion of 'verified' gadgets from Lynch [Lyn20]. To simulate a verified version of a gadget G, you simulate a gadget with the same locations as G and an additional tunnel, such that the added tunnel is only available if G was used correctly. The verified gadget may have additional transitions, but using them must close the added tunnel. It is reasonable to describe this as a form of simulation, but does not fit our definition. Perhaps one could define a broader notion of simulation between gizmos which allows for this case.

3.5.3 Determinization

When described as state machines, there is a natural notion of deterministic gadgets. This notion does not at all translate to gizmos: gizmos only care about the legal sequences that can be made. If the player is able to nondeterministically choose the state of a gadget when making a traversal, the gadget can be thought of as being in a 'superposition' state: it may be in any of the states that could have been chosen, and 'collapses' to one of them when the agent makes another traversal which was only legal in one such state.

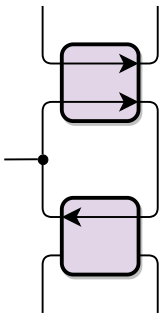

Figure 3-3: An NCL edge built out of locking 2-toggles; Figure 7a in Demaine, Hendrickson, and Lynch [DHL20]. Used here as an example of a nondeterministic gadget.

For instance, consider the NCL edge built out of locking 2-toggles (Figure 2-8) from Demaine, Hendrickson, and Lynch [DHL20], shown in Figure 3-3. If the agent enters at the left, it can choose to either immediately exit or go around the loop and then exit. In terms of state machines, we would say that $(1, L) \to (2, L)$ and $(1, L) \to (1, L)$ are both legal traversals, where L is the left location and the states are named 1 and 2. Which of these paths is taken controls which of the other two tunnels—let us call them $A \to B$ and $C \to D$—is available. After traversing $L \to L$, think of the gadget as being in a superposition of both of these states, with both tunnels open. If one of the tunnels is used, the state collapses to the state in which that tunnel was legal.

As a gizmo G, this allows the sequences $[L \to L, A \to B, B \to A]$ and $[L \to L, C \to D, D \to C]$, but not $[L \to L, A \to B, B \to A, C \to D]$. The gizmo corresponding to the superposition state $G[[L \to L]]$ is not reversible:[17] taking the tunnel $A \to B$ collapses the state, and even after returning $B \to A$, $C \to D$ has closed. However, the initial gizmo G is reversible, since it is simulated by the reversible locking 2-toggle. Thus we have the following surprising fact:

Lemma 3.26. *Reversible gizmos are not closed under making traversals. That is, even if G is reversible and $X \in G$, $G[X]$ may not be reversible.*

In general, superpositions can result when merging locations to form a quotient gizmo. If we want our gizmos to always have a 'definite' state, then the states of $G/\!\sim$ are really *subsets* of the states of G, representing the states G could be in depending which path was taken through G among paths indistinguishable in $G/\!\sim$.

3.5.4 Representations of gizmos

We do not have a universal way to describe gizmos, along the lines of state diagrams. In particular, finite state diagrams cannot describe all gizmos; see Section 3.6. Moreover, there are uncountably many gizmos even with just 2 locations,[18] and any finite

[17] See Section 4.2 for the precise definition of reversible gizmo.

[18] For any infinite sequence on $\{A \to B, B \to A\}$, there is a distinct gizmo which allows exactly the prefixes of the sequence with self-loops inserted.

convention can describe at most countably many of them. Fortunately, we are almost exclusively interesting in gizmos which can be represented by finite state diagrams (again, see Section 3.6), of which there are countably many.

3.5.5 Unreachable states

One could consider a gadget which has four locations and four states: the first two states behave like the 2-toggle, and the last two states behave like the mismatched dicrumblers. Consider the gizmo corresponding to the first state of this gadget. This gizmo is exactly state 1 of the 2-toggle. Gizmos only know about legal traversal sequences, and thus cannot detect states which can never be reached. If one is interested in simulations using this four-state gadget, it would be best to allow any of its states to be used; thus S would consist of four gizmos (only two up to isomorphism). More generally, one needs to distinguish between the situation in which a single gizmo is being used for simulations, and the situation in which a gizmo corresponding to each state of a gadget is being used.

3.5.6 Identical ports in simulations

Consider a gadget similar to the 1-toggle (Figure 2-1), but with one location duplicated. That is, in state 1 the traversals $A \to C$ and $B \to C$ both switch to state 2, and in state 2 the traversals $C \to A$ and $C \to B$ both switch to state 1. The gizmo corresponding to state 1 trivially simulates the 1-toggle by merging A and B, or by discarding either of A and B. It seems reasonable to say that the 1-toggle also simulates this 3-location gizmo: just connect A and B to one end of a 1-toggle and C to the other. However, this simulation is not valid, and more generally, one cannot simply connect multiple 'ports' of a simulated gizmo to the same location in the simulation. This is because our notion of connecting locations, the quotient gizmo, identifies the locations rather than leaving two locations with free traversals between them, and because our notion of subgizmo simply takes a subset of locations. If, for instance, we had instead defined subgizmo using an arbitrary 'port-labeling' function, we could make identical ports.

Often when one would like to have identical ports, it is possible to create them, such as by adding a wire (Figure 2-13) to the simulation and using each location of the wire as a port. One advantage of not allowing simulations to freely duplicate ports in this way is that it becomes easier to define 'balanced' gizmos, which we do in Section 4.5: under our definition, the gadget described above is not balanced, so we can prove that the 1-toggle cannot simulate it.

3.6 Regular gizmos

The definition of gizmos allows for gizmos that are exotic infinite sets, not arising from a gadget which can be drawn with a finite state diagram. To prevent this, we

introduce a notion of *regular* gizmos, exactly analogous to regular languages. Most gizmos we consider are regular.

Observe that a state diagram of a gadget can be thought of as describing a finite automaton on the language of traversals, which accepts exactly the legal traversal sequences.

Definition 3.27. A gizmo G is *regular* if $\operatorname{locs} G$ is finite and G is regular when considered as a language over $\mathcal{T}(\operatorname{locs} G)$.

We can think of regular gizmos as those corresponding to gadgets with finitely many states. The collection of states reachable from a gizmo G is $\{G[X] \mid X \in \mathcal{T}(\operatorname{locs} G)^*\}$.

In the NFAs we consider, we do not allow transitions on the empty string ε. All of our DFAs and NFAs will have alphabet $\mathcal{T}(L)$ for some location set L. We use $\mathcal{L}(D)$ to express the language recognized by an automaton D. We use the notation $t : s \to s'$ to say that an NFA or DFA has a transition from state s to s' when reading symbol t; since t is a traversal, this often looks like $a \to b : s \to s'$.

Lemma 3.28. *Let G be a gizmo with finitely many locations. Then G is regular if and only if $\{G[X] \mid X \in \mathcal{T}(\operatorname{locs} G)^*\}$ is finite.*

This is exactly a specialization of the Myhill-Nerode theorem [Ner58]. We provide a proof for completeness.

Proof. Suppose G is regular, so there is a DFA D which recognizes G. If two traversal sequences X, Y on $\operatorname{locs} G$ put D in the same state, then $G[X] = G[Y]$:

$$Z \in G[X] \iff D \text{ accepts } XZ \iff D \text{ accepts } YZ \iff Z \in G[Y].$$

Since D has finitely many states, there are finitely many options for $G[X]$.

Now suppose $T = \{G[X] \mid X \in \mathcal{T}(\operatorname{locs} G)^*\}$ is finite. We construct a DFA on $\mathcal{T}(\operatorname{locs} G)$ with states T. For each $G' \in T$ and $a \to b \in \mathcal{T}(\operatorname{locs} G)$, the DFA has a transition $a \to b : G' \to G'[[a \to b]]$. We set each state G' to accept if and only if $[\,] \in G'$. The result of feeding X into this DFA starting in state G is $G[X]$, which is accepting if and only if $X \in G$. So the DFA recognizes G, and thus G is regular. □

Regular prefix-closed gizmos are recognized by an NFA with a specific structure, which we will take advantage of in Chapter 5.

Lemma 3.29. *If G is a nonempty regular prefix-closed gizmo, G is recognized by an NFA where every state is accepting.*

Proof. Begin with the DFA constructed by the proof of Lemma 3.28, which has one state for each reachable state $G[X]$. Since G is prefix-closed, if $G[X]$ is nonempty then $[\,] \in G[X]$. In particular, the only rejecting state is the empty gizmo $\emptyset$, and all transitions from $\emptyset$ stay there. Remove this rejecting state and all transitions to it to obtain the desired NFA. □

This makes more explicit the relationship between prefix-closed gizmos and gadgets with full target set (for reachability): the only way the NFA can reject is by making an illegal traversal, and then there is no way to fix it. The operation of removing the rejecting state is simple enough that it preserves all of the other properties of NFAs we consider. Hereafter, all NFAs we discuss for prefix-closed gizmos have all accepting states.

Theorem 3.30. *Regular gizmos are closed under finite simulation.*

Proof. We consider each operation in simulation, and construct an NFA for the resulting gizmo using NFAs for the input gizmos. The correctness of each NFA follows immediately from the definitions of the operations.

- Let I be a finite set and suppose D_i is an NFA recognizing G_i for each $i \in I$. We build a product DFA whose states are tuples containing a state of each D_i. For each transition $a \to b : s \to s'$ of D_i, we include a transition $a \to b : (\dots, s, \dots) \to (\dots, s', \dots)$. A state $(s_i)_{i \in I}$ is accepting if each s_i is.

- Suppose D recognizes G and $\sim$ is an equivalent relation on $\operatorname{locs} G$. We build an NFA for $G/\!\sim$ in two steps. First, change the alphabet to $\mathcal{T}(\operatorname{locs} G/\!\sim)$ by replacing label t of a transition with $\pi_{\sim\circ}(t)$. Then ensure transitive closure: whenever there are transitions $a \to b : s_1 \to s_2$ and $b \to c : s_2 \to s_3$, add a transition $a \to c : s_1 \to s_3$.

- Suppose D recognizes G and $L \subset \operatorname{locs} G$. To build an NFA for $G|_L$, simply remove all transition for traversals not in L from D.

If we start with all DFAs, only the quotient can result in an NFA. This reflects that quotients result in superpositions, as discussed in Section 3.5.3. □

Lemma 3.31. *Regular gizmos are* not *closed under arbitrary simulation.*

Proof. Let G be the gizmo corresponding to state 1 of the ordered dicrumblers (Figure 2-9); clearly G is regular. We name the four locations of G with letters A through D such that $[A \to B, C \to D] \in G$. Let

$$X = [X_0, X_1, \dots] = [x_0 \to y_0, x_1 \to y_1, \dots]$$

be a nonconstant infinite sequence on $\{A \to B, C \to D\}$. We will show that G can arbitrarily simulate the minimal gizmo G_X on $\{A, B, C, D\}$ containing every prefix of X. This minimal gizmo contains exactly the prefixes of X with self-loops inserted.

First, we will use G to arbitrarily simulate a gizmo G_∞ with countably many locations $\{A_i, B_i\}_{i \in \mathbb{N}}$, which is the minimal gizmo containing every sequence $[A_0 \to B_0, \dots, A_n \to B_n]$. Think of G_∞ as having an infinite sequence of directed tunnels, which must be traversed in order.

To simulate G_∞ using G, first take $I = \mathbb{N}$, and $G_i = G$ for all i. We will describe a location of $\bigotimes_{i \in \mathbb{N}} G_i$ using subscripts; e.g. A_0 is the copy of location A in G_0. Let

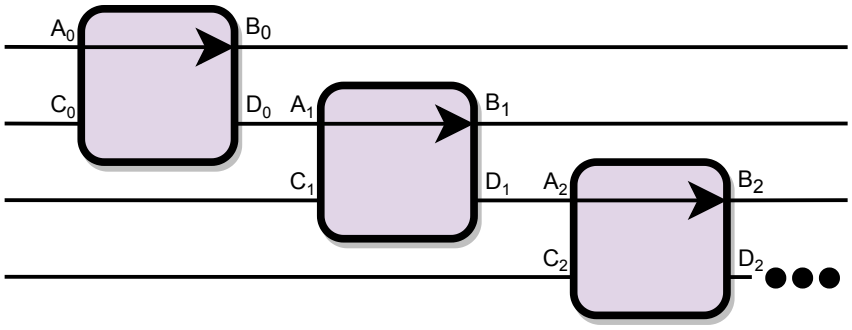

Figure 3-4: The arbitrary simulation of G_∞ using ordered dicrumblers. This pattern continues infinitely. The only nontrivial sequences of traversals that can be made is $[A_0 \to B_0, C_0 \to B_1, C_1 \to B_2, C_2 \to B_3, \dots]$.

$D_i \sim A_{i+1}$ for all i, and let

$$L = \{B_i \mid i \in \mathbb{N}\} \cup \{C_i \mid i \in \mathbb{N}\} \cup \{A_0\}.$$

Then $\left(\bigotimes_{i\in\mathbb{N}} G_i\right) \Big/ \sim \Big|_L \cong G_\infty$, where the isomorphism sends $A_0 \mapsto A_0$, $C_i \mapsto A_{i+1}$, and $B_i \mapsto B_i$. This simulation is shown in Figure 3-4.

Next we use G_∞ to simulate G_X. This is relatively simple: set $A_i \sim A_j$ and $B_i \sim B_j$ when $X_i = X_j$. The resulting quotient has four locations because X is nonconstant; and the isomorphism will be given by $[A_i] \mapsto x_i$, $[B_i] \mapsto y_i$. This merges the infinitely many tunnels in G_∞ down to the two tunnels in G_X, such that the ith tunnel becomes the tunnel that is supposed to be used on the ith (non self-loop) traversal. Thus we have $G_\infty/\sim = G_X$, and hence G arbitrarily simulates G_X.

By Lemma 3.28, G_X is regular if and only if X is eventually periodic. Taking X to be any sequence which is not eventually periodic, we have that G arbitrarily simulates a nonregular gizmo. □

We will often discuss both traversals and transitions, which may be easy terms to confuse. A *traversal* looks like $a \to b$, and is a member of the alphabet on of our gizmos or NFAs uses. A *transition* is meaningless for a gizmo on its own; it has the form $t : s \to s'$ for a traversal t, and describes how an NFA can change state. For an NFA that recognizes a gizmo, there may be several transitions corresponding to the same traversal.

Chapter 4

Unsimulability

In this chapter, we prove several results implying that some gizmos cannot simulate other gizmos. These take the form of showing that a class $\mathcal{C}$ of gizmos is closed under simulation, and sometimes closed under arbitrary simulation. Such a result provides a negative answer to Question 3.1 when $S \subset \mathcal{C}$ and $G \notin \mathcal{C}$. We have already seen one result of this form: Theorem 3.30, that regular gizmos are closed under simulation.

First, we will show that an infinite family of gizmo classes, defined by "implication properties," are each closed under arbitrary simulations. In particular, this family includes the prefix-closed gizmos. Then we will see several gizmo classes which, like regular gizmos, are closed under finite simulation but not arbitrary simulation.

While we are primarily interested in simulations between regular prefix-closed gizmos, most of the results in this chapter apply to non-prefix-closed gizmos. Similarly, many of our results apply even to nonregular gizmos and arbitrary simulations.

To prove that a class $\mathcal{C}$ of gizmos is closed under simulations, we must show that performing each of the operations in a simulation—tensor product, quotient, subgizmo, and isomorphism—on gizmos in $\mathcal{C}$ yields a gizmo also in $\mathcal{C}$. That isomorphism preserves $\mathcal{C}$ will always be trivial since our definitions of gizmo classes are invariant under isomorphism. When proving $\mathcal{C}$ is closed under finite simulation, we need only consider finite tensor products, but for arbitrary simulations we must allow arbitrary tensor products.

