Computational Complexity of k-Block Conjugacy

Tyler Schrock   Rafael Frongillo
University of Colorado, Boulder

September 4, 2020

Abstract

We consider several computational problems related to conjugacy between subshifts of finite type, restricted to k-block codes: verifying a proposed k-block conjugacy, deciding if two shifts admit a k-block conjugacy, and reducing the representation size of a shift via a k-block conjugacy. We give a polynomial-time algorithm for verification, and show GI- and NP-hardness for deciding conjugacy and reducing representation size, respectively. Our approach focuses on 1-block conjugacies between vertex shifts, from which we generalize to k-block conjugacies and to edge shifts. We conclude with several open problems.

1 Introduction

One-dimensional subshifts of finite type (SFTs) are of fundamental importance in the study of symbolic dynamical systems. Despite their central role in symbolic dynamics, however, several basic questions about SFTs remain open, particularly with regard to computation. Most prominent is the conjugacy problem: whether it is possible to decide if two given SFTs are conjugate. In this work, we study restricted versions of the conjugacy problem, with an eye toward applications (algorithms to simplify representations of SFTs) as well as developing insights toward the full conjugacy problem. In particular, we address the computational complexity of deciding or verifying conjugacy when given a bound on the block size of the corresponding sliding block code. We focus on the case of vertex shifts; see below for other representations, notably edge shifts.

First consider the question of verification: given two vertex shifts and a proposed sliding block code, what is the computational complexity of verifying that the code induces a conjugacy? A polynomial-time algorithm is known for the irreducible case; we give a polynomial-time algorithm for the general case (§ 3). Second, the question of deciding k-block conjugacy: given two vertex shifts, what is the complexity of deciding if there exists a sliding block code, with block length at most k, that induces a conjugacy? By the first result on efficient verification, this problem is in NP; we show it to be GI-hard (at least as hard as the Graph Isomorphism problem) for all $k \geq 1$ (§ 4). Third, the question of reduction: given a vertex shift and integer ℓ, what is the complexity of deciding whether there exists a k-block conjugacy which reduces the number of vertices by ℓ? Extending a construction from previous work [5], we show that this problem, for $k = 1$, is NP-complete (§ 5). Similar problems have been studied for higher-dimensional shifts, and are generally Σ_1^0-complete or harder [1, 7, 13, 20].

It is interesting to contrast our results with those of the recent work [5], which studies the special case $k = 1$ with the restriction that the block code be a sequence of amalgamations. (Recall that any conjugacy can be expressed as a sequences of splittings followed by amalgamations; see § 2.) This previous work shows that the analogous version of our third problem, of reducing the number of vertices using only amalgamations, is NP-complete, but it does not address the verification problem; intuitively it seems plausible that verification would also be NP-hard. Returning to our setting, note that general 1-block codes need not be sequences of amalgamations (Figure 1). Thus, while it is unsurprising that the reduction problem remains NP-hard in our setting, it was perhaps unclear whether verification could be done in polynomial time, as a priori the number of splittings required could be super-polynomial. Finally, recent related work shows the decidability of conjugacy via higher-block codes, though the computational complexity of this problem is not discussed [3].

Edge shifts have received more attention in the literature, perhaps because of their succinct representations as integer matrices. Precisely because of their succinct representations, the question of verification is somewhat nuanced: verifying a given sliding block code requires writing down the proposed code, which can be exponential in the description size of the original edge shifts, so while the runtime of our algorithm can be exponential in the description sizes of the shifts, it is still polynomial-time (§ 6). We also show GI-hardness for the corresponding conjugacy problems, and leave several open questions (§ 7).

2 Setting

We begin with basic graph-theoretic definitions and convention. A directed graph $G = (V, E)$ is a set of vertices V along with a set of edges $E \subseteq V \times V$. When multiple graphs are in play, we will write $G = (V_G, E_G)$ to clarify which graphs the vertices or edges correspond to. For a directed graph $G = (V, E)$ and a vertex $v \in V$, we define $N^+(v) = \{u \in V : (v, u) \in E\}$ and $N^-(v) = \{u \in V : (u, v) \in E\}$ to be the set of out-neighbors and in-neighbors of v, respectively.

Unless specified otherwise, *a cycle of length* n will mean a sequence $v_1 v_2 \cdots v_n \in V$ such that $(v_i, v_{i+1}) \in E$ for $i \in \{1, \ldots, n+1\}$ where $v_{n+1} := v_1$. That is, $v_1 v_2 \cdots v_n$ is a cycle in our terminology if the path $v_1 v_2 \cdots v_n v_1$ forms a cycle in G. We define $C_n(G)$ to be the set of cycles of length n in $G = (V, E)$. Cycles are words (see below), and in particular a graph-theoretic cycle corresponds to multiple cycles in our terminology. For example, if the path $v_1 v_2 v_3 v_1$ forms a cycle in G, then $v_1 v_2 v_3, v_2 v_3 v_1, v_3 v_1 v_2 \in C_3(G)$.