Beyond closure under simulation, another property of gizmo classes that is sometimes relevant is whether gizmos stay in the class after making traversals.

Definition 4.1. A gizmo class $\mathcal{C}$ is *closed under traversals* if for every $G \in \mathcal{C}$ and $X \in \mathcal{T}(\operatorname{locs} G)^*$, also $G[X] \in \mathcal{C}$.

We already saw in Section 3.5.3 that a gizmo class which—as we will soon show—is closed under simulations need not be closed under traversals. But many classes closed under simulation also happen to be closed under traversals.

Lemma 4.2. *Prefix-closed gizmos are closed under traversals.*

Proof. Let G be a prefix-closed gizmo, and let X be a traversal sequence on $\operatorname{locs} G$. Then $YZ \in G[X] \implies XYZ \in G \implies XY \in G \implies Y \in G[X]$, so $G[X]$ is prefix-closed. □

Lemma 4.3. *Regular gizmos are closed under traversals.*

Proof. If a DFA D recognizes G, then the DFA which is D with start state changed to the state resulting from feeding X into D recognizes $G[X]$. □

To show a class is closed under traversals it suffices to consider the case when X consists of a single traversal; the general case follows by induction.

4.1 Implication properties

Our first results in this chapter concern an infinite family of gizmo classes. This is a natural generalization of properties like prefix-closed, which can be expressed in a particular convenient form. The infinite family is based on the crucial facts about these properties that are used to prove that they are closed under simulations, so we are able to prove a much more general result.

Definition 4.4. Let $X = [a_1 \to b_1, \ldots, a_n \to b_n]$ be a traversal sequence on L. Its *reverse* X^{-1} is the traversal sequence $[b_n \to a_n, \ldots, b_1 \to a_n]$ on L.

The reverse is intuitively the traversal sequence resulting from walking through X backwards.

Definition 4.5. Let Σ be a finite set of *variables*. A *traversal formula* is a finite sequence of variables and their formal inverses, such as $XX^{-1}Y$ or XX (which we abbreviate X^2) if $X, Y \in \Sigma$. For a gizmo G, a *variable assignment* is a function $\Sigma \to \mathcal{T}(\operatorname{locs} G)^*$. The *evaluation* $f[t]$ of a traversal formula t under a variable assigment f is the traversal sequence obtained by replacing each variable in the formula with its assignment, interpreting inverse as the reverse sequence and concatenating. A set S of traversal formulas is *simple* if each element of Σ appears exactly once in exactly one element of S, and no inverses appear in elements of S. For a simple set S of traversal formulas and a traversal formula x, the *implication property* $S \implies x$ is the following statement about a gizmo G: for any variable assignment for G, if the evaluation of each element of S is in G, then the evaluation of x is in G. An implication property defines a class of gizmos, namely the gizmos for which $S \implies x$ is true.

We generally write S as a comma-separated list, without braces. For instance, the implication property $XY \implies X$ says of a gizmo G that whenever $XY \in G$, also $X \in G$. This is exactly the statement that G is prefix-closed, so the prefix-closed gizmos are described by $XY \implies X$.

We often use implication properties as adjectives, using brackets for readability. For instance, a '$[X \implies X^{-1}]$ gizmo' is a gizmo satisfying $X \implies X^{-1}$.

The hypothesis that S is simple is necessary for the resulting class to be closed. For instance, the mismatched dicrumblers (2-5) satisfies $XX \implies XXX$, which is *not* an implication property since X appears twice in XYX. But as we will see in Corollary 5.9, the mismatched dicrumblers simulates every regular gizmo, including ones which do not satisfy $XX \implies XXX$.

it is important to note that the variables in an implication property can stand for arbitrary traversal *sequences*, not just single traversals. This means implication properties are quite strong, and checking whether a specific gizmo satisfies one is not trivial.

For an implication property $S \implies x$, we can assume without loss of generality that S is finite: finitely many variables (counting inverses) appear in x, and each appears it only one element of S. Take the finite set $S' \subset S$ of traversal formulas in S containing a variable that appears in x, and remove all variables in Σ which only appear in elements of $S \setminus S'$. Then $S' \implies x$ is equivalent to $S \implies x$.[19]

Theorem 4.6. *For any implication property $S \implies x$, the class of gizmos satisfying $S \implies x$ is closed under arbitrary simulation.*

Proof. We consider each operation in simulation.

- Let I be an arbitrary index set, and suppose each G_i satisfies $S \implies x$ for $i \in I$. Let $G = \bigotimes_{i \in I} G_i$. Consider a variable assignment $f : \Sigma \to \mathcal{T}(\operatorname{locs} G)^*$, and assume the evaluation under f of each element of S is in G. For each $i \in I$, there is an induced variable assignment $f_i : \Sigma \to \mathcal{T}(\operatorname{locs} G_i)^*$ with $f_i : X \mapsto f(X)|_{G_i}$. For $X \in \Sigma$, since S is simple, X appears in some $t \in S$. So $f[t]$ contains $f(X)$ as a substring and is in G, so each traversal in $f(X)$ is on a single G_i. Thus for any traversal formula t, $f[t]$ is an interleaving of $f_i[t]$, and in particular $f_i[t] = f[t]|_{G_i}$.

  For each $t \in S$ and $i \in I$, since $f[t] \in G$, we have $f_i[t] \in G_i$. But G_i satisfies $S \implies x$, so $f_i[x] \in G_i$. Since $f[x]$ is an interleaving of $f_i[x]$, we have $f[x] \in G$.

- Suppose G satisfies $S \implies x$ and $\sim$ is an equivalence relation on $\operatorname{locs} G$. Let $f : \Sigma \to \mathcal{T}(\operatorname{locs} G/\!\sim)^*$ be a variable assignment such that for every $t \in S$, $f[t] \in G/\!\sim$. So $\pi_{\sim \circ}^*(X_t) \rightsquigarrow f[t]$ for some $X_t \in G$, and we can decompose X_t into sequences leading to the evaluations of individual variables in $f[t]$. Since each variable in Σ appears exactly once in some $t \in S$, this gives a variable assignment $\widetilde{f} : \Sigma \to \mathcal{T}(\operatorname{locs} G)^*$ such that $\widetilde{f}[t] = X_t \in G$ for each $t \in S$, and $\pi_{\sim \circ}^*(\widetilde{f}(X)) \rightsquigarrow f(X)$ for each $X \in \Sigma$. Thus $\pi_{\sim \circ}^*(\widetilde{f}[t]) \rightsquigarrow f[t]$ for any traversal formula t. Since G satisfies $S \implies x$, $\widetilde{f}[x] \in G$, but then $f[x] \in G/\!\sim$.

- Suppose G satisfies $S \implies x$ and $L \subset \operatorname{locs} G$. Let $f : \Sigma \to \mathcal{T}(L)^*$ be a variable assignment such that for every $t \in S$, $f[t] \in G|_L$. Then $f[t] \in G$, so $f[x] \in G$, and since $f[x]$ is on L also $f[x] \in G|_L$.

□

Here is a more intuitive but less careful argument for this theorem: suppose we have a network of gadgets all satisfying $S \implies x$ and an assignment f of variables to paths through the network, where the concatenation of these paths $f[t]$ is legal for

[19] This fails only for the trivial case with the empty gizmo and an implication property of the form $S \implies [\,]$.

each $t \in S$. If we restrict to any particular gadget, the restricted path $f[t]|_{G_i} = f_i[t]$ is still legal, so by hypothesis the path $f_i[x]$ is legal for that gadget. But then the path $f[x]$ corresponding to x is legal according to every gadget, so it is legal over all. The assumption that S is simple is needed to obtain a unique assignment for each variable: if X appears multiple times in S, the different occurrences might represent different paths through the network, and we would not obtain consistent variable assignments for the individual gadgets.

Since prefix-closed gizmos are exactly $[XY \implies X]$ gizmos, we have an immediate corollary:

Corollary 4.7. *Prefix-closed gizmos are closed under arbitrary simulation.*

This fact helps justify considering simulation specifically between prefix-closed gizmos. If some class $\mathcal{C}$ of prefix-closed gizmos is closed within prefix-closed gizmos, meaning it cannot simulate any prefix-closed gizmos outside $\mathcal{C}$, then it is also closed for general gizmos.

If we interpret prefix-closed gizmos as representing gadgets for reachability (as opposed to targeted set reconfiguration), then it seems obvious that prefix-closed gizmos should be closed under simulation: if none of the gadgets available care what state they end in, how could we possibly simulate a gadget which does care?

Some gizmo classes defined by implication properties, including prefix-closed gizmos (Lemma 4.2), are closed under traversals, while others, including gizmos satisfying $X, Y \implies XX^{-1}Y$ (see Lemma 3.26), are not.

4.2 Reversible gizmos

Next, we consider the class of gizmos corresponding to reversible gadgets [Lyn20, DHL20, DGLR18].

Definition 4.8. An NFA with alphabet $\mathcal{T}(L)$ is *reversible* if for every transition $a \to b : s \to s'$, there is a reverse transition $b \to a : s' \to s$.[20] A gizmo is *reversible* if it is recognized by a reversible NFA.

This definition is essentially the same as earlier definitions of reversible gadgets, except that we more clearly distinguish between gizmos and NFAs. This definition has the issue that it can be hard to tell if a given gizmo is reversible, since only some of the many NFAs recognizing a gizmo might be reversible; see the discussion in Section 3.5.3. In particular, after making a traversal, the NFA may transition to multiple states, resulting in a nonreversible gizmo; see Lemma 3.26.

We would prefer to have a definition which does not rely on NFAs, but finding one has proven difficult. it is easy to show that any reversible gizmo satisfies the implication properties $X, Y \implies XX^{-1}Y$ and $XYZ \implies XYY^{-1}YZ$. Can we go the other way?

[20] Reversible NFAs are unrelated to the more widely studied reversible DFAs, where there are never multiple transitions for the same symbol leading into a state. Reversible DFAs are more related to the reverse-deterministic gadgets of Lynch [Lyn20], which do not correspond a class of gizmos.

Conjecture 4.9. *Every regular gizmo satisfying both* $X, Y \implies XX^{-1}Y$ *and* $XYZ \implies XYY^{-1}YZ$ *is recognized by a reversible NFA.*

This would be useful even restricted to prefix-closed gizmos. Since we do not know how to describe reversible gizmos in terms of implication properties, we must prove closure under simulation separately.

Theorem 4.10. *Reversible gizmos are closed under simulation.*

Proof. It suffices to show that each NFA operation described in the proof of Theorem 3.30 preserves reversibility of NFAs, since these operations correspond to simulation by the recognized gizmos. Inspecting each case, it is clear that if the input NFAs are reversible, than the resulting NFA will be as well. One way to see this is that if we replace each $\to$, in both traversals and state transitions, with $\leftrightarrow$, the operations act the same on reversible NFAs. □

Reversibility, like an implication property, is intuitively 'local,' and thus we expect it to be preserved by arbitrary simulation as well. However, this fails because we have required reversible gizmos to be recognized by NFAs, so any nontrivial infinite tensor product of reversible gizmos will not be reversible (or regular). To fix this, we can consider instead automata which are allowed to have infinitely many states, which we do in Chapter 6. This is an additional reason it would be preferable to have an alternative definition of reversible.

4.3 Strictly bounded gizmos

Our next classes are defined by bounding the number of 'interesting' traversals we can make. This corresponds to DAG or LDAG gadgets [DHL20, Lyn20], depending on what we mean by 'interesting.' These classes will turn out to be closed under finite but not arbitrary simulation, since the bound for a tensor product becomes infinite when there are infinitely many multiplicands.

Definition 4.11. A gizmo G is *strictly bounded by* a number n if every traversal sequence $X \in G$ contains at most n traversals other than self-loops $a \to a$.

A gizmo is *strictly bounded* if it is strictly bounded by any number.

Lemma 4.12. *Strictly bounded gizmos are closed under traversals.*

Proof. If G is strictly bounded by n and $Y \in G[X]$, then XY has at most n non-self-loop traversals, but Y cannot have more such traversals than XY; thus $G[X]$ is strictly bounded by n. □

Strictly bounded gizmos correspond to DAG gadgets [DHL20, Lyn20].

Theorem 4.13. *Strictly bounded gizmos are closed under simulation.*

Proof. We consider each operation in simulation.

- Let I be a finite set, and suppose G_i is strictly bounded by n_i for each $i \in I$. Let $X \in \otimes_{i\in I} G_i$. Each non-self-loop traversal in X corresponds to exactly one non-self-loop traversal in some $X|_{\operatorname{locs} G_i}$. But $X|_{\operatorname{locs} G_i} \in G_i$ has at most n_i such traversals, so X has at most $\sum_{i\in I} n_i$ traversals other than self-loops. Thus $\otimes_{i\in I} G_i$ is strictly bounded by $\sum_{i\in I} n_i$.

- Suppose G is strictly bounded by n, and let $\sim$ be an equivalence relation on $\operatorname{locs} G$. Let $X \in G/\!\sim$. Then there is some $\widetilde{X} \in G$ with $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X$. By assumption, $\widetilde{X}$ has at most n non-self-loop traversals. Since $\pi_{\sim\circ}$ preserves self-loops, $\pi^*_{\sim\circ}(\widetilde{X})$ also has at most n non-self-loop traversals. Transitive contraction cannot increase the number of non-self-loop traversals, so X also has at most n of them. So $G/\!\sim$ is strictly bounded by n.

- Suppose G is strictly bounded by n, and $L \subset \operatorname{locs} G$. Any $X \in G|_L$ is also in G, and hence has at most n traversals other than self-loops.

□

4.4 Weakly bounded gizmos

The key characteristic of the LDAG gadgets defined by Lynch [Lyn20] is that they can change state boundedly many times. Unfortunately for us, changing state is not a natural notion for gizmos, and in fact some gizmos corresponding to LDAG gadgets can change state arbitrarily many times, in the sense that $G[XY] = G \neq G[x]$. For instance, let G be the minimal gizmo containing $(A|B)^*AC$ where the letters are disjoint traversals; this gizmo describes an LDAG gadget, but $G[AB] = G \neq G[A]$.

It seems that defining gizmos corresponding to LDAG gadgets directly would require another layer of abstraction, to keep track of which traversals are considered 'state-changing.' Instead, we resort to state machines, using essentially the same definition as Lynch [Lyn20].

Definition 4.14. An NFA is *weakly acyclic* if it has no cycles containing multiple states. A gizmo is *weakly bounded* if it is recognized by an acyclic NFA.

We say 'weakly' since we allow loops on single states, and there is a stricter notion which defines NFAs which describe strictly bounded gizmos (see Lemma 5.12).

For the example gizmo containing $(A|B)^*AC$, there is a weakly acyclic NFA with three states and transitions $A, B : 1 \to 1$, $A : 1 \to 2$, and $C : 2 \to 3$. The apparent state changing when fed AB is a superposition which is entered and then collapsed, which can occur arbitrarily many times despite each path having bounded length (ignoring loops).

We will show in Corollary 5.13 that strictly bounded gizmos are also weakly bounded.

Lemma 4.15. *Weakly bounded gizmos are closed under traversals.*

Proof. If G is recognized by a weakly acyclic NFA D, we can construct a weakly acyclic NFA for $G[X]$ as follows. Let S be the set of states D can end in when fed X. We add a new state S representing the superpositions of all states in S. For each $s \in S$ and transition $t : s \to s'$, we have a transition $t : S \to s'$. The start state is S, which is accepting if any $s \in S$ is. The original NFA is weakly acyclic and there are no transitions to S, so the new NFA is weakly acyclic. □

Because our definition is based on state machines, many of the same issues arise as for reversible gizmos when we attempt to adapt to arbitrary simulation. However, in this case it is not as much of a problem: since LDAG and weakly bounded are fundamentally about finiteness, LDAG gadgets are not naturally closed under infinite simulations. In fact, Theorem 6.13 shows that weakly bounded gizmos can arbitrarily simulate all (prefix-closed) gizmos.