Let $\mathcal{A}$ be a finite set. The *full shift* $\mathcal{A}^{\mathbb{Z}}$ over alphabet $\mathcal{A}$ is the set $\{(x_i)_{i \in \mathbb{Z}} : x_i \in \mathcal{A} \text{ for all } i \in \mathbb{Z}\}$. An element of $\mathcal{A}^{\mathbb{Z}}$ is called a *point*. A *block* (or *word*) in $\mathcal{A}$ is a string $a_1 a_2 \cdots a_n$ of symbols from $\mathcal{A}$. We will use the term *infinite word* to describe strings in $\mathcal{A}$ which are infinite in exactly one direction. If $x = (x_i)_{i \in \mathbb{Z}} \in \mathcal{A}^{\mathbb{Z}}$, we use $x_{[a,b]}$ for $a \leq b$ to denote the block $x_a x_{a+1} \cdots x_b$. Similarly, we use $x_{[a,\infty)}$ to denote the infinite word $x_a x_{a+1} \cdots$. Let $\mathcal{F}$ be a set of blocks over $\mathcal{A}$ called *forbidden blocks*. Then $X_{\mathcal{F}}$ is defined to be the subset of $\mathcal{A}^{\mathbb{Z}}$ where each $x \in X_{\mathcal{F}}$ contains none of the forbidden block in $\mathcal{F}$. A *shift space* (or *shift*) is a subset $X \subseteq \mathcal{A}^{\mathbb{Z}}$ such that $X = X_{\mathcal{F}}$ for some set of forbidden blocks $\mathcal{F}$. If there exists a finite set $\mathcal{F}$ such that $X = X_{\mathcal{F}}$, then X is called a *shift of finite type*.

Given a directed graph $G = (V, E)$, we associate to it the shift space $X_G = \{(v_i)_{i \in \mathbb{Z}} : v_i \in V, (v_i, v_{i+1}) \in E \text{ for all } i \in \mathbb{Z}\}$, which is the collection of all bi-infinite walks on G. Note that X_G is a shift of finite type with $\mathcal{F} = \{v_i v_j : (v_i, v_j) \notin E\}$. Any shift space of this form is called a *vertex shift*. Similarly, given a directed multigraph $G = (V, E)$, i.e. where E is a multiset, and an injective labeling of the edges from $\mathcal{A}$, we define the *edge shift* X_G^e of labelings of bi-infinite walks on G. Again edge shifts are shifts of finite type with $\mathcal{F} = \{e_1e_2 : e_1$ does not terminate at the initial vertex of $e_2\}$.

A shift X is *irreducible* if for every pair of words w_1, w_2 appearing in any points in X, there is a word w_3 such that $w_1w_3w_2$ appears as a word in some point in X. Similarly, X is *reducible* if it is not irreducible. In graph-theoretic terms, first consider any graph containing a vertex with either no out-neighbors or no in-neighbors. Such a vertex is called *stranded*. A graph (or multigraph) with no stranded vertices is called *essential*. A graph (or multigraph) with the property that for every pair of vertices u, v there is a path from u to v is called *strongly connected*. Finally, a vertex shift X_G (or edge shift X_G^e) is irreducible if G is essential and strongly connected. Even more, note that no point in X_G (or X_G^e) can pass through a stranded vertex and that stranded vertices can be efficiently trimmed from G (in an iterative fashion) to create an essential graph. Thus we assume without loss of generality that any directed graph (or multigraph) representing a (irreducible or reducible) vertex shift (or edge shift) is essential.

Given a shift X with alphabet $\mathcal{A}_1$, we can transform X into a shift space over another alphabet $\mathcal{A}_2$ in the following way. Fix integers m, a with $-m \le a$. Then letting $\mathcal{B}_n(X)$ denote the set of blocks of size n from the shift X and given a function $\Phi : \mathcal{B}_{m+a+1}(X) \to \mathcal{A}_2$, the corresponding *sliding block code* with memory m and anticipation a is the function Φ_∞ defined by $\Phi_\infty((x_i)_{i\in\mathbb{Z}}) = (\Phi(x_{[i-m,i+a]}))_{i\in\mathbb{Z}}$. That is, Φ_∞ looks at a block of size $m + a + 1$ through a window to determine a character from $\mathcal{A}_2$. Then the window is slid infinitely in both directions. Letting $k = m + a + 1$, we will call any sliding block code with window size k a *k-block code*. Given a sliding block code as $\Phi : \mathcal{A}_1^k \to \mathcal{A}_2$, we extend Φ to all finite and infinite words w of length at least k by $\Phi((w_i)_{i\in I}) = (\Phi(w_{[i-m,i+a]}))_{i-m,i+a\in I}$, where $I \subsetneq \mathbb{Z}$. That is, we extend Φ to words by sliding Φ over the entire word.

A sliding block code $\Phi_\infty : X \to Y$ which is bijective is called a *conjugacy*. If the conjugacy Φ_∞ is a k-block code, we call Φ_∞ a *k-block conjugacy*. We note that every conjugacy has an inverse conjugacy which is k'-block for some k'. Of particular concern for this paper are 1-block conjugacies, where we specifically point out that the inverse of a 1-block conjugacy is almost never 1-block.

Let X be any shift space with alphabet $\mathcal{A}_1$. We define the kth higher block shift $X^{[k]}$ with alphabet $\mathcal{A}_2 = \mathcal{B}_k(X)$ by the image of X under $\beta_N : X \to (\mathcal{B}_k(X))^{\mathbb{Z}}$ where for any point $p \in X$, $\beta_N(p)_i = p_{[i,i+k-1]}$. If $X = X_G$ happens to be a vertex shift, we can construct the kth higher block shift in terms of the graph. For any directed graph G, construct the graph $G^{[k]}$ by $V_{G^{[k]}} = \{v_1 \cdots v_k : v_1 \cdots v_k$ is a path in $G\}$ and $E_{G^{[k]}} = \{(v_1v_2 \cdots v_k, v_2 \cdots v_kv_{k+1}) : v_1 \cdots v_{k+1}$ is a path in $G\}$. Then $X_{G^{[k]}} = X_G^{[k]}$. When dealing with k-block codes, it is often useful to *pass to a higher block shift* by noting that there is a k-block conjugacy $\Phi_\infty : X \to Y$ if and only if there is a 1-block conjugacy $\Phi_\infty^{[k]} : X^{[k]} \to Y$ [12, Proposition 1.5.12].

Furthermore, any sliding block code $\Phi_\infty : X_G \to X_H$ between vertex shifts induces the function $\Phi_{\mathsf{c}} : \bigcup_{n=1}^\infty C_n(G) \to \bigcup_{n=1}^\infty C_n(H)$, which we define as follows. Given a cycle c in G, there is a unique cycle d in H with $|c| = |d|$ such that $\Phi_\infty(c^\infty.c^\infty) = d^\infty.d^\infty$; we set $\Phi_{\mathsf{c}}(c) = d$. Note that we have overloaded c^∞ to mean both a right infinite and a left infinite word; while the notation is overloaded, the meaning will always be clear from context. In the special case of a 1-block code, the block map $\Phi : \mathcal{A}_1 \to \mathcal{A}_2$ is simply a map between the alphabets. In this case, when X_G is a vertex shift, we have

$\Phi_{\mathsf{c}}(v_1 \cdots v_n) = \Phi(v_1) \cdots \Phi(v_n)$.

Definition 1. Let X_G be a vertex shift. We say vertices $u, v \in V_G$ can be *amalgamated* if one the following conditions is met.

1. $N^+(u) = N^+(v)$ and $N^-(u) \cap N^-(v) = \emptyset$
2. $N^-(u) = N^-(v)$ and $N^+(u) \cap N^+(v) = \emptyset$

We say u and v are amalgamated when they are replaced by the vertex uv which has $N^+(uv) = N^+(u) \cup N^+(v)$ and $N^-(uv) = N^-(u) \cup N^-(v)$.

Definition 2. Let X_G be a vertex shift. A vertex $v \in V_G$ can be split into two vertices v_1 and v_2 provided the edges of v_1, v_2 satisfy one of the following conditions.

1. $\{N^+(v_1), N^+(v_2)\}$ is a partition of $N^+(v)$ and $N^-(v_1) = N^-(v_2) = N^-(v)$.
2. $\{N^-(v_1), N^-(v_2)\}$ is a partition of $N^-(v)$ and $N^+(v_1) = N^+(v_2) = N^+(v)$.

The corresponding new graph is called a *state splitting* of v. Note that state splittings and amalgamations are inverse operations.

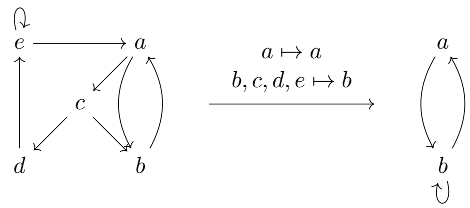
The definitions for edge shifts are similar. Since edges shifts are based on multigraphs, $N^-(v)$ and $N^+(v)$ are multisets. The definition of a state splitting is identical noting that the partition is a multiset partition. For amalgamations, two vertices u, v can be amalgamated if $N^-(u) = N^-(v)$ or $N^+(u) = N^+(v)$. In the case where $N^-(u) = N^-(v)$, u, v are replaced by a single vertex uv with $N^-(uv) = N^-(u) = N^-(v)$ and $N^+(uv) = N^+(u) \uplus N^+(v)$, where $N^+(u) \uplus N^+(v)$ is the multiset disjoint union.