Theorem 4.16. *Weakly bounded gizmos are closed under simulation.*

Proof. Each operation in the proof of Theorem 3.30 preserves weakly acyclic NFAs. □

Lemma 4.17. *Weakly and strictly bounded gizmos are* not *closed under arbitrary simulation.*

Proof. Consider the arbitrary simulation from Lemma 3.31. Take $X = [A \to B, C \to D]^\infty$ to alternate between the two traversals. Then G_X is the mismatched dicrumblers (Figure 2-5). So we have a simulation from the ordered dicrumblers, which is both weakly and strictly bounded, to the mismatched dicrumblers, which is neither. □

4.5 Balanced gizmos

Our next gizmo class distinguishes the 2-toggle (Figure 2-6) from the locking 2-toggle (Figure 2-8). Both are reversible, and it is easy to simulate a locking 2-toggle with 2-toggles (by protecting each tunnel with a 1-toggle), but not the other way around. We will see that the locking 2-toggle is balanced but the 2-toggle is not, so it is impossible to simulate the 2-toggle with locking 2-toggles. Intuitively, the relevant property of the locking 2-toggle is that if the agent enters some location, it must exit that location before it enters it again. Balanced gizmos make this explicit, and allow any constant number of consecutive entrances to 'build up' before they need corresponding exits.

Definition 4.18. For a location set L, let $X \in \mathcal{T}(L)^*$, and $a \in L$. The *deficit* $\operatorname{def}_a X$ at a in X in the net number of times a is entered; that is, the number of traversals in X of the form $a \to \cdot$ minus the number of the form $\cdot \to a$. The *total deficit* of X is the sum of the absolute deficits at each location $\operatorname{def} X = \sum_{a \in L} |\operatorname{def}_a X|$.

A gizmo G is *balanced* if its *maximum total deficit* $\max_{X \in G} \operatorname{def} X$ is finite.

For gizmos with finitely many locations, it is equivalent to say that the absolute deficit $|\operatorname{def}_a X|$ at each location is bounded.

Theorem 4.19. *Balanced gizmos are closed under simulation.*

Proof. We consider each operation in simulation.

- Let I be a finite set, and suppose G_i has maximum total deficit n_i for each $i \in I$. Consider a traversal sequence $X \in \bigotimes_{i \in I} G_i$. Then $X|_{\operatorname{locs} G_i} \in G_i$, so

$$\operatorname{def} X = \sum_{i \in I} \operatorname{def} X|_{\operatorname{locs} G_i} \leq \sum_{i \in I} n_i.$$

  Hence $\bigotimes_{i \in I} G_i$ has maximum total deficit at most $\sum_{i \in I} n_i$, which is finite.

- Suppose G has maximum total deficit n, and let $\sim$ be an equivalence relation on $\operatorname{locs} G$. Contracting $[a \to b][b \to c] \rightsquigarrow [a \to c]$ removes one entrance and one exit at b, which does not change the deficit at b. Thus if $X \rightsquigarrow Y$, $\operatorname{def} X = \operatorname{def} Y$. Let $X \in G/\!\sim$, so $\pi_{\sim \circ}^*(Y) \rightsquigarrow X$ for some $Y \in G$. Then $\operatorname{def} X = \operatorname{def} \pi_{\sim \circ}^*(Y) \leq \operatorname{def} Y \leq n$ by the triangle inequality, so $G/\!\sim$ has maximum total deficit at most n.

- Suppose G has maximum total deficit n and $L \subset \operatorname{locs} G$. If $X \in G|_L$, then $X \in G$ so $\operatorname{def} X \leq n$. Thus $G|_L$ has maximum total deficit at most n.

□

Lemma 4.20. *Balanced gizmos are not closed under arbitrary simulation.*

Proof. We can use exactly the same counterexample as for bounded gizmos, in the proof of Lemma 4.17. The ordered dicrumblers is balanced, with maximum total deficit 4, but the mismatched dicrumblers is not. □

Lemma 4.21. *Balanced gizmos are closed under traversals.*

Proof. Suppose G is a balanced gizmo with maximum total deficit k. Let X be a traversal sequence on $\operatorname{locs} G$. We claim $G[X]$ has maximum total deficit at most $k + \operatorname{def} X$, and thus $G[X]$ is balanced.

Let $Y \in G[X]$. Then $XY \in G$, so $\operatorname{def} XY \leq k$. Clearly $\operatorname{def}_a Y = \operatorname{def}_a XY - \operatorname{def}_a X$, so by the triangle inequality and summing over the locations on G, we have $\operatorname{def} Y \leq \operatorname{def} XY + \operatorname{def} X \leq k + \operatorname{def} X$. □

Chapter 5

Universal simulation

In this chapter, we provide positive answers to many instances of Question 3.2. All gizmos we consider in this chapter are prefix-closed, regular, and nonempty. Among prefix-closed gizmos, nonempty gizmos are those satisfying $\emptyset \implies [\,]$, so nonempty prefix-closed gizmos are closed under arbitrary simulation. We will show, for several classes $\mathcal{C}$, that a specific small set S of gizmos simulates every nonempty regular prefix-closed gizmo in $\mathcal{C}$. Note that if $\mathcal{C}$ is closed under finite simulation, which it will be in each of our results, then so is the set of regular prefix-closed gizmos in $\mathcal{C}$. If in addition $S \subset \mathcal{C}$, which again will be the case for us, we exactly characterize the gizmos which can be simulated by S: they are precisely the nonempty regular prefix-closed gizmos in $\mathcal{C}$. Thus we are able to answer every instance of Question 3.2 a particular value of S.

Definition 5.1. A set S of gizmos is *universal* for a class $\mathcal{C}$ of gizmos if S simulates every nonempty gizmo in $\mathcal{C}$. Similarly, S is *arbitrarily universal* if S arbitrarily simulates every element of $\mathcal{C}$.

We avoid empty gizmos because they would otherwise be an uninteresting technical detail in each of our universality results. Empty gizmos are boring; there is only one empty gizmo on each set of locations, and there is little to say about simulations of empty gizmos. We shall say what there is: it is easy to show that empty gizmos and and gizmos containing $[\,]$ are each closed under arbitrarily simulation. Any gizmo G not containing $[\,]$ simulates $(G \otimes G)|_{\operatorname{locs} G \times \{0\}}$, which is the empty gizmo on $\operatorname{locs} G \times \{0\}$. The empty gizmo with one location can simulate any finite empty gizmo and arbitrarily simulate any empty gizmo.

We often say that a gadget is universal for a class to mean that the set of gizmos corresponding to the states of that gadget is universal. If S is finite, then the tensor product $\bigotimes S$ of the gizmos in S can be simulated by S and can simulate each element of S, so $\bigotimes S$ alone is universal as well. Thus, unless we aim to minimize our universal gizmos, finding any finite universal set is as good as finding a single universal gizmo. All of the universal sets in our results will be small and contain only relatively simple, "natural" gizmos.

Universality is very closely analogous to the notion of hardness for a complexity class in complexity theory; we have replaced classes of decision problems with classes

of gizmos, and reductions with simulations.[21] One of the crucial facts in complexity theory—that reductions preserve hardness—also has an analog for gizmos.

Lemma 5.2. *Suppose a set S is (resp. arbitrarily) universal for a class $\mathcal{C}$, and another set S' (resp. arbitrarily) simulates every gizmo in S. Then S' is also (resp. arbitrarily) universal for $\mathcal{C}$.*

Proof. This follows from transitivity of simulation: if $H \in \mathcal{C}$, then S simulates H, so S' also simulates H. □

As in complexity theory, once we know of one universal set S of gizmos for a class, it is easy to find more by simply showing they simulate each gizmo in S. For instance, we will prove (Theorem 5.8) that the door (Figure 2-3) is universal for prefix-closed regular gizmos; the mismatched dicrumblers (Figure 2-5) simulates the door [ABD+20], so it is also universal for prefix-closed regular gizmos.

While this chapter considers only finite simulation, all of the discussion above applies equally well to arbitrary simulation. In Chapter 6, we adapt some of the results in this chapter to arbitrary simulation; generally this involves removing the assumption the assumption that gizmos are regular, since regular gizmos are only closed under finite simulation.

We begin with some abstract results in Section 5.1 which will help to prove our simulations correct. In the remaining subsections, we prove several universality results, in roughly decreasing order of the power of the gizmo classes involved.

5.1 How to verify simulations

Before we prove specific universality results, we will develop a bit of theory which makes it easier to discuss and prove correctness of simulations.

Definition 5.3. Let $S = (I, \{G_i\}_{i\in I}, \sim, L)$ be a simulation. Let $G_\otimes = \bigotimes_{i\in I} G_i$ and $G = G_\otimes/\sim|_L$, so G is the simulated gizmo. A *basis* for S is a set $\mathcal{B} \subset \mathcal{T}(\operatorname{locs} G_\otimes)^*$ of traversal sequences such that

1. For each $X \in G$, there is a sequence $B_1, \ldots, B_k$ of elements of $\mathcal{B}$ and a traversal sequence $\widetilde{X}$ such that $B_1 \cdots B_k \in G_\otimes$, we can construct $\widetilde{X}$ by inserting self-loops into $B_1 \cdots B_k$, and $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X$.

2. For each $B \in \mathcal{B}$, for some $a, b \in L$, $\pi^*_{\sim\circ}(B) \rightsquigarrow [a \to b]$.

3. For each $B = [a_1 \to b_1, \ldots, a_k \to b_k] \in \mathcal{B}$, the only locations in $\pi^*_{\sim\circ}(B)$ which are in L are the very first and last, namely $[a_1]$ and $[b_k]$.

[21] Formally, these both define preorders, and a complete problem or universal gizmo is precisely a maximum element of a downwards-closed subset of the preorder.

An *internal state* of S is a state of $G_\otimes$, i.e. a nonempty gizmo of the form $G_\otimes[X]$. An internal state is *reachable* if it can be written as $G_\otimes[B_1 \cdots B_k]$ for $B_1, \ldots, B_k \in \mathcal{B}$.

The *induced NFA*[22] $D_\mathcal{B}$ has for states the reachable internal states, alphabet $\mathcal{T}(L)$, and a transition $a \to b : s \to s'$ if there is some $B \in \mathcal{B}$ with $s[B] = s'$ and $\pi^*_{\sim\circ}(B) \rightsquigarrow [a \to b]$. A state $G_\otimes[X]$ is accepting if $[\,] \in G_\otimes[X]$. The starting state is $G_\otimes$.

A basis is supposed to capture all the nontrivial paths the agent might need to take through the simulation. For the example simulation in Figure 2-14, one basis contains

$$[A \to B, E \to F, I \to J, R \to Q, L \to K, C \to D, F \to E, G \to H],$$
$$[M \to N, P \to O, L \to K, C \to D, I \to J, R \to Q, O \to P, S \to T],$$

and the reverses of these two sequences.

The NFA induced by a basis is an abstract description of how the basis paths behave and interact. This abstraction makes it easier to reason about simulations, particularly since the simulated gizmo is entirely determined by $D_\mathcal{B}$:

Theorem 5.4. *Let $\mathcal{B}$ be a basis for a simulation $(I, \{G_i\}_{i\in I}, \sim, L)$. Then the simulated gizmo $\bigotimes_{i\in I} G_i/\sim|_L$ is the minimal gizmo containing $\mathcal{L}(D_\mathcal{B})$.*

Proof. We write $G_\otimes = \bigotimes_{i\in I} G_i$ and $G = G_\otimes/\sim|_L$. We first prove a characterization of $\mathcal{L}(D_\mathcal{B})$:

Lemma 5.5. *For any $X = [a_1 \to b_1, \ldots, a_k \to b_k] \in \mathcal{T}(L)^*$ and reachable internal state $G'_\otimes$, $D_\mathcal{B}$ accepts X starting at state $G'_\otimes$ if and only if there is a sequence $B_1, \ldots, B_k \in \mathbb{B}$ such that $B_1 \cdots B_k \in G'_\otimes$ and for each i, $\pi^*_{\sim\circ}(B_i) \rightsquigarrow [a_i \to b_i]$.*

Proof. This only relies on the definition of $D_\mathcal{B}$; we do not even need the fact that $\mathcal{B}$ is a basis. We proceed by induction on X. For $X = [\,]$, $k = 0$, so the only possible sequence B_i is empty. By construction, $D_\mathcal{B}$ accepts $[\,]$ starting at $G'_\otimes$ if and only if $G'_\otimes$ is an accepting state if and only if $[\,] \in G'_\otimes$.

Let $X = [a_1 \to b_1, \ldots, a_k \to b_k] = [a_1 \to b_1]X'$. Suppose $D_\mathcal{B}$ accepts X starting at $G'_\otimes$. Then for some internal state $G''_\otimes$, $D_\mathcal{B}$ has a transition $a \to b : G'_\otimes \to G''_\otimes$ and $D_\mathcal{B}$ accepts X' starting from $G''_\otimes$. This transition means there is some $B_1 \in \mathcal{B}$ with $G'_\otimes[B_1] = G''_\otimes$ and $\pi^*_{\sim\circ}(B_1) \rightsquigarrow [a_1 \to b_1]$. By inductive hypothesis, there is a sequence $B_2, \ldots, B_k \in \mathcal{B}$ where $B_2 \cdots B_k \in G''_\otimes$ and $\pi^*_{\sim\circ}(B_i) \rightsquigarrow [a_i \to b_i]$ for $i \geq 2$. But then $B_1, \ldots, B_k$ is the desired sequence.

Conversely, suppose $B_1, \ldots, B_k$ satisfy the hypothesis. Consider the reachable internal state $G''_\otimes = G'_\otimes[B_1]$. We know $B_2 \cdots B_k \in G''_\otimes$, and for each $i \geq 2$, $\pi^*_{\sim\circ}(B_i) \rightsquigarrow [a_i \to b_i]$. So by inductive hypothesis $D_\mathcal{B}$ accepts X' starting at $G''_\otimes$. But by the definition of $D_\mathcal{B}$, it has a transition $a_1 \to b_1 : G'_\otimes \to G''_\otimes$, so $D_\mathcal{B}$ accepts X starting at $G'_\otimes$. □

[22] If there are infinitely many reachable internal states or locations, this does not define an NFA, but an 'arbitrary NFA' (see Chapter 6). In this chapter it will always be an NFA.

We will show that if $D_{\mathcal{B}}$ accepts X (from the start state $G_\otimes$), then X is in G, and if X is in G, then X can be constructed from some sequence which $D_{\mathcal{B}}$ accepts by inserting self-loops and performing transitive contractions. This is exactly the condition for X to be in the minimal gizmo containing $\mathcal{L}(D_{\mathcal{B}})$. Together these two claims suffice to prove the theorem.

Suppose $D_{\mathcal{B}}$ accepts X. Lemma 5.5 gives us a sequence $B_1, \ldots, B_k$ with each $B_i \in G_\otimes$ and $\pi^*_{\sim\circ}(B_1 \cdots B_k) \rightsquigarrow X$. Thus $X \in G_\otimes/\!\sim$, and since X is on L, also $X \in G$.

Now let $X \in G$. Since $\mathcal{B}$ is a basis, there is a sequence $B_1, \ldots, B_k \in \mathcal{B}$ and a traversal sequence $\widetilde{X}$ such that $B_1 \cdots B_k \in G_\otimes$, we can construct $\widetilde{X}$ be inserting self-loops into $B_1 \cdots B_k$, and $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X$.