Theorem 3 ([12, 19])**.** *Let X_G, X_H be vertex shifts (or edge shifts). Then X_G and X_H are conjugate if and only if there is a sequence of state splittings followed by a sequence of amalgamations which transform G into H.*

In the case of a 1-block code $\Phi : V_G \to V_H$, we may view the block map as a partition of the vertices of G, where each element of the partition is converted to a vertex of H. In light of Theorem 3, it may be tempting to think that every 1-block code can be written as a sequence of amalgamations only, as intuitively splitting a vertex while requiring the vertices be re-amalgamated has no benefit. Yet this statement is not true; there are simple examples of two graphs admitting a 1-block conjugacy, where no pair of vertices can be amalgamated in either graph (Figure 1).

We conclude the background with a common way a sliding block code can fail to be injective. Given a k-block code $\Phi_\infty : X_G \to X_H$, if there exist distinct words $w_1 w_2 w_3, w_1 w_2' w_3$ appearing in X_G with $|w_1| = |w_3| = k$, such that $\Phi(w_1 w_2 w_3) = \Phi(w_1 w_2' w_3)$, we say Φ_∞ *collapses a diamond*. As we now state, if a sliding block code is injective, it cannot collapse a diamond. (As we discuss in § 3.2, if Φ_{c} is injective, collapsing a diamond is actually the only way Φ_∞ can fail to be injective.) We prove the result for completeness; see, e.g., [12, Theorem 8.1.16] for a similar result in the irreducible case.

Lemma 4. *Let $\Phi_\infty : X_G \to X_H$ be a k-block code. If Φ collapses a diamond, then Φ_∞ is not injective.*

(a) 

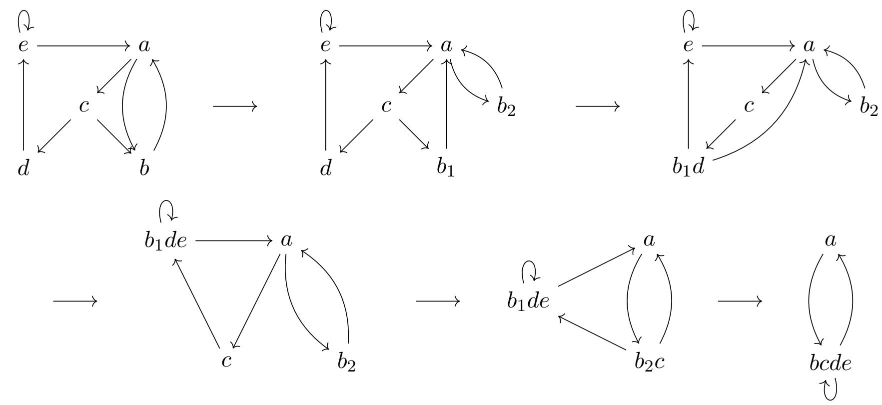
(b) 

Figure 1: (a) Two vertex shifts which are conjugate by a 1-block code but not by a sequence of amalgamations. In fact, one can verify by checking all smaller graphs that this is a minimal such example. (b) The conjugacy, demonstrated via a splitting followed by four amalgamations.

Proof. Suppose Φ collapses a diamond. That is, $\Phi(w_1w_2w_3) = \Phi(w_1w_2'w_3)$ for some words w_1, w_3 of length k and distinct words w_2, w_2' in G. Consider any infinite word w_0 which can precede w_1 and any infinite word w_4 which can follow w_3. Then we have $\Phi_\infty(w_0w_1w_2w_3w_4) = \Phi_\infty(w_0w_1w_2'w_3w_4)$, so Φ_∞ is not injective. □

3 Verification: Testing a k-Block Map for Conjugacy

Given a pair of directed graphs G, H, and a proposed k-block map Φ, we wish to verify whether or not Φ induces a conjugacy between the vertex shifts X_G, X_H. We will focus in this section on the case $k = 1$, as the case $k > 1$ follows immediately by recoding to the kth higher block shift. When G and H are irreducible (strongly connected), this problem boils down to checking that the two graphs have the same number of cycles of each length up to some constant, and furthermore that Φ induces an injection on these cycles. Cycle counting can be done efficiently using powers of the adjacency matrices, and an efficient injectivity check is given by Sutner [16]. We detail the full procedure in § 3.1 for completeness.

One may expect the general case, when G and H need not be strongly connected, to be much more complex. Reducible vertex shifts are perhaps most naturally understood by first decomposing them into irreducible components. As such, tThe difficulty of the general case is evidenced by the failure of several proposed algorithms wherein one subdivides the graph into its irreducible components and uses the algorithm for the irreducible case on each, together with some other global checks; in § 3.2, we give counter-examples to several such statements. [1] Instead, we solve the general case more directly by reduction to the irreducible case: we efficiently augment the graphs and block map with new vertices and edges, until the resulting graphs are irreducible, in such a way as to preserve conjugacy (or lack thereof).

3.1 Irreducible Case

As described above, we will focus first on 1-block codes. An efficient algorithm to verify a given 1-block code is known in the case where G, H are irreducible, from the cellular automata literature [16, Theorem 5]. For completeness, we restate a version of this result in the language of symbolic dynamics.

The following straightforward topological result allows us to restrict attention to the map induced on cycles between the graphs. The proof follows from three facts: the image of a compact set under a continuous map is itself compact, compact sets are closed in a Hausdorff space, and a closed set containing a set dense in X is all of X. [2]

Proposition 5. *Suppose X, Y are compact metric spaces, $\psi : X \to Y$ is continuous, and $D \subseteq Y$ is a dense subset of Y. If ψ surjects onto D, then ψ surjects onto all of Y.*

We will apply Proposition 5 with D being the set of periodic points of X_H. The following result, that Φ induces a 1-block conjugacy if and only if it induces a bijection on cycles, appears to be known (see e.g. [12, Exercise 2.3.6]); we give the proof for completeness.

Theorem 6. *Irreducible vertex shifts X_G, X_H are conjugate via a 1-block code if and only if there is a vertex map $\Phi : V_G \to V_H$ such that the induced map Φ_{c} is a bijection.*

Proof. If Φ_∞ is a conjugacy, then it is a bijection on periodic points, which are in bijection with cycles of least period; we conclude Φ_{c} is a bijection. For the converse, suppose Φ_{c} is bijective. Then every periodic point in X_H is mapped to by Φ_∞. Since the periodic points are a dense subset of the compact metric space X_H, by Proposition 5 Φ_∞ is surjective. For injectivity, by contrapositive, suppose Φ_∞ is not injective, so there exist distinct points $p, q \in X_G$ such that $\Phi_\infty(p) = \Phi_\infty(q)$. We proceed in cases.

(Case 1) Suppose first that p, q disagree at $|V_G|^2 + 1$ consecutive indices, meaning the words $p_{[a,b]}, q_{[a,b]}$ disagree at every index for $a, b \in \mathbb{Z}$ with $b - a = |V_G|^2 + 1$. Consider all possible pairs of vertices in G; there are $|V_G|^2$ such pairs. Thus there exist distinct indices $c, d \in \{a, a+1 \ldots, b\}$ such that $(p_c, q_c) = (p_d, q_d)$. But then $\Phi_{\mathsf{c}}(p_{[c,d-1]}) = \Phi_{\mathsf{c}}(q_{[c,d-1]})$.

(Case 2) Suppose instead that p, q do not disagree at $|V_G|^2 + 1$ consecutive indices: there exist indices a, b with $a < b - 1$ such that p, q agree at indices a and b, but p, q disagree at every index between a and b. Let w be any word connecting $p_b = q_b$ to $p_a = q_a$. Then $\Phi_{\mathsf{c}}(p_{[a,b]}w) = \Phi_{\mathsf{c}}(q_{[a,b]}w)$. □

To verify that the cycle map Φ_{c} is bijective, we will test for injectivity explicitly, and rely on counting arguments to check surjectivity. For injectivity, it turns out that checking cycles up to length $|V_G|^2$ suffices.

[1] As further evidence that the reducible case might be more complex, consider the problem of minimizing right-resolving presentations of a sofic shifts; there is an efficient algorithm for irreducible shifts [8], while the general case is likely to be PSPACE-complete [14].

[2] We thank an anonymous reviewer for this succinct proof.

Proposition 7. *Suppose $\Phi_\infty : X_G \to X_H$ is a 1-block code between irreducible vertex shifts. If Φ_{c} is injective on $\bigcup_{n=1}^{|V_G|^2} C_n(G)$, then Φ_{c} is injective.*

Proof. Let c, d be distinct cycles of size $|c| = |d| = k > |V_G|^2$. Proceeding by strong induction, suppose Φ_{c} is injective on all cycles of size less than k. There are $|V_G|^2$ possible pairs of vertices in G. Thus there exist distinct indices a, b such that $(c_a, d_a) = (c_b, d_b)$. That is, $c_{[a,b-1]}, d_{[a,b-1]}$ are cycles of the same length and $c_{[b,a-1]}, d_{[b,a-1]}$ are cycles of the same length. Since c, d were distinct, we can assume without loss of generality that $c_{[a,b-1]}, d_{[a,b-1]}$ are distinct. By the induction hypothesis, $\Phi_{\mathsf{c}}(c_{[a,b-1]}) \neq \Phi_{\mathsf{c}}(d_{[a,b-1]})$. Thus $\Phi_{\mathsf{c}}(c) \neq \Phi_{\mathsf{c}}(d)$. □

Proposition 7 suggests the naïve algorithm of checking all cycles up to length $|V_G|^2$ to verify injectivity of Φ_{c}. This algorithm is remarkably inefficient, however; letting $n = |V_G|$, there can be $\Omega(n^{n^2})$ cycles of length up to n^2, as is the case for the complete graph. Fortunately, these checks can be performed much more efficiently, as noted by Sutner [16], by rephrasing them as a search problem in a graph built from pairs of vertices in G. This procedure is outlined in Algorithm 1.