Since $\mathcal{B}$ is a basis, we have a_i and b_i such that $\pi^*_{\sim\circ}(B_i) \rightsquigarrow [a_i \to b_i]$. By Lemma 5.5, $D_{\mathcal{B}}$ accepts $X' = [a_1 \to b_1, \ldots, a_k \to b_k]$. We will show that X can be constructed from X' by inserting self-loops and performing transitive contractions.

Consider the sequence of contractions realizing $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X$. Since X is on L, every location in $\pi^*_{\sim\circ}(\widetilde{X})$ not in L must be contracted away, and thus is adjacent to the same location in the next or previous traversal. In particular, since $\mathcal{B}$ is a basis, all locations internal to each B_i must be contracted away. Transitive contractions commute, so we may assume we perform these contractions first. Specifically: first, if there are extra self-loops inserted inside some B_i, contract them with either adjacent traversal. Now the sequence has the form $\bullet B_1 \bullet \cdots \bullet B_k \bullet$, where each $\bullet$ consists of self-loops. We can now safely contract each B_i to $[a_i \to b_i]$. Hence $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X$ factors through $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow \bullet[a_i \to b_i] \bullet \cdots \bullet [a_k \to b_k] \bullet \rightsquigarrow X$, where each $\bullet$ consists of self-loops. Call this intermediate traversal sequence Y. Then we have shown that we can insert self-loops into X' to obtain Y, and perform transitive contractions in Y to obtain X, as desired. □

Corollary 5.6. *If $\mathcal{L}(D_{\mathcal{B}})$ is a gizmo, $\bigotimes_{i \in I} G_i/\!\sim|_L = \mathcal{L}(D_{\mathcal{B}})$.*

Finally, we have a lemma which will help with proving that a set of sequences is a basis.

Lemma 5.7. *Suppose G is a gizmo, $\sim$ an equivalence relation on $\operatorname{locs} G$, and $L \subset \operatorname{locs} G/\!\sim$. Then for any $X \in G/\!\sim|_L$, there is a sequence $\widetilde{X} \in G$ with $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X$, such that each segment of $\widetilde{X}$ which contracts to a single traversal in X cannot be transitively contracted before applying $\pi_\sim$, meaning it contains no consecutive traversals of the form $[a \to b, b \to c]$.*

If, for some location $a \in \operatorname{locs} G$, we have $G[Y[a \to a]] = G[Y]$ for all Y, then we can additionally stipulate that each such segment of $\widetilde{X}$ which has length more than 1 does not contain $a \to a$.

Proof. Begin with the minimum-length $\widetilde{X} \in G$ with $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow X$. Consider each segment of $\widetilde{X}$ which contracts to a single traversal separately. All locations in the segment except for the very first and last must be transitively contracted. Suppose a segment contains $[a \to b, b \to c]$, and let $\widetilde{X}'$ be the sequence with these traversals

replace by $[a \to c]$. Since transitive contractions commute, we can assume the corresponding $[b]$ (after applying $\pi_\sim$) is contracted first. Then after this contraction the sequence is exactly $\pi_{\sim\circ}^*(\widetilde{X'})$, so $\pi_{\sim\circ}^*(\widetilde{X'}) \rightsquigarrow X$. This violates our assumption that $\widetilde{X}$ has minimum length.

Suppose $G[Y[a \to a]] = G[Y]$ for all Y, and a segment contains $[a \to a, b \to c]$ (or symmetrically $[c \to b, a \to a]$). Let $\widetilde{X'}$ be the sequence obtained by removing $a \to a$. Since all internal locations are contracted, we must have $a \sim b$. Assume this contraction happens first; the resulting portion of the sequence is $[[a] \to [c]] = [[b] \to [c]]$, so the resulting sequence is again $\pi_{\sim\circ}^*(\widetilde{X'})$, and similarly we violate the minimum-length assumption. □

Our universality results will generally follow the following structure, to show that S is universal for $\mathcal{C}$:

1. Show that every gizmo G in $\mathcal{C}$ is recognized by an NFA D with a particular property.
2. Construct a simulation using S based on D, using the particular property.
3. Describe a basis $\mathcal{B}$ for the simulation, and show that $D_\mathcal{B} \cong D$.
4. By Corollary 5.6, the simulated gizmo is isomorphic to G.

For clarity, we treat gizmo isomorphisms as equality, and in particular we do not distinguish between the locations of G and the locations of the simulated gizmo, which are technically equivalence classes of the locations of the gizmos in the tensor product.

One may wonder why we do not define gizmo classes in terms of finite automata in the first place, since we show automaton-based characterizations anyway on our way to proving universality results, and automaton-based definitions would more directly align with definitions in prior work on gadgets. Here are three reasons we prefer to use definitions which do not require automata when possible, all related to the fact that the objects we are actually interested in are the sets of legal traversal sequences, not the automata which recognize them:

1. It is often easier to prove closure under simulation, and especially under arbitrary simulation, for gizmo classes defined 'internally.' Implication properties (Section 4.1) are a superb example of this. However, this elegance comes at the cost of having to prove connections to NFAs to unify closure and universality results, and in some cases it is not hard to show that a class of NFAs is closed under the operations corresponding to simulation of the gizmos they recognize (see Theorem 3.30).
2. A gizmo is typically recognized by many different NFAs, not all of which will have the property in question; see for instance the discussion in Section 3.5.3. A significant part of the motivation for the definition of gizmos is to stop distinguishing gadgets which behave identically but are described by different state machines. Definitions beginning 'there exists an NFA such that' obscure the properties of the more fundamental set of legal sequences.

3. To usefully apply automaton-based definitions to nonregular gizmos, we would need to allow the automata to have infinitely many states. We do this in Chapter 6 to achieve universality results, but it adds a layer of complication with more details to check.

In Section 4.2, we begrudgingly defined reversible gizmos in terms of NFAs, but only because we are not aware of a more direct definition.

5.2 All gizmos

Our first universality result concerns *all* regular prefix-closed gizmos. The 'nice' NFAs for regular prefix-closed gizmos are simply those with no rejecting states.

Theorem 5.8. *The door gadget (Figure 2-3) is universal for regular prefix-closed gizmos.*

Recall that when we say the door gadget simulates something, we mean the two gizmos representing the states of the door gadget simulate it. A version of Theorem 5.8 is proved by Ani, Bosboom, et al. [ABD+20], and that proof can be adapted to gizmos. We will present a somewhat different proof, which follows the structure described in Section 5.1, and can more easily be adapted to the case of arbitrary simulation, as we do in Chapter 6.

Proof. [$s \to s'$ **instead of** $s_i \to s_j$**?**] We name the locations on the door O, o, C, c, T, and t as in Ani, Bosboom, et al. [ABD+20]: these stand for 'open,' 'close,' and 'traverse,' corresponding to the green, red, and blue tunnels in Figure 2-3. Capital letters represent entrances and lowercase letters represent exits. So the closed state is represented by the minimal prefix-closed gizmo containing $((C \to c \mid O \to o)^*\ O \to o\ (O \to o \mid T \to t)^*)^*$.

Let G be a nonempty regular prefix-closed gizmo. By Lemma 3.29, G is recognized by an NFA D with only accepting states.

Suppose G has n locations, and D has k states $s_1, \ldots, s_k$, starting state s_1, and $m \le k^2n^2$ transitions. Then we will simulate G using $k + 2m$ doors, labeled i for each state s_i, and $(a \to b, i \to j)$ and $(a \to b, i \to j)'$ for each transition $a \to b : s_i \to s_j$. Door 1 is open, and all others are closed. We use subscripts to indicate the copy of the door a location belongs to.

Informally, door i will be open exactly when the simulated version of G is in state i. We will build a path for each transition $a \to b : s_i \to s_j$ that lets the agent enter at a, close door i, open door j, and exit at b, but only when the simulated G was initially in state i.

First, we identify $t_i \sim C_i$ for each i.

For each transition $a \to b : s_i \to s_j$, we identify $o_{(a\to b,i\to j)} \sim T_i$, $c_i \sim T_{(a\to b,i\to j)}$, $t_{(a\to b,i\to j)} \sim C_{(a\to b,i\to j)}$, $c_{(a\to b,i\to j)} \sim O_{(a\to b,i\to j)'}$, $o_{(a\to b,i\to j)'} \sim O_j$, $o_j \sim T_{(a\to b,i\to j)'}$, and $t_{(a\to b,i\to j)'} \sim C_{(a\to b,i\to j)'}$.

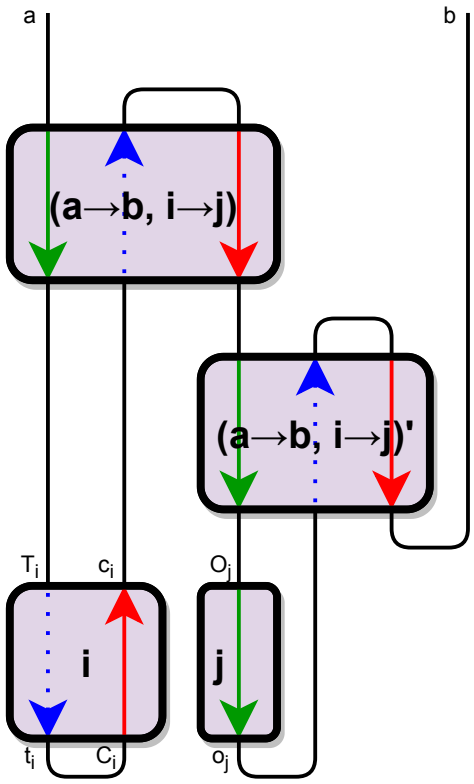

Figure 5-1: How to simulate an arbitrary regular prefix-closed gizmo using doors, showing the four doors involved in implementing a single transition $a \to b : s_i \to s_j$.

Next, for each location $a \in \operatorname{locs} G$, we identify all locations of the form $O_{(\cdot, a \to \cdot)}$ or $c_{(\cdot, \cdot \to a)'}$ into a single location which will correspond to a. We take the subgizmo on this locations, and claim that this gives an isomorphism to G.

The portion of our simulation corresponding to a single transition is shown in Figure 5-1. The two doors $(a \to b, i \to j)$ and $(a \to b, i \to j)'$ are used to prevent 'leaking': the agent can only exit along the traverse and close tunnels of $(a \to b, i \to j)'$ if it crossed the same open tunnel, which requires using the traverse and close tunnels of $(a \to b, i \to j)$, so the agent must have entered on the open tunnel of $(a \to b, i \to j)$. This has the effect of 'duplicating' the tunnels on doors i and j, since each copy of this construction provides an independent way the agent can traverse those tunnels.

To show that this is a simulation of G, we will show that the simulated gizmo is recognized by the same NFA D, by giving a basis and proving that the induced NFA is isomorphic to D. Then Corollary 5.6 proves that our simulation is correct. Let $G^i_\otimes$ be the tensor product of the doors in our simulation with door i open and all others closed, and let G^i be the gizmo our simulation constructs when we make door i open and all others closed. Then the alleged simulation of G simulates G^1, and $G^i = G^i_\otimes / \sim$.

By construction, the only nontrivial paths through the simulation have the form

$$\begin{aligned} a \sim O_{(a\to b, i\to j)} \to o_{(a\to b, i\to j)} \sim T^i \to t^i \sim C_i \to c_i \sim T_{(a\to b, i\to j)} \to t_{(a\to b, i\to j)} \\ \sim C_{(a\to b, i\to j)} \to C_{(a\to b, i\to j)} \sim O_{(a\to b, i\to j)'} \to o_{(a\to b, i\to j)'} \sim O^j \to o^j \\ \sim T_{(a\to b, i\to j)'} \to t_{(a\to b, i\to j)'} \sim C_{(a\to b, i\to j)'} \to c_{(a\to b, i\to j)'} \sim b \end{aligned}$$

for some transition $a \to b : s_i \to s_j$. This is the obvious path through Figure 5-1. We call such a path a *transition path* and denote it by $B_{i\to j}^{a\to b}$. Specifically, $B_{i\to j}^{a\to b}$ is the sequence of traversals (but not $\sim$ relations) above, which is a traversal sequence on $\operatorname{locs} G_\otimes$. Take $\mathcal{B}$ to be the set containing $B_{i\to j}^{a\to b}$ for each transition $a \to b : s_i \to s_j$. We will show that $\mathcal{B}$ is a basis, and then than $D_\mathcal{B} \cong D$. For a transition $a \to b : s_i \to s_j$, clearly $G_\otimes^i[B_{i\to j}^{a\to b}] = G_\otimes^j$, so the reachable internal states are the $G_\otimes^i$ and possibly the empty gizmo.

The second and third properties of a basis are immediate from our definition of $B_{i\to j}^{a\to b}$. To prove the first property, let $X \in G^1$. Then there is a $\widetilde{X} \in G_\otimes^1$ with $\pi_{\sim\circ}^*(\widetilde{X}) \rightsquigarrow X$. Self-loops never affect the traversability of the door, so the sequence $\widetilde{X}'$ obtained by removing all self-loops from X is also in $G_\otimes^1$.

We claim that if for any i and any $Y \in G_\otimes^i$, if $\pi_{\sim\circ}^*(Y) \rightsquigarrow Z \in G^i$ and Y contains no self-loops, then Y is a concatenation of transition paths. To prove this carefully, we consider the first nine traversals in $Y = [a_1 \to b_1, \ldots, a_9 \to b_9]Y'$, show that they are a transition path, and induct on Y'. Since $\pi_{\sim\circ}^*(Y) \rightsquigarrow Z \in G^i$, the equivalence class of the first entrance $[a_1]$ is a location of G, so a_1 has one of the forms $O_{(\cdot, a\to\cdot)}$ or $c_{(\cdot,\cdot\to a)'}$. Only the former starts a non-self-loop traversal, so $a_1 \to b_1$ is the open tunnel on door $(i', a \to b)$ for some i' and b. Then b_1 must be contracted, so $a_2 = b_1$. The only location related to a_2 by $\sim$ which starts a non-self-loop traversal is $T_{i'}$, so $a_2 \to b_2$ is the traverse tunnel on door i'. Moreover, this traversal is only legal if door i' is open, so $i' = i$. Continuing in this fashion, $a_3 \to b_3$ is the close tunnel on door i, and $a_4 \to b_4$ must be the traverse tunnel on some door $(i, \dot{\to})$, but the only open one is $(a \to b, i \to j)$ which we just opened. The remaining five traversals are similar, and must complete the transition path $B_{i\to j}^{a\to b}$. Now the equivalence class of b_9 is a location of G, so we cannot argue that b_9 must be contracted. But we have shown that the first nine traversals of Y are a transition path $B_{i\to j}^{a\to b}$, and after traversing them the state is $G_\otimes^i[B_{i\to j}^{a\to b}] = G_\otimes^j$, so by induction Y consists entirely of transition paths.

In particular, $\widetilde{X}'$ consists entirely of transition paths. So we can write it as $\widetilde{X}' = B_1 \ldots B_k \in G_\otimes^1$ for $B_i \in \mathcal{B}$. By the definition of $\widetilde{X}'$, we can construct $\widetilde{X}$ by inserting self-loops into $B_1 \ldots B_k$. Thus B_i and $\widetilde{X}$ are a witness to the first property of a basis.

Now we show that $D_\mathcal{B} \cong D$. The (nonempty) states of $D_\mathcal{B}$ are $G_\otimes^i$, which corresponds to state s_i of D. There is a transition $a \to b : G_\otimes^i \to G_\otimes^j$ exactly when there is a $B \in \mathcal{B}$ with $G_\otimes^i[B] = G_\otimes^j$ and $\pi_{\sim\circ}^*(B) \rightsquigarrow [a \to b]$, but then B must be $B_{i\to j}^{a\to b}$. So such a B exists if and only if we built a path for it in the simulation, which we did for exactly the transitions $a \to b : s_i \to s_j$ in D. Since [ ] is legal in both the open and closed door, [ ] $\in G_\otimes^i$, so every state of $D_\mathcal{B}$ is accepting; this is also true of D by assumption. $\square$

Corollary 5.9. *The mismatched dicrumblers (Figure 2-5) and the self-closing door (Figure 2-4) are each universal for regular prefix-closed gizmos.*

Proof. This follows from Theorem 5.8 using Lemma 5.2, since Ani, Bosboom, et al. [ABD+20] showed that each of these gadgets can simulate the door. □

5.3 Reversible gizmos

Because our definition of reversible gizmos is based on NFAs, the first step of our framework for universality is a tautology. We just need to build, and prove correct, simulations based on reversible NFAs.

Theorem 5.10. *The 2-toggle (Figure 2-6) is universal for (regular) reversible prefix-closed gizmos.*

Whether we need to restrict to regular gizmos depends on whether we use NFAs or ANFAs to define reversible gizmos.

Proof. We call the states of the 2-toggle 'left' and 'right' indicating the direction each tunnel can currently be traversed. We label the locations L, ℓ, R, and r, where the letter indicates the side at the capitalization indicates the tunnel. In particular, in the left state $R \to L$ and $r \to \ell$ are traversable, and in the right state $L \to R$ and $\ell \to r$ are. The 1-toggle is the subgizmo of the 2-toggle on $\{L, R\}$, so we can use 1-toggles in our simulation.

Let G be a prefix-closed gizmo which is recognized by a reversible NFA D with starting state s_0. For a state s of D, let $G^s_\otimes$ be the tensor product of:

- For each state s of D, a 1-toggle labeled s.
- For each pair transition $a \to b : s \to s'$ of D, a 2-toggle labeled $(a \to b, s \to s')$.

Each 1-toggle is the left state except for the one labeled s, and every 2-toggle is the left state.

The 1-toggles will encode the the state of D by having exactly one in the right state. The 2-toggles will prevent leaking when walking through the simulation. A pair of inverse transitions $a \leftrightarrow b : s \leftrightarrow s'$ is implemented together using both corresponding 2-toggles. For the trivial pair $a \leftrightarrow a : s \leftrightarrow s$, there is only one corresponding 2-toggle, so the construction will look different but our description is the same; the corresponding path through $G_\otimes$ will transitively contract to a self-loop.

Now we connect locations as shown in Figure 5-2. More specifically, for each transition $a \to b : s \to s'$, we relate:

- $L_{(a\to b, s\to s')} \sim L_s$
- $R_{(a\to b, s\to s')} \sim R_s$
- $r_{(a\to b, s\to s')} \sim r_{(b\to a, s'\to s)}$.

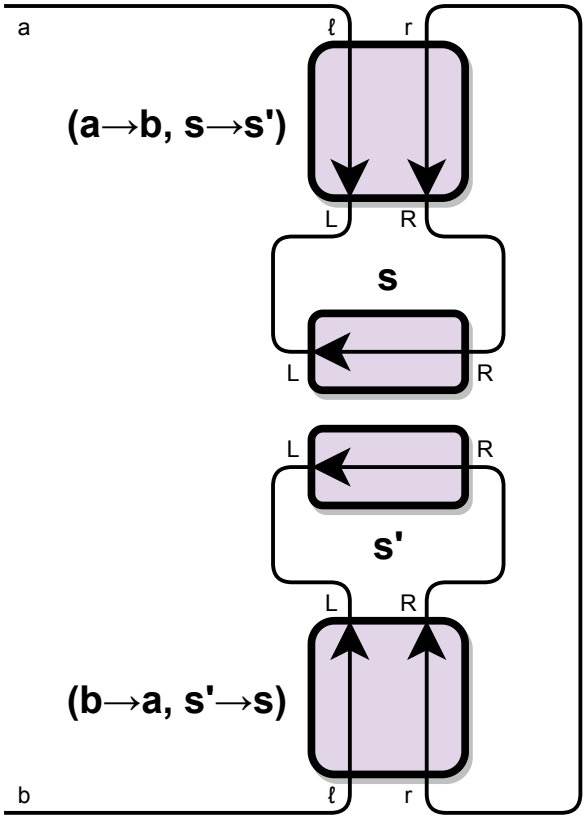

Figure 5-2: Our simulation of an arbitrary regular reversible prefix-closed gizmo using 2-toggles, showing the component involved in the transitions $a \leftrightarrow b : s \leftrightarrow s'$.

Finally, for each location a on G, we merge all locations of the form $\ell_{(a\to\cdot,\cdot\to\cdot)}$ into a single location which corresponds a. Let G^s be the subgizmo on $G^s_\otimes$ on these locations. We will show that $G^{s_0} \cong G$.

The obvious path from a to b in Figure 5-2 corresponds to the transition $a \to b : s \to s'$. Let $B^{a\to b}_{s\to s'}$ be this traversal sequence on $G^{s_0}_\otimes$. To prove correctness, we must show that the set of these paths $\mathcal{B}$ is a basis:

1. Consider a sequence $X \in G^{s_0}$, and use the $\widetilde{X} \in G^{s_0}_\otimes$ provided by Lemma 5.7. Consider a segment of $\widetilde{X}$ which contracts to a single traversal in X. Unless the segment is a single self-loop traversal, it contains no self-loops or contractible locations. The segment must begin and end at location ℓ on some 2-toggle. If it starts at $\ell_{(a\to b,s\to s')}$, the first traversal must be $\ell_{(a\to b,s\to s')} \to L_{(a\to b,s\to s')}$, and continuing the first six traversals must be $B^{a\to b}_{s\to s'}$. By induction, the segment in either a single self-loop or the concatenation of basis elements.

2. We have $\pi^*_{\sim_\circ}(B^{a\to b}_{s\to s'}) \rightsquigarrow [a \to b]$.

3. The locations in the subgizmo are ℓ on some 2-toggle, and non of these is an internal location in $B^{a\to b}_{s\to s'}$.

Now all that remains is showing that $D_\mathcal{B}$ is isomorphic to D. For each transition $a \to b : s \to s'$ in D, we have $G^s_\otimes[B^{a\to b}_{s\to s'}] = G^{s'}_\otimes$, since the path traverses both relevant 1-toggles and resets every 2-toggle it uses. The sequence $B^{a\to b}_{s\to s'}$ is only legal when the s 1-toggle is in the right state, which occurs only in $G^s_\otimes$. So the nonempty reachable internal states (starting from $G^{s_0}_\otimes$) are $G^s_\otimes$, and D and $D_\mathcal{B}$ are isomorphic under the bijection $s \mapsto G^s_\otimes$. □

Corollary 5.11. *The tripwire-lock (Figure 2-7) is universal for (regular) reversible prefix-closed gizmos.*

Proof. As shown by Demaine, Grosof, et al. [DGLR18] (and our example in Figure 2-14), the tripwire-lock simulates the 2-toggle. □

5.4 Strictly bounded gizmos

Next we consider bounded gizmos, which have a reasonably simple characterization in terms of NFAs.

Lemma 5.12. *Every nonempty strictly bounded gizmo with finitely many locations is recognized by an NFA which has no cycles, other than those consisting entirely of transitions corresponding to self-loops of the form $a \to a : s \to s$. In particular, every strictly bounded gizmo with finitely many locations is regular.*

We call such an NFA *acyclic*. Other than the self-loops, the underlying directed graph is a DAG; this is the source of the name for DAG gadgets [DHL20, Lyn20] corresponding to our strictly bounded gizmos. Because gizmos are closed under insertion of self-loops, every NFA recognizing a nontrivial strictly bounded gizmo will have these trivial cycles.

Proof. Suppose G is nonempty, strictly bounded by n and has finitely many locations. We first show that G is regular by constructing a NFA D recognizing it, and then show D is acyclic.

We define D in such a way that it could have infinitely many states, but then show that since G is strictly bounded, D is finite. We define D as in Lemma 3.28, and later in Lemma 6.1: the states of D are the nonempty gizmos $G[X]$, the starting state is G and G' accepts if $[\,] \in G'$ (we need G to be nonempty so that we have a state). There is a transition $t : G' \to X[t]$ for each X and t. Clearly D recognizes G. We say a transition in D of the form $a \to a : s \to s$ is *trivial*.

Suppose for contradiction that D has infinitely many states. Consider the directed graph whose edges are the nontrivial transitions of D. Since G has finite locations, each vertex has boundedly many outgoing edges. Then by König's lemma, since every vertex is reachable from G, there is an infinite path starting at G. Let $X = [t_1, t_2, \dots]$ be the infinite sequence of traversals which make D follow this path, and let $X_i = [t_1, \dots, t_i]$ be the prefix of length i. Since the states of D are nonempty, for each i there is a Y_i with $X_i Y_i \in G$, so X_i contains at most n non-self-loop traversals. Then X also contains at most n non-self-loop traversals.

Let t_m be the last non-self-loop traversal in X, which exists since there are at most n of them. For $i \geq m$, t_{i+1} is a self-loop. Since the graph used to find an infinite path has only nontrivial transitions, t_{i+1} must correspond to a state-changing transition: $G[X_i] \neq G[X_{i+1}]$ for $i \geq m$. Since gizmos are closed under self-loops, moreover $G[X_i] \subsetneq G[X_{i+1}]$. Let $Y_i \in G[X_{i+1}] \setminus G[X_i]$ for each i. Since Y_i has at most n non-self-loop traversals, there must be infinitely many with the same number of non-self-loop traversals. Call their indices $i_1, i_2, \dots$.

By Higman's lemma [Hig52], there are some $j < k$ such that Y_{i_j} is a subsequence of Y_{i_k}. Since Y_{i_j} and Y_{i_k} have the same number of non-self-loop traversals, we can construct Y_{i_k} by inserting self-loops into Y_{i_j}. So $Y_{i_k} \in G[X_{i_j+1}]$. But since $G[X_i] \subset G[X_{i+1}]$ for all $i \geq m$ and $i < i_j + 1 \leq i_k$, also $Y_{i_k} \in G[X_{i_k}]$. But we chose Y_{i_k} to not be in $G[X_{i_k}]$, so we have our contradiction: hence D has finitely many states and is actually a DFA.

We have actually shown that D satisfies the stronger property of having no infinite paths of nontrivial transitions. In particular, it has no cycles of nontrivial transitions. We can remove any trivial transitions from any cycle with at least one nontrivial transition to make a cycle with only nontrivial transitions. Thus D has no cycles other than those consisting entirely of trivial transitions, as desired. □

Clearly acyclic NFAs are also weakly acyclic, so we can explain the names 'strictly bounded' and 'weakly bounded':

Corollary 5.13. *Every strictly bounded gizmo is weakly bounded.*

Theorem 5.14. *The ordered dicrumblers (Figure 2-9) is universal for strictly bounded prefix-closed gizmos with finitely many locations.*

Proof. We label the locations of the ordered dicrumblers so that it is the minimal prefix-closed gizmo containing $[A \to B, C \to D]$.

Let G be a strictly bounded prefix-closed gizmo, and let D be an acyclic NFA provided by Lemma 5.12. We define trivial transitions as in Lemma 5.12. Our simulation will use

- An ordered dicrumblers labeled s for each state s of D
- Two ordered dicrumblers labeled $(a \to b, s \to s')$ and $(a \to b, s')'$ for each nontrivial transition $a \to b : s \to s'$
- An ordered dicrumblers labeled a for each location a of G.

For a set U of labels and a state $s \notin U$, let $G_{\otimes}^{U,s}$ be the tensor product of these ordered dicrumblers all in state 1, except that the one labeled s is in state 2 and each with a label in U is in state 2.

At a high level, the idea is that the gizmos labeled with states encode the state of D: in $G_{\otimes}^{U,s}$, the current state is s, and U contains the previously visited states and the gizmos implementing previously used transitions. At any time, exactly one gizmo will be in state 2 to represent the current state, and gizmos in state 3 have been consumed and will not be needed again. The gizmos labeled with transitions will prevent leaking when implementing transitions. A transition $a \to b : s \to s'$ should change gizmo s from state 2 to 3, and change gizmo s' from state 1 to 2. The gizmos labeled with locations are simply to ensure every location in G has at least one preimage under $\pi_{\sim}$, and thus exists in the simulated gizmo—this is only necessary when a location is used by no transition in D.[23]

One transition's worth of our simulation is shown in Figure 5-3. Formally, for each nontrivial transition $a \to b : s'$, we identify:

- $B_{(a\to b, s\to s')} \sim C_s$
- $D_s \sim C_{(a\to b, s\to s')}$
- $D_{(a\to b, s\to s')} \sim A_{(a\to b, s\to s')'}$
- $B_{(a\to b, s\to s')'} \sim A_{s'}$
- $B_{s'} \sim D_{(a\to b, s\to s')'}$.

Finally, for each $a \in \operatorname{locs} G$, we identify all locations of the form $A_{(a\to\cdot, \cdot\to\cdot)}$ or $D_{(\cdot\to a, \cdot\to\cdot)'}$ into a single location which we identify with a. Let $G^{U,s}$ be the subgizmo of $G_{\otimes}^{U,s}/\!\sim$ on these locations. We will show that for appropriate U, the language recognized by D starting from s is $G^{U,s}$, and in particular the language recognized from the starting state s_0 is $G^{\emptyset,s_0} \cong G$.

Let $B_{s\to s'}^{a\to b}$ be the obvious path from a to b through Figure 5-3. For trivial transitions, let $B_{s\to s}^{a\to a} = [D_a \to D_a]$. Suppose none of s, s', $(a \to b, s \to s')$, and $(a \to b, s \to s')'$ is in U, so gizmo s is in state 2 and the others are in state 1 in

[23] We do not need this for other universality proofs since those NFAs include self-loop traversals, which are guaranteed to exist.

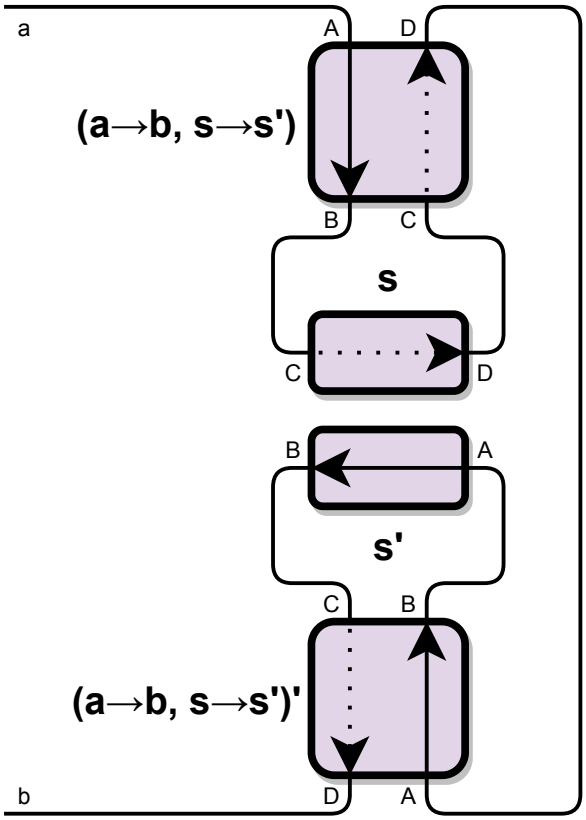

Figure 5-3: Our simulation of an arbitrary strictly bounded prefix-closed gizmo using ordered dicrumblers, showing the component involved in the nontrivial transition $a \to b : s \to s'$.