Theorem 8 ([16, Theorem 5])**.** *Let X_G be a vertex shift and $A = \{1, 2, \ldots, m\}$. Then any given map $\Phi : V_G \to A$ induces a map $\Phi_{\mathsf{c}} : \bigcup_n C_n(G) \to \bigcup_n A^n$. Deciding if Φ_{c} is injective can be determined in $O(|V_G|^4)$ time.*

Proof. First we build the directed meta-graph $M = (V_M, E_M)$ where $V_M = \{(u, v) : u, v \in V_G\}$ and $E_M = \{((u_1, v_1), (u_2, v_2)) : \Phi(u_1) = \Phi(v_1), \Phi(u_2) = \Phi(v_2) \text{ and } (u_1, u_2) \in E_G, (v_1, v_2) \in E_G\}$. That is, M is a graph on pairs of vertices from G, with an edge connecting pairs P_1, P_2 if and only if (i) there is a pair of (possibly non-distinct) edges in G connecting the two vertices in P_1 to the vertices in P_2, and (ii) the induced map on words of length two (i.e., edges) maps the two edges together. M can be constructed in $O(|V_G|^4)$ time.

Given M, the map Φ_{c} is injective if an only if there is no cycle in M which passes through a vertex $(v_1, v_2) \in V_M$ with $v_1 \neq v_2$. Furthermore, such a cycle in M exists if and only if M has a strongly connected component containing an edge and a vertex (u, v) with $u \neq v$. Tarjan's strongly connected components algorithm [18] now applies, in $O(|V_M| + |E_M|) = O(|V_G|^4)$ time. □

Putting the above results together with the higher-block codes gives the desired algorithm to verify k-block conjugacies; the full conjugacy algorithm for $k = 1$ is outlined in Algorithm 2. As the description size of a k-block code is $\Omega(|V_G|^k)$, the algorithm runs in polynomial time.

Corollary 9. *Given a k-block code $\Phi_\infty : X_G^k \to X_H$ between irreducible vertex shifts, deciding if Φ_∞ is a conjugacy is in P. In particular, it can be determined in $O(|V_G|^{4k})$ time.*

Proof. Given G, H, we first pass to the kth higher block shift $X_{G^{[k]}}$ of X_G, recalling that $\Phi_\infty^{[k]}$ is a 1-block code and Φ_∞ is a conjugacy if and only if $\Phi_\infty^{[k]}$ is a conjugacy [12, Proposition 1.5.12]. We can construct $\Phi_\infty^{[k]} : X_{G^{[k]}} \to X_H$ in time $O(|V_{G^{[k]}}| + |E_{G^{[k]}}|) = O(|V_{G^{[k]}}|^3)$. Noting that $|V_{G^{[k]}}| \leq |V_G|^k$, it thus suffices to show the case $k = 1$.

By Theorem 6, Φ_∞ is a conjugacy if and only if Φ_{c} is a bijection. As $k = 1$, Theorem 8 shows that injectivity of Φ_{c} can be determined in $O(|V_G|^4)$ time. To show Φ_{c} is surjective, it suffices to check that $|C_i(G)| = |C_i(H)|$ for all $i \in \mathbb{N}$. Letting $A(G), A(H)$ be the adjacency matrices of G, H, we note $|C_i(G)| = \operatorname{tr}(A(G)^i)$, so our desired check is equivalent to checking $\operatorname{tr}(A(G)^i) = \operatorname{tr}(A(H)^i)$ for all $i \in \mathbb{N}$ [12, Proposition 2.2.12]. This equality is in turn equivalent to $A(G)$ and $A(H)$ have the same nonzero eigenvalues, meaning the characteristic polynomials p_G,p_H satisfy $p_G(x) = x^\ell p_H(x)$ for some $\ell \in \mathbb{N}$. We can now check this condition by computing the coefficients of p_G, p_H in $O(|V_G|^\omega \log |V_G|)$ time [4, 10], where ω is the exponent of matrix multiplication, and then checking the above equality in $O(|V_G|^2 \log |V_G|)$ time. As $\omega < 3$, the overall runtime is therefore $O(|V_G|^4)$. □

3.2 General Case

As discussed above, the general case appears much more complex. In particular, several useful statements about conjugacy between irreducible vertex shifts fail to hold in the general case. First, given a sliding block code $\Phi_\infty : X_G \to X_H$ between irreducible vertex shifts, it is known that if Φ_∞ is injective and G, H have the same topological entropy, then Φ_∞ is a conjugacy [12, Corollary 8.1.20]. (The topological entropy of a shift X is defined as $h(X) = \lim\limits_{n\to\infty} \frac{1}{n} \log_2 |\mathcal{B}_n(X)|$.) If the shifts are reducible, however, Φ_∞ can satisfy these conditions but fail to be surjective (Figure 2a). Second, we have from Theorem 6 that if Φ_∞ is a 1-block code between irreducible vertex shifts, then Φ_c being a bijection implies Φ_∞ is bijective. In the reducible case, Φ_c can be bijective while Φ_∞ fails injectivity (Figure 2b) or surjectivity (Figure 2a).

As an even stronger test, one might guess for general vertex shifts that if $\Phi_\infty : X_G \to X_H$ is surjective and the induced maps between irreducible subgraphs are all conjugacies, then Φ_∞ is a conjugacy. If true, this statement would suggest applying the algorithm in Corollary 9 to each irreducible subgraph, at which point one would only need to test surjectivity. Yet this statement is also false; Φ_c being a bijection implies neither the injectivity nor the surjectivity of Φ_∞ (Figure 2). By extending the argument of Theorem 6, one can correct the statement by adding a check for diamonds: if Φ_∞ is surjective, the induced maps between irreducible subgraphs are all conjugacies, and Φ does not collapse a diamond, then Φ_∞ is a conjugacy. Unfortunately, while this revised statement does break the problem of verifying a proposed 1-block conjugacy into more manageable pieces, how to turn it into a decision procedure, let alone an efficient algorithm, is far from clear.

To verify a potential conjugacy between vertex shifts efficiently, we will instead apply a more direct reduction to the irreducible case. Given a 1-block code $\Phi_\infty : X_G \to X_H$ between reducible vertex shifts, we will extend G and H to irreducible graphs while preserving the conjugacy or non-conjugacy of Φ_∞. The key operation for this extension is the following procedure, which adds a new sink vertex to the graph G and H in such a way as to preserve conjugacy/non-conjugacy. We will then apply this procedure in reverse to add a new source vertex, at which point we have enough structure to connect the new sink vertex to the new source vertex, which renders both graphs irreducible.

Let $\{T_i\}_{i\in I}$ be the sink components of H and $T'_i = \Phi^{-1}(T_i)$ be the subgraph of G which maps to T_i under Φ_∞. The procedure is as follows:

1. For each i, pick an arbitrary vertex v_i in T_i.

2. For each i, pick an arbitrary cycle c_i in T_i ending at v_i of length $|c_i| \le |T_i|$.

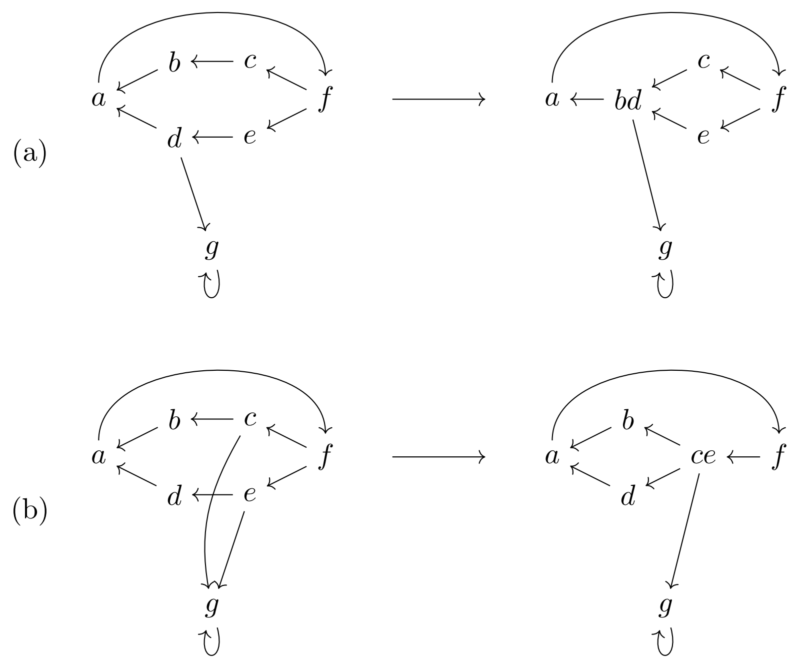


Figure 2: Counter-examples showing various statements which hold in the irreducible case fail in the reducible case. Note that all four shifts have the same topological entropy, $h(X) = \frac{1}{4}$. (a) A 1-block code between two reducible shifts which restricts to conjugacies between the irreducible components (and hence Φ_{c} is a bijection) but is not surjective. (b) A 1-block code between two reducible shifts which restricts to conjugacies between the irreducible components but is not injective.

3. Form the graph $\hat{H}$ by adding a vertex t to H along with the edge (t, t) and the edges $\{(v_i, t)\}_{i \in I}$.

4. For each i, define V_i' to be the collection of all the vertices $v' \in \Phi^{-1}(v_i)$ which are followed by an infinite word w' such that $\Phi(v'w') = v_i c_i^{\infty}$.

5. Form the graph $\hat{G}$ by adding a vertex t' to G along with the edge (t', t') and the edges $\{(v', t') : v' \in V_i'\}_{i \in I}$.