$G^{U,s}$. Then by design, $G_\otimes^{U,s}[B_{s\to s'}^{a\to b}] = G_\otimes^{U\cup\{s,(a\to b,s\to s'),(a\to b,s\to s')'\},s'}$. If any of those labels is in U, then $G_\otimes^{U,s}[B_{s\to s'}^{a\to b}] = \emptyset$. Similarly to our previous universality proofs, the set $\mathcal{B}$ of $B_{s\to s'}^{a\to b}$ is a basis. By induction, the reachable internal states (from $G_\otimes^{\emptyset,s_0}$) are all $G_\otimes^{U,s}$ with $s \notin U$ (but not all states of this form are necessarily reachable).

Now it suffices to show that $D_\mathcal{B}$ recognizes the same language as D. These NFAs are not isomorphic, since states $D_\mathcal{B}$ encode the entire history in D, but this history will not affect what is allowed for the future. Both NFAs have a transition $a \to a : s \to s$ for all a and s, and no other transition $a \to b : s \to s$. So we can consider only paths that do not use these trivial transitions, and both NFAs will allow freely inserting self-loops. The nontrivial transitions in $D_\mathcal{B}$ are $a \to b : G_\otimes^{U,s} \to G_\otimes^{U\cup\{s\},s'}$ for each nontrivial transition $a \to b : s \to s'$ of D and U not containing s, s', $(a \to b, s \to s')$, or $(a \to b, s \to s')'$.

Suppose D has a path $[a_1 \to b_1, \ldots, a_k \to b_k] : s_0 \to \cdots \to s_k$ of nontrivial transitions. Let the set of gizmos consumed before traversal i be

$$U_i = \{s_j, (a_{j+1} \to b_{j+1}, s_j \to s_{j+1}), (a_{j+1} \to b_{j+1}, s_j \to s_{j+1})' \mid j < i\}.$$

Since D is acyclic, the path does not repeat any states or transitions, so none of s_i, $(a_{i+1} \to b_{j+1}, s_i \to s_{i+1})$, and $(a_{i+1} \to b_{j+1}, s_i \to s_{i+1})'$ is in U_j for $j \geq i$. So $D_\mathcal{B}$ has the path $[t_1, \ldots, t_k] : G_\otimes^{U_0,s_0} \to \cdots \to G_\otimes^{U_k,s_k}$. Thus $\mathcal{L}(D) \subset \mathcal{L}(D_\mathcal{B})$. Conversely, suppose $D_\mathcal{B}$ has a path $[t_1, \ldots, t_k] : G_\otimes^{U_0,s_0} \to \cdots \to G_\otimes^{U_k,s_k}$ of nontrivial transitions with $U_0 = \emptyset$. Then we can ignore the U_i, and D has the corresponding path $[t_1, \ldots, t_k] : s_0 \ldots, s_k$, so $\mathcal{L}(D_\mathcal{B}) \subset \mathcal{L}(D)$. □

5.5 Weakly bounded gizmos

Our definition of weakly bounded provides a characterization of weakly bounded gizmos in terms of NFAs. We need to construct simulations based on weakly acyclic NFAs.

Theorem 5.15. *The brittle door (Figure 2-10) is universal for weakly bounded prefix-closed gizmos.*

Proof. We label the locations of the brittle door the same way we label those of the door. The simulation will be similar to the one in Theorem 5.14, except that we need to account for state-preserving transitions.

Let G be a weakly bounded gizmo, recognized by a weakly acyclic NFA D. Our simulation will use

- A brittle door s for each state s of d
- A brittle door $(s, a \to b)$ for each state-preserving transition $a \to b : s \to s$
- Two brittle doors $(a \to b, s \to s')$ and $(a \to b, s \to s')'$ for each state-changing traversal $a \to b : s \to s'$ with $s \neq s'$

□

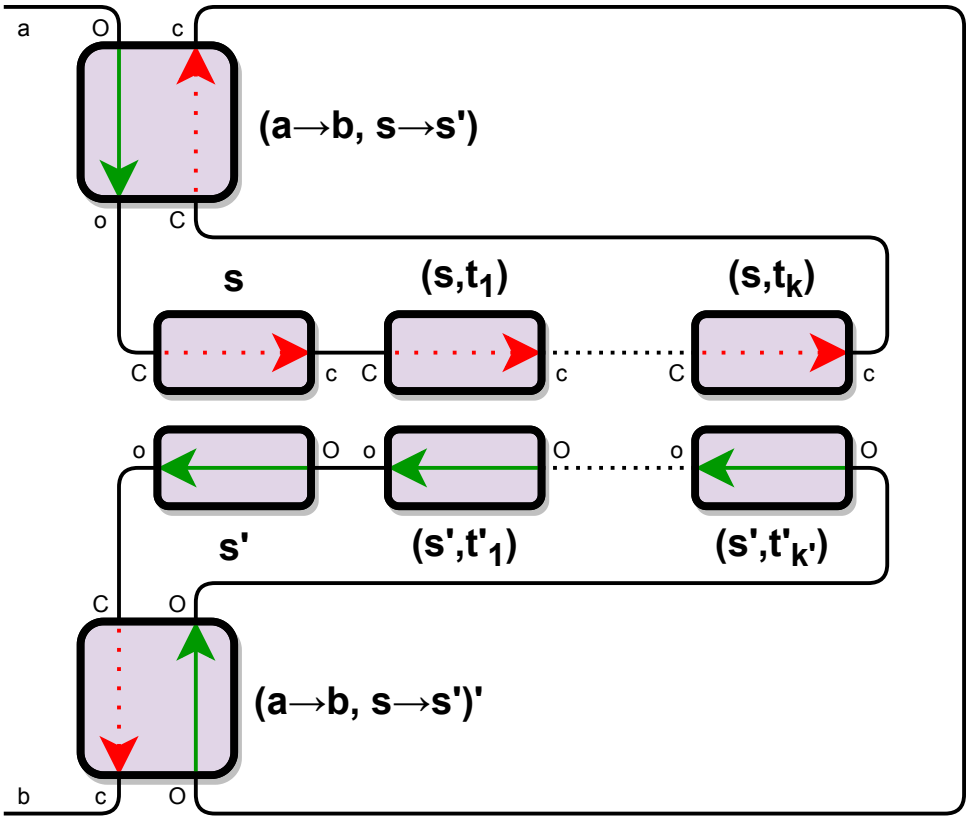

Figure 5-4: The component implementing the state-changing transition $a \to b : s \to s'$ in our simulation of an arbitrary weakly bounded prefix-closed gizmo using brittle doors.

We will use gizmos s and $(s, a \to b)$ to encode the state, $(a \to b, s \to s')$ and $(a \to b, s \to s')'$ to support state-changing traversals, and $(s, a \to b)$ also to support state-preserving traversals. For a set of labels U and a state $s \notin U$ with each $(s, a \to b) \notin U$, let $G_{\otimes}^{U,s}$ be the tensor product of these brittle doors all in state 1, except that s and $(s, a \to b)$ are in state 2, and each gizmo with label in U is in state 3.

Now we identify locations as shown in Figure 5-4. Formally, for each state s, let the state-preserving transitions $a \to b : s \to s$ be t_1 through t_k, and identify

- $c_s \sim C_{(s,t_1)}$
- For each $i < k$, $c_{(s,t_i)} \sim C_{(s,t_{i+1})}$
- For each $i < k$, $o_{(s,t_{i+1})} \sim O_{(s,i)}$
- $o_{(s,t_1)} \sim O_s$.

Then, for each state-changing transition $a \to b : s \to s'$, suppose the state preserving transitions on s and s' are t_1 through t_k and t'_1 through $t'_{k'}$, respectively. Identify:

- $o_{(a \to b, s \to s')} \sim C_s$

- $c_{(s,t_k)} \sim C_{(a\to b, s\to s')}$
- $c_{(a\to b,s\to s')} \sim O_{(a\to b,s\to s')'}$
- $o_{(a\to b,s\to s')'} \sim O_{(s,t'_{k'})}$
- $o_{s'} \sim C_(a \to b, s \to s')$.

If $k = 0$, use c_s instead of $c_{(s,t_k)}$, and similarly for k'.

Finally, merge and identify with $a \in \operatorname{locs} G$ all locations of the forms

- $O_{(a\to\cdot,\cdot\to\cdot)}$
- $c_{(\cdot\to a,\cdot\to\cdot)}$
- $T_{(\cdot,a\to}.c)$
- $t_{(\cdot,\cdot\to a}c)$.

Let $G^{U,s}$ be the subgizmo of $G^{U,s}_\otimes$ on these locations. If s_0 is the starting state of D, we will show that D recognizes $G^{\emptyset,s_0}$, so $G^{\emptyset,s_0} \cong G$.

The idea is that we implement state-changing transitions the same way as in Theorem 5.14, except that we open and close additional doors when setting the state. A state-preserving transition $a \to b : s \to s$ is implemented by the traverse tunnel on gizmo $(s, a \to b)$. The state $G^{U,s}_\otimes$ represents state s of D for appropriate U: the gizmos s and $(s, a \to b)$ which represent state s are all open, and only gizmos for states and transitions which have been used previously and thus will not be used again are in U. Gizmo s is included to ensure there is at least one gizmo representing state s, to ensure the path through Figure 5-4 is only traversable in states representing s, though this is only needed in trivial cases.

To prove correctness, let $B^{a\to b}_{s\to s'}$ be the obvious path from a to b through Figure 5-4 for each state-changing transition $a \to b : s \to s'$ (note that its length depends on k and k'), and let $B^{a\to b}_{s\to s} = [T_{(s,a\to b)} \to t_{(s,a\to b)}]$ for each state-preserving transition $a \to b : s \to s$. By design, for a state-changing transition $a \to b : s \to s'$, if none of s, (s, t_i), s', (s', t_j), $(a \to b, s \to s')$, and $(a \to b, s \to s')'$ is in U, then

$$G^{U,s}_\otimes[B^{a\to b}_{s\to s'}] = G^{U\cup\{s,(s,t_i),(a\to b,s\to s'),(a\to b,s\to s')'\},s'}_\otimes.$$

For a state-preserving transition $a \to b : s \to s$, if $(s, a \to b)$ is not in U, then $G^{U,s}_\otimes[B^{a\to b}_{s\to s}] = G^{U,s}_\otimes$.

As in Theorem 5.14, these form a basis $\mathcal{B}$. The reachable states all have the form $G^{U,s}_\otimes$, with start state $G^{\emptyset,s_0}_\otimes$. The transitions of $D_\mathcal{B}$ are described by the equations above.

It suffices to show that D and $D_\mathcal{B}$ recognize the same language. This follows much like in Theorem 5.14:

Suppose D has a path $[t_1, \ldots, t_k] : s_0 \to \cdots \to s_k$. We describe a path in $D_\mathcal{B}$ by defining U_i recursively, beginning with $U_0 = \emptyset$. If t_i is state-preserving, let $U_i = U_{i-1}$.

If t_i is state-changing, let $U_i = U_{i-1} \cup \{s_{i-1}, (s_{i-1}, t_j), (t_i, s_i \to s_{i+1}), (t_i, s_i \to s_{i+1})'\}$; we are simply adding the brittle doors consumed by the ith transition. Since D has no cycles containing multiple states, there is never a state-changing transition from s and then later a transition to s, and no state-changing transition is used multiple times. Thus U_{i-1} contains no label of a gizmo needed for $B^{t_i}_{s_{i-1} \to s_i}$, so $D_{\mathcal{B}}$ has the path $[t_1, \ldots, t_k] : G_{\otimes}^{U_0, s_0} \to \cdots \to G_{\otimes}^{U_k, s_k}$.

Conversely, if $D_{\mathcal{B}}$ has the path $[t_1, \ldots, t_k] : G_{\otimes}^{U_0, s_0} \to \cdots \to G_{\otimes}^{U_k, s_k}$ with $U_0 = \emptyset$, then D has the path $[t_1, \ldots, T_k : s_0 \to \cdots \to s_k]$.

5.6 $[X \implies XX]\,[XY \implies YX]$ gizmos

This class of gizmos, and the classes discussed in the remainder of this chapter, are simple enough to characterize that we will not need the machinery of Section 5.1. Instead, our last few universality results follow this simpler template:

- Show that a gizmo in the class is determined by some relatively simple property.
- Simulate a gizmo which shares that property with a given gizmo in the class.
- By the characterization, the given gizmo is isomorphic to the simulated gizmo.

What it takes to determine a gizmo will of course depend on the class under consideration. For $[X \implies XX]\,[XY \implies YX]$ gizmos, we have the following characterization.

Lemma 5.16. *Let G be a $[X \implies XX]\,[XY \implies YX]$ prefix-closed gizmo. Suppose X and Y are traversal sequences on $\operatorname{locs} G$, and the set of traversals in X is the same as the set of traversals in Y. Then $X \in G$ if and only if $Y \in G$.*

Proof. Since they have the same sets of traversals, it is possible to transform X into Y (and vice-versa) through reordering traversals, copying traversals, and removing duplicate traversals. Thus it suffices to show that performing a single one of these operations preserves membership in G. Moreover, we can assume each reordering operation simply swaps consecutive traversals, and (taking advantage of reordering) copied traversals are placed at and duplicates are removed from the end of the sequence. So it suffices to show that G satisfies

$$\begin{aligned} ABCD \in G &\implies ACBD \in G \\ ABC \in G &\implies ABCB \in G \\ ABCB \in G &\implies ABC \in G \end{aligned}$$

for any traversal sequences A, B, C, and D, corresponding to reordering, copying, and removing duplicates, respectively. The first two happen to be implication properties.

First, we prove the useful intermediate that $ABC \in G \implies AC \in G$:

$$ABC \in G \implies CAB \in G \implies CA \in G \implies AC \in G.$$

Hereafter we will omit '$\in G$' in implication chains like this. The third property above is trivial since G is prefix-closed. We now prove the other two:

$$ABCD \implies ABCDABCD \implies ACDABCD \implies ACBCD \implies ACBD$$
$$ABC \implies ABCABC \implies ABCBC \implies ABCB.$$

□

We say a set S of traversals is *good* for a $[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmo if a traversal sequence containing exactly the traversals in S is allowed. Lemma 5.16 says goodness does not depend on the choice of traversal sequence, and in particular the gizmo is determined by its good sets. It follows that every such gizmo with finitely many locations is regular.

Corollary 5.17. *$[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmos are closed under traversals.*

Proof. Let G be such a gizmo and X be a traversal sequence. Let S be the set of traversals in S. Then $Y \in G[X]$ exactly when the set of traversals in XY is good for G. This set is $S \cup T$, where T is the set of traversals in Y. Thus whether $Y \in G[X]$ is determined by T. Then the converse of Lemma 5.16, which is immediate since union is idempotent and commutative, implies that $G[X]$ satisfies $X \implies XX$ and $XY \implies YX$. □

On the other hand, neither $X \implies XX$ nor $XY \implies YX$ alone defines a class closed under traversals, even among prefix-closed gizmos.

Self-loops in good sets will be inconvenient, so we intend to ignore them. Specifically, whether a set of traversals is good does not depend on which self-loops it has: by prefix-closure, subsets of good sets are always good, so removing self-loops preserves goodness. Conversely, since gizmos are closed under insertion of self-loops, adding self-loops preserves goodness. Thus we have the following refinement, which is important enough to be called a corollary:

Corollary 5.18. *A $[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmo is determined by its good sets of non-self-loop traversals.*

Now we can finally prove the universality result of this section.

Theorem 5.19. *The initial state of the mutually closing diodes (Figure 2-11) is universal for $[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmos with finitely many locations.*

Proof. Let G be a nonempty such gizmo with n locations. Suppose G has k good sets, labeled S_i for $1 \le i \le k$. Our simulation of G will use $\left(\sum_{i=1}^{k} |S_i|\right)^2 - \sum_{i=1}^{k} \left(|S_i|^2\right)$ copies of the mutually closing diodes and $\sim_{i=1}^{k} |S_i|$ copies of the diode—note that we can easily simulate a diode by taking the subgizmo on a single tunnel. This is one copy of the mutually closing diodes for each pair of traversals in unequal good sets.

For each $i \neq i'$ and traversals $t \in S_i$ and $t' \in i'$, we include a copy of the mutually closing diodes labeled (i, t, i', t'). For each i and $t \in S_i$, we include a copy of the diode labeled (i, t). Let $G_\otimes$ be the tensor product of these gizmos. We partition the tunnels of all of these gizmos into paths which are connected in series: for each i and $t \in S_i$, connect the diode (i, t), the first tunnel of each gizmo labeled $(i, t, \cdot, \cdot)$, and the second tunnel of each gizmo labeled $(\cdot, \cdot, i, t)$. The order of the tunnels in the path does not matter. The idea is for each path to close all paths from other good sets, so that the agent can only use paths from a single good set.

Finally, for each location $a \in \operatorname{locs} G$, merge the start of each path corresponding to a traversal $a \to \cdot$ in a good set and the end of each path corresponding to $\cdot \to a$. The resulting location corresponds to a; let G' be the subgizmo on these locations.

To show that $G = G'$, we argue that they have the same good sets. Suppose a set S is good for G. Then if each traversal t_i is in S, the concatenation of paths through $G_\otimes$ corresponding to $t_i \in S$ is in $G_\otimes$, since none of these paths use different tunnels on the same mutually closing diodes. Thus $[t_0, \ldots, t_m] \in G'$, so S is good for G'.

Conversely, suppose S is good for G' and does not contain self-loops. Let X contain exactly the traversals in S, so $X \in G'$. Then there is some $\widetilde{X} \in G_\otimes$ with $\pi^*_{\widetilde{\sim}\circ}(\widetilde{X}) \rightsquigarrow X$. Any self-loop in $\widetilde{X}$ must be contracted away, so we can assume $\widetilde{X}$ contains no self-loops, and thus is a concatenation of the paths we built in the simulation (that is, these paths are a basis). By construction, it is never legal in $G_\otimes$ to traverse two such paths from different good sets for G, so the paths comprising $\widetilde{X}$ come from traversals in the same good set. Let Y be the sequence of these traversals; then we have $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow Y \rightsquigarrow X$. But since each traversal in Y is in the same good set, $Y \in G$, so $X \in G$. Hence S is also good for G.

Our simulation is somewhat wasteful: to make it simpler to describe, we use twice as many copies of the mutually closing diodes as needed, and the diodes serve only to prevent paths from being traversal backwards, which is only relevant when G has only one good set. □

5.7 Unchanging gizmos

We are now reaching some rather weak classes of gizmos.

Definition 5.20. A prefix-closed gizmo is *unchanging* if it satisfies both implication properties $XY \implies Y$ and $X, Y \implies XY$.

Together the two implication properties say that $G = G[X]$ whenever $X \in G$; since G is also prefix-closed, $G[X]$ is always either G or empty. Unchanging prefix-closed gizmos correspond to the unchanging gadgets from Lynch [Lyn20]. Since each implication property defines a closed class, and unchanging gizmos are the intersection of two of them, unchanging gizmos are closed under arbitrary simulation. The combination of the three implication properties defining unchanging prefix-closed gizmos has a simple description: XY is allowed if and only if both X and Y individually are.

We first show that a nonempty unchanging prefix-closed gizmo is determined by the single traversals it allows, and then use this to prove universality.

Lemma 5.21. *Unchanging prefix-closed gizmos are closed under traversals.*

Proof. Suppose G is unchanging and prefix-closed, and consider a traversal sequence Z. By Lemma 4.2, $G[Z]$ is prefix-closed. We must show it also satisfies $XY \implies Y$ and $X, Y \implies XY$. For the former, if $ZXY \in G$, then Z, X, and Y are each in G, so $ZY \in G$. For the latter, if ZX and ZY are in G, then so are Z, X, and Y, so $ZXY \in G$. □

Lemma 5.22. *If G is a nonempty unchanging prefix-closed gizmo, then a traversal sequence $X = [t_1, \ldots, t_k]$ is in G if and only if $[t_i] \in G$ for each i.*

Proof. For the forwards direction, since G is prefix-closed $[t_1, \ldots, t_i] \in G$ and then since G satisfies $XY \implies Y$, $[t_i] \in G$.

For the reverse direction, we induct on k, and essentially use the nice structure mentioned above. When $k = 0$ the claim is that $[\,] \in G$, which follows from G being nonempty and prefix-closed. Assume $k > 0$ and $[t_i] \in G$ for each $1 \le i \le k$. Consider the gizmo $G[[t_1]]$: it is nonempty since $[t_1] \in G$ and unchanging and prefix-closed by Lemma 5.21. Since G satisfies $X, Y \implies XY$, for each i we have $[t_i] \in G[[t_1]]$. So by inductive hypothesis on $G[[t_1]]$, we have $[t_2, \ldots, t_k] \in G[[t_1]]$, and thus $X \in G$ as desired. □

Theorem 5.23. *The diode (Figure 2-12) is universal for unchanging prefix-closed gizmos with finitely many locations.*

This universality result implies all unchanging prefix-closed gizmos with finitely many locations are regular; it is not hard to prove this directly. Gizmos with finitely many locations are trivially closed under finite but not arbitrary simulation.

Proof. We call the input and output of the diode I and O, so the diode is the minimal prefix-closed gizmo containing $(I \to O)^*$.

Let G be a nonempty unchanging prefix-closed gizmo with finitely many locations. Our simulation of G will have a diode labeled with each traversal t for which $[t] \in G$. For each $a \in \operatorname{locs} G$, we identify all locations of the form $I_{a\to\cdot}$ or $O_{\cdot\to a}$ into a single location, which we identify with a. Let $G_\otimes$ be the tensor product of the diodes and G' be the resulting simulated gizmo. We wish to show that $G = G'$.

Lemma 5.22 tells us that a nonempty prefix-closed gizmo is determined by its traversal sequences of length one. So it suffices to prove that for any single traversal t, $[t] \in G$ if and only if $[t] \in G'$.

If $[t] \in G$, notice that $[I_t \to O_t] \in G_\otimes$ and $\pi_{\sim\circ}(I_t \to O_t) = t$. Thus $[t] \in G'$. This is easy because we specifically included a diode to support t.

Conversely, suppose $[t] \in G'$. Then there is some $\widetilde{X} \in G_\otimes$ with $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow [t]$. Each traversal in $\widetilde{X}$ is either a self-loop or crosses a diode, and thus each traversal in $\pi^*_{\sim\circ}(\widetilde{X})$ is in G. By Lemma 5.22, $\pi^*_{\sim\circ}(\widetilde{X}) \in G$. Since G is a gizmo and $\pi^*_{\sim\circ}(\widetilde{X}) \rightsquigarrow [t]$, also $[t] \in G$. □

5.8 $[X \implies X^{-1}]$ unchanging gizmos

This class is even weaker than unchanging gizmos, and is the subject of our final universality result.

Theorem 5.24. *The wire (Figure 2-13) is universal for $[X \implies X^{-1}]$ unchanging prefix-closed gizmos with finitely many locations.*

Proof. The gizmos in question satisfy four implication properties:

$$X, Y \implies XY; \quad XY \implies X; \quad XY \implies Y; \quad X \implies X^{-1}.$$

Note in particular that if $[a \to b]$ is allowed by such a gizmo, then so is $[b \to a]$.

Our proof is very similar to the proof of Theorem 5.23, but using wires instead of diodes. We call the locations of the wire I and O, so the wire in the minimal prefix-closed gizmo containing $(I \to O | O \to I)^*$.

Let G be a nonempty $[X \implies X^{-1}]$ unchanging prefix-closed gizmo. Our simulation has a wire labeled $a \leftrightarrow b$ for each pair of opposite traversals $a \to b$ and $b \to a$ with $[a \to b]$ and $[b \to a]$ both in G; this is equivalent to either one being in G. For each $a \in \operatorname{locs} G$, we identify all locations of the form $I_{a\leftrightarrow}$ or $O_{\cdot\leftrightarrow a}$ into a single location, which we identify with a. Let $G_\otimes$ be the tensor product of the wires and G' be the resulting simulated gizmo.

To show that $G = G'$, by Lemma 5.22 it suffices to show that they agree on length-one traversal sequences. If $[a \to b] \in G$, then there is a wire labeled either $a \leftrightarrow b$ or $b \leftrightarrow a$. Then $[I_{a\leftrightarrow b} \to O_{a\leftrightarrow b}]$ or $[O_{b\leftrightarrow a} \to I_{b\leftrightarrow a}]$, respectively, is a witness for $[a \to b] \in G'$.

Conversely, suppose $[t] \in G'$. Then there is some $\widetilde{X} \in G_\otimes$ with $\pi_{\sim_\circ}^*(\widetilde{X}) \rightsquigarrow [a \to b]$. Each traversal in $\widetilde{X}$ is either a self-loop or is one direction of $I_{a'\leftrightarrow b'} \leftrightarrow O_{a'\leftrightarrow b'}$ across some wire. So each traversal in $\pi_{\sim_\circ}^*(\widetilde{X})$ is either a self-loop or one direction of $a' \leftrightarrow b'$, and in either case is in G. By Lemma 5.22, $\pi_{\sim_\circ}^*(\widetilde{X}) \in G$, so $[t] \in G$. □

Chapter 6

Arbitrary simulation

In this chapter, we consider arbitrary simulations, and adapt some of our results for finite simulations to this case. When proving closure under simulation in Chapter 4, we noted which classes are even closed under arbitrary simulation. In particular, classes defined by implication properties are closed under simulation (Theorem 4.6), and all other classes of gizmos we considered are not, with the possible exception of reversible gizmos we discuss below.

Universality results are more complicated to adapt to arbitrary simulation. If we know, for some small set of gizmos S and some class $\mathcal{C}$ closed under arbitrary simulation, that S is universal for the regular gizmos in $\mathcal{C}$, then we would like to show that S is arbitrarily universal for $\mathcal{C}$. This is not always true—for example, if every gizmo in $\mathcal{C}$ is regular—but it is suggestive. In particular: for each of our universality results for a class closed under arbitrary simulation, is it still true if we remove "regular" and and replace "universal" with "arbitrarily universal"?

Several of our universality results are based on NFAs for gizmos. For nonregular gizmos, we need to consider automata like NFAs except that they are allowed to have an infinite alphabet and infinitely many states. Despite the oxymoron, we call these *arbitrary NFAs*, or ANFAs. When they are deterministic (meaning they have exactly one transition for each state and symbol), we call them *arbitrary DFAs*, or ADFAs. ADFAs are sufficient to capture any gizmo, or indeed any language:

Lemma 6.1. *Any language L on Σ is recognized by an ADFA.*

Proof. For a sequence $X \in \Sigma^*$, let $L[X] = \{Y \mid XY \in L\}$ (as with gizmos). Construct the ADFA which has states $\{L[X] \mid X \in \Sigma^*\}$ and a transition $a : s \to s[a]$. This ADFA recognizes L as in the proof of Lemma 3.28. □

Then, following exactly Lemma 3.29, we have

Lemma 6.2. *If G is a nonempty prefix-closed gizmo, G is recognized by an ANFA where every state is accepting.*

The results of Section 5.1 do not assume that the gizmos involved are regular, provided we acknowledge that $D_{\mathcal{B}}$ is actually an ANFA. So we can use the same framework for proving universality results, but with "NFA" replaced by "ANFA."

Note that we do, however, rely on the assumption that traversal sequences are finite (such as when inducting on them), but this is not affected by arbitrary simulation.

Now we can check that that specific universality results still hold for arbitrary simulation. In particular, by simply inserting (the appropriate form of) 'arbitrary' in the appropriate locations in the proofs of their counterparts[24] from Chapter 5, we have the following results:

Theorem 6.3 (cf. Theorem 5.8)**.** *The door gadget (Figure 2-3) is arbitrarily universal for prefix-closed gizmos.*

Corollary 6.4 (cf. Corollary 5.9)**.** *The mismatched dicrumblers (Figure 2-5) and the self-closing door (Figure 2-4) are each arbitrarily universal for prefix-closed gizmos.*

Theorem 6.5 (cf. Theorem 5.23)**.** *The diode (Figure 2-12) is arbitrarily universal for unchanging prefix-closed gizmos.*

Theorem 6.6 (cf. Theorem 5.24)**.** *The wire (Figure 2-13) is arbitrarily universal for reversible unchanging prefix-closed gizmos.*

For these last two, all that we gain by allowing arbitrary simulation is gizmos with infinitely many locations.

Not all of our universality results generalize like this, even for classes closed under arbitrary simulation. We examine those that do not more carefully.

6.1 Reversible gizmos

By a quirk of Definition 4.8, all reversible gizmos as defined are regular. But reversibility is at heart a property that can hold for nonregular gizmos, so we adapt our definition to use ANFAs:

Definition 6.7. An ANFA with alphabet $\mathcal{T}(L)$ is *reversible* if for every transition $a \to b : s \to s'$, there is a reverse transition $b \to a : s' \to s$.A gizmo is *reversible* if it is recognized by a reversible ANFA.

With this new definition, closure under arbitrary simulation follows just as before.

Theorem 6.8 (cf. Theorem 4.10)**.** *Reversible gizmos are closed under arbitrary simulation.*

We can extend Conjecture 4.9 to include nonregular gizmos. Again this would be interesting even if only for prefix-closed gizmos.

Conjecture 6.9. *Every gizmo satisfying both* $X, Y \implies XX^{-1}Y$ *and* $XYZ \implies XYY^{-1}YZ$ *is recognized by a reversible ANFA.*

Now our universality results for reversible gizmos adapts with no issues.

[24] Also replace references to other results with their arbitrary counterparts, and in some cases remove "with finitely many locations."

Theorem 6.10 (cf. Theorem 5.10)**.** *The 2-toggle (Figure 2-6) is arbitrarily universal for reversible prefix-closed gizmos.*

Corollary 6.11 (cf. Corollary 5.11)**.** *The tripwire-lock (Figure 2-7) is arbitrarily universal for reversible prefix-closed gizmos.*

6.2 Bounded gizmos

Strictly bounded and weakly bounded gizmos have largely the same considerations here. Both classes are not closed under arbitrary simulation, so we should not expect a universality result for bounded gizmos. But we can say something else interesting: every gizmo is 'strictly bounded by ∞.'

Lemma 6.12. *Any language L on Σ is recognized by an acyclic ADFA.*

Proof. This requires a different, even more naive, approach to proving Lemma 6.1. We construct an ADFA D whose states are strings in Σ^*. A state X accepts if $X \in L$, and the start state is the empty string. We have a transition $t : X \to Xt$ for each $t \in \Sigma$.

Clearly after being fed X, D is in state X, so it recognizes L. It is acyclic because paths only make the state a longer string. □

Now adapting our universality result for strictly bounded gizmos to arbitrary simulation yields the following theorem.

Theorem 6.13 (cf. Theorem thm:ordered dicrumblers universal strictly bounded)**.** *The ordered dicrumblers (Figure 2-9) is arbitrarily universal for prefix-closed gizmos.*

We can simulate all gizmos, not just the strictly bounded ones. It follows that the brittle door is also arbitrarily universal for prefix-closed gizmos. An alternative way to prove Theorem 6.13 is to use the fact that we have already arbitrarily simulated mismatched dicrumblers using ordered dicrumblers (Lemma 4.17), and then use Corollary 6.4.

6.3 $[X \implies XX]\ [XY \implies YX]$ gizmos

Adapting our universality result for $[X \implies XX]\ [XY \implies YX]$ gizmos fails for a more interesting reason. Our simulation in Theorem 5.19 involves a path through a series of tunnels whose length depends

For a cardinal κ, let the *κ-closing diodes* be the gizmo with κ tunnels $A_i \to B_i$, which is the minimal prefix-closed gizmo containing $(A_i \to B_i)^*$ for each i. Think of it as κ diodes, any one of which closes all the others. In particular, the mutually closing diodes (Figure 2-11) is the 2-closing diodes, and Theorem 5.19 implies that the mutually closing diodes simulatios the κ-closing diodes for any finite κ.

For countable $\kappa = \aleph_0$, it is possible with some cleverness to simulate the $\aleph_0$-closing diodes. To avoid infinite-length paths, one can instead have finite paths of arbitrary

length; the path corresponding to $A_i \to B_i$ has length $O(i)$[25]. It likely follows that the mutually closing diodes can simulate any $[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmo with countably many locations. For uncountable κ, this does not seem to be possible. We seem be able to achieve only locally finite 'fanout,' and thus only countably many locations can be a finite distance away. It would be interesting to prove this more carefully.

The universality proofs of Ani, Bosboom, et al. [ABD$^+$20] and Ani, Demaine, et al. [ADHL20] (the latter requires a slightly different notion of simulation) would also fail to adapt to arbitrary simulation and nonregular gizmos for the same reason—they set the state using paths that pass through a tunnel for each state of the simulated gadget, which would become infinitely long. The first, that doors are universal, can be fixed, as we have done in Theorem 6.3. The latter, that the set-up/set-down/switch is universal for input-output gadgets (using a notion of simulation appropriate to these gadgets), seems to face more fundamental issues similar to those just discussed.

[25] In particular, length $2i$ is attainable for $i = 1, 2, \ldots$.

Chapter 7

Future directions

We have defined a formal notion corresponding to the simulation of motion-planning gadgets studied in previous work [AAD+20, ABD+20, ADHL20, DGLR18, DHL20, Lyn20], and proved a handful of both positive and negative results about the existence of simulations. Each of our results is a partial answer to Question 3.2, which is itself a special case of Question 3.1. In particular, each of our universality results exactly characterizes the gizmos which a specific small set of gizmos can simulate. A far-reaching goal would be to obtain a complete answer to Question 3.1, but there is a lot more work left to do if we are to fully understand gadget simulation, both within our formalism and beyond it.

7.1 Questions from this work

First, we collect specific questions brought up elsewhere in this thesis.

- Conjecture 4.9: is every regular gizmo satisfying $X, Y \implies XX^{-1}Y$ and $XYZ \implies XYY^{-1}YZ$ recognized by a reversible NFA? We can extend to nonregular gizmos and NFAs (Conjecture 6.9) or restrict to prefix-closed gizmos.

- Can we define weakly bounded gizmos (Definition 4.14) without resorting to NFAs?

- Can the mutually closing diodes arbitrarily simulate every $[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmo with countably many locations (Section 6.3)? Can it arbitrarily simulate the κ-closing diodes for uncountable κ, or any meaningfully uncountable $[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmo?

7.2 Within the system

Next we pose several problems about gizmo simulation as we have defined it, which would contribute to answering Question 3.1.

- Can we find universal sets of gizmos for other closed classes we have defined? Our method for proving universality seems to work fairly generally, so likely without too much trouble we can adapt to additional classes. One challenge is showing equivalences between natural definitions which do not involve NFAs and characterizations in terms of NFAs with nice properties. In particular, a reasonable candidate for universality for bounded gizmos would be something like the minimal prefix-closed gizmo containing $(A \to B, C \to D, B \to A, D \to C)^*$, and the locking 2-toggle is a natural candidate for bounded reversible gizmos.

- It would be particularly exciting to prove infinitely many universality results to match our infinite family of closed classes. Can we find a universal set for any implication property? Perhaps it helps to impose a further constraint on the implication property, such as that it has only one antecedent.

- What other interesting classes of gizmos are closed under simulation? Can they be described in terms of implication properties, or another infinite family of closed properties? Can we find universal sets of gizmos for them?

- One approach to characterizing simulability is to define, for each gizmo G, the class $\mathcal{C}_G$ of gizmos which G simulates. This is trivially closed under simulation, and G is universal. If we consider a family of gizmos, such as those which have 2 locations and (are recognized by an NFA with) at most 2 states, it may be tractable to determine which of these gizmos simulate each other, defining a preorder on them and determining which $\mathcal{C}_G$ are equal. This may in turn lead to insights toward a more direct characterization of $\mathcal{C}_G$ for specific gizmos.

- What about universality for classes of gizmos that are not prefix-closed? For instance, can we adapt Theorem 5.8 to show that the set containing all gizmos corresponding to the door gadget, in different states and with different target sets, is universal for regular gizmos?

- What can we say about relationships between gizmo classes? In complexity theory, people care about results like BPP $\subset \Sigma_2^{\mathrm{P}}$, but in this thesis we have largely ignored inclusions between gizmo classes like this (with some exceptions, such as Corollary 5.13). Just as in complexity theory and completeness, one way to prove such results is using universality; for instance, since the mutually closing diodes is weakly bounded, Theorem 5.19 implies every $[X \implies XX]$ $[XY \implies YX]$ prefix-closed gizmo with finitely many locations is weakly bounded.

- What are the applications to complexity theory? Gadgets were invented for proving hardness, so studying them should give insight into complexity theory. Sometimes there are upper bounds corresponding to closed classes of gizmos; for instance, we claim without proof that targeted set reconfiguration with any finite set of weakly bounded gizmos is in NP. Universality has limited use for

proving hardness: it helps find simulations of gadgets one might need in a hardness proof, but it only resolves the complexity of one decision problem. We would like more results that go in the other direction, showing that any gadget with some properties can simulate a known-hard gadget. We know of two results of this form (which can easily be adapted to gizmos) from Demaine, Hendrickson, and Lynch [DHL20]: that every interacting tunnels reversible deterministic gadget simulates a locking 2-toggle, and that every nontrivial DAG (i.e. strictly bounded) gadget simulates either a crumbler or a dicrumbler.

7.3 Outside the system

Finally, we pose several problems about gadget simulation which are not specifically about gizmos. These are situations outside the limited scope of targeted set reconfiguration which gizmos are appropriate for. Each of these situations would require a new notion of simulation, and possibly a new notion of gadgets when gizmos are not sufficient. These appear to vary in difficulty from being resolved by a minor modification to gizmo simulation, to requiring a brand new formalism which likely needs to be significantly more complicated than gizmos.

Planarity. What are the right definitions of planar gizmos and planar simulations, and which of our results can be adapted to them? When is there a simulation but *not* a planar simulation? Prior work has extensively considered gadgets in planar environments [DGLR18, DHL20, ADGV15, HS04, Lyn20, DVW16, AAD$^+$20, Vig14, ADG$^+$21], so there are already a lot of known simulations and this would help solidify an important part of the motion-planning gadgets framework. Bosboom's thesis [Bos20] offers one approach: a planar gizmo would come with a cyclic ordering of its locations, and planar gizmo operations would need to respect this cyclic order.

Reconfiguration. Gizmos are able to handle targeted reconfiguration, but not reconfiguration when the agent is allowed to end anywhere. Can we define a notion of simulation which preserves hardness of reconfiguration problems? The difficulty arises when the agent ends inside a simulation; we would likely need to make sure that it is impossible to win the reconfiguration problem while inside a simulation.

Verified gadgets. As discussed in Section 3.5.2, there are structures built out of gadgets that seem like, but are not exactly, simulations. A limited notion of these, called verified gadgets, is considered by Lynch [Lyn20]. Can we define a notion of verified gadgets, or a more general type of simulation-like relationship, in terms of gizmos?

Input/output gadgets. Input/output gadgets, studied by Ani, Demaine, et al. [ADHL20], are the gadget framework's answer for fully deterministic situations, like a train following tracks according to the behavior of switches. Can we describe this situation using something like gizmos?

1-toggle-protected. 1-toggle-protected motion planning is when instead of connecting our gadgets with freely usable wires, we can only connect them with 1-toggles. This model is used by Akitaya et al. [ADG+21] to prove PSPACE-hardness for problems where traversing a wire requires moving a resource across it, so wires are naturally balanced. 1-toggle-protected motion planning seems applicable to any situation like that; what can we say about simulations with this constraint?

No branching hallways. To generalize both input/output gadgets and 1-toggle-protected motion planning, what if we change how we are allowed to connect locations? Our notion of simulations allows arbitrary connections, which you might call 'branching hallway-protected' (except that branching hallways can only connect countably many locations). If we do not allow arbitrary connections, the natural operation seems to be connecting and removing pairs of locations. To reintroduce merging many locations, we can allow the branching hallway in our simulations.[26] But when we 'connect' two locations, it does not need to be with a wire—it could be with a 1-toggle for 1-toggle-protected motion planning. Input/output gadgets can be thought of in terms of 'diode-protected' motion planning. Can we say anything in general about this kind of model?

Multiple robots. Often, such as in the situations considered by Holzer and Schwoon [HS04] and Akitaya et al. [ADG+21], an agent is allowed to control several 'robots' in disparate locations. To use the gadget framework, one must ensure that only one robot is 'active' at a time, so that there is effectively a since agent navigate the system. To define simulation for this situation, it seems that we need to allow multiple robots to be 'inside' a gadget at the same time, so the traversal sequences making up gizmos are not sufficient.

Multiplayer. Multiplayer games on systems of gadgets have been studied before [DHL20], and naturally describe many situations with multiple agents with conflicting goals. There are several issues that arise, and a full theory of multiplayer gadget simulation would need to address all of them. In addition to the issues that arise by having multiple navigators in the system of gadgets described above, here are some of them:

One-player incentives. We need to know not just which sequences are legal, but when one player can choose to make a sequence stop being legal. This is related the the one-player incentives discussed in Section 3.2.2. By analogy with the difference between 1-player SAT and 2-player QBF, it seems we need to replace traversal sequences with some notion of 'quantified' traversal sequences, possibly decorating each traversal with either $\forall$ or $\exists$.

Superposition. Somewhat similarly, for one-player motion planning the agent can always know in advance what they will need to do, and thus choose the correct state of a gadget whenever the gizmo representing it would

[26] In fact, the motion-planing gadget framework has been described this way [DGLR18].

enter a superposition. In multiplayer, which traversals a player will want available can depend on what the other player does, which can depend on which state of the superposition was chosen.

Timing. Hardness proofs for multiplayer games using the gadget framework need to worry about how long it takes players to do things. The sections of Demaine, Hendrickson, and Lynch [DHL20] on multiplayer problems spend a lot of time building complicated timer gadgets and ensuring that paths are just the right length to prevent each player from 'cheating.' Multiplayer simulation would need to consider how long it takes to traverse a simulation, and be robust to (or prevent) the other player from entering the simulation during this interval.

Visibility. If we want to consider games with imperfect information, such as those considered by Demaine, Hendrickson, and Lynch [DHL20], we need to define what information each player gets. We need some map from the location of a player to the set of states a gadget might be in given the information they can see at that location. How does such a map transform under the operations of simulation?

Bibliography

[AAD+20] Joshua Ani, Sualeh Asif, Erik D. Demaine, Yevhenii Diomidov, Dylan Hendrickson, Jayson Lynch, Sarah Scheffler, and Adam Suhl. PSPACE-completeness of pulling blocks to reach a goal. *Journal of Information Processing*, 28:929–941, 2020.

[ABD+20] Joshua Ani, Jeffrey Bosboom, Erik D. Demaine, Yenhenii Diomidov, Dylan Hendrickson, and Jayson Lynch. Walking through doors is hard, even without staircases: Proving PSPACE-hardness via planar assemblies of door gadgets. In *10th International Conference on Fun with Algorithms (FUN 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[ADG+21] Hugo A. Akitaya, Erik D. Demaine, Andrei Gonczi, Dylan H. Hendrickson, Adam Hesterberg, Matias Korman, Oliver Korten, Jayson Lynch, Irene Parada, and Vera Sacristán. Characterizing universal reconfigurability of modular pivoting robots. In *37th International Symposium on Computational Geometry (SoCG 2021)*, 2021.

[ADGV15] Greg Aloupis, Erik D. Demaine, Alan Guo, and Giovanni Viglietta. Classic Nintendo games are (computationally) hard. *Theoretical Computer Science*, 586:135–160, 2015.

[ADHL20] Joshua Ani, Erik D. Demaine, Dylan H. Hendrickson, and Jayson Lynch. Trains, games, and complexity: 0/1/2-player motion planning through input/output gadgets. *arXiv preprint arXiv:2005.03192*, 2020.

[Bos20] Jeffrey Jeffrey William Bosboom. *Exhaustive search and hardness proofs for games*. PhD thesis, Massachusetts Institute of Technology, 2020.

[DDHO03] Erik D. Demaine, Martin L. Demaine, Michael Hoffmann, and Joseph O'Rourke. Pushing blocks is hard. *Computational Geometry*, 26(1):21–36, 2003.

[DGLR18] Erik D. Demaine, Isaac Grosof, Jayson Lynch, and Mikhail Rudoy. Computational complexity of motion planning of a robot through simple gadgets. In *9th International Conference on Fun with Algorithms (FUN 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[DHL20] Erik D. Demaine, Dylan H. Hendrickson, and Jayson Lynch. Toward a general complexity theory of motion planning: Characterizing which gadgets make games hard. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[DVW16] Erik D. Demaine, Giovanni Viglietta, and Aaron Williams. Super Mario Bros. is harder/easier than we thought. In *8th International Conference on Fun with Algorithms (FUN 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[EK06] Birgit Engels and Tom Kamphans. Randolphs robot game is np-hard! *Electronic Notes in Discrete Mathematics*, 25:49–53, 2006. CTW2006 - Cologne-Twente Workshop on Graphs and Combinatorial Optimization.

[FG87] Aviezri S. Fraenkel and Elisheva Goldschmidt. PSPACE-hardness of some combinatorial games. *Journal of Combinatorial Theory, Series A*, 46(1):21–38, 1987.

[GLN14] Luciano Guala, Stefano Leucci, and Emanuele Natale. Bejeweled, candy crush and other match-three games are (np-) hard. In *2014 IEEE Conference on Computational Intelligence and Games*, pages 1–8. IEEE, 2014.

[Hig52] Graham Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, 3(1):326–336, 1952.

[Hol81] Ian Holyer. The NP-completeness of edge-coloring. *SIAM Journal on computing*, 10(4):718–720, 1981.

[HS04] Markus Holzer and Stefan Schwoon. Assembling molecules in ATOMIX is hard. *Theoretical Computer Science*, 313(3):447 – 462, 2004. Algorithmic Combinatorial Game Theory.

[JG79] David S. Johnson and Michael Garey. *Computers and Intractability*. 1979.

[Loe93] Martin Loebl. Gadget classification. *Graphs and Combinatorics*, 9(1):57–62, 1993.

[Lyn20] Jayson Lynch. *A Framework for Proving the Computational Intractability of Motion Planning Problems*. PhD thesis, Massachusetts Institute of Technology, 2020.

[Mey16] Bernd Meyer. Generalized Pete's Pike is PSPACE-complete. *Theoretical Computer Science*, 613:115–125, 2016.

[Ner58] A. Nerode. Linear automaton transformations. *Proceedings of the American Mathematical Society*, 9(4):541–544, 1958.

[Sza09] Jácint Szabó. Good characterizations for some degree constrained subgraphs. *Journal of Combinatorial Theory, Series B*, 99(2):436 – 446, 2009.

[TSSW00] Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson. Gadgets, approximation, and linear programming. *SIAM Journal on Computing*, 29(6):2074–2097, 2000.

[Tut54] W. T. Tutte. A short proof of the factor theorem for finite graphs. *Canadian Journal of Mathematics*, 6:347–352, 1954.

[Vig14] Giovanni Viglietta. Gaming is a hard job, but someone has to do it! *Theory of Computing Systems*, 54(4):595–621, 2014.