6. Define $\hat{\Phi}_\infty : X_{\hat{G}} \to X_{\hat{H}}$ by $\hat{\Phi}(u) = \begin{cases} \Phi(u), & \text{if } u \neq t' \\ t, & \text{if } u = t' \end{cases}$.

Proposition 10. *Let $\Phi_\infty : X_G \to X_H$ be a 1-block code between reducible vertex shifts. Then $\hat{\Phi}_\infty : X_{\hat{G}} \to X_{\hat{H}}$ as described above is a conjugacy if and only if Φ_∞ is a conjugacy.*

Proof. Since X_G is a subshift of $X_{\hat{G}}$ (and similarly for H) and $\hat{\Phi}_\infty$ preserves Φ_∞, it immediately follows that Φ_∞ is a conjugacy whenever $\hat{\Phi}_\infty$ is. For the converse, suppose $\hat{\Phi}_\infty$ is not a conjugacy.

If $\hat{\Phi}_\infty$ is not injective, we have distinct points $p_1, p_2 \in X_{\hat{G}}$ such that $\hat{\Phi}_\infty(p_1) = \hat{\Phi}_\infty(p_2) = q$. If $q \in X_H$, then $p_1, p_2 \in X_G$ by the definition of $\hat{\Phi}_\infty$, so Φ_∞ is not injective. If $q \notin X_H$, then $q = wv_it^\infty$. By the definition of $\hat{\Phi}_\infty$, we have $p_1 = w_1v't'^\infty$ and $p_2 = w_2v''t'^\infty$ where $w_1v' \neq w_2v''$. By the construction of $N^-(t')$ in step 5 above, there exist infinite words w', w'' such that $v'wv', v''w''$ are words in G and $\Phi(v'w') = v_ic^\infty = \Phi(v''w'')$. Thus $\Phi_\infty(w_1v'w') = \Phi_\infty(w_2v''w'')$, and Φ_∞ is not injective.

If $\hat{\Phi}_\infty$ is not surjective, then there exists $p \in X_{\hat{H}}$ which is not mapped to. If $q \in X_H$, then Φ is not surjective. Otherwise, $q \notin X_H$, so $q = wv_it^\infty$ for some $i \in I$. But noting the construction of $N^-(t')$ in step 4 above, we added an edge from v' to t' if and only if v' was followed by an infinite word w' such that $\Phi(v'w') = v_ic_i$. That is, we selectively added edges from vertices in H to t' so that the point wv_it^∞ is mapped to if and only if the point $wv_ic_i^\infty$ is mapped to. Thus Φ_∞ was still already not surjective. □

We now construct the final graphs G^* and H^* by

1. First, using the procedure above to add new sink vertices t, t' to each graph.

2. Reverse the edges in the two graphs; the new sink vertices from the step above are now source vertices. Use the procedure above to again add new sink vertices s, s'. Reverse the edges in the two graphs once again. The newly added vertices s, s' are now newly added source vertices.

3. Form the strongly connected graphs G^* and H^* by adding the edge (t, s) to H and the edge (t', s') to G.

4. Define $\Phi_\infty^* : X_{G^*} \to X_{H^*}$ by $\Phi^*(u) = \begin{cases} \Phi(u), & \text{if } u \notin \{t', s'\} \\ t, & \text{if } u = t' \\ s, & \text{if } u = s' \end{cases}$.

Proposition 11. *Let $\Phi_\infty : X_G \to X_H$ be a 1-block code between reducible vertex shifts. Then $\Phi_\infty^* : X_{G^*} \to X_{H^*}$ as described in the construction above is a conjugacy if and only if Φ_∞ is a conjugacy.*

Proof. By Proposition 10, $\hat{\Phi}_\infty : X_{\hat{G}} \to X_{\hat{H}}$ is a conjugacy if and only if Φ_∞ is a conjugacy, where the graphs $\hat{G}, \hat{H}$ immediately precede the addition of the edges (t, s) and (t', s'). As in the proof of Proposition 10, $X_{\hat{G}}$ is a subshift of X_{G^*} (and similarly for $\hat{H}$) and Φ^*_∞ preserves Φ_∞, so $\hat{\Phi}_\infty$ is a conjugacy if Φ^*_∞ is. For the other direction, suppose Φ^*_∞ is not a conjugacy.

First suppose Φ^*_∞ is not injective. Since G^*, H^* are irreducible, we have cycles c, d in G^* from Theorem 6 such that $\Phi^*_{\mathsf{c}}(c) = \Phi^*_{\mathsf{c}}(d)$. Without loss of generality, we can assume $c = s'w_ct'$ and $d = s'w_dt'$, where $w_c \neq w_d$. Thus $\hat{\Phi}$ collapses the diamond $(s'w_ct', s'w_dt')$, so by Lemma 4, $\hat{\Phi}_\infty$ is not injective.

Now suppose Φ^*_∞ is not surjective. Again by Theorem 6, we know there is a cycle c which is not in the image of Φ^*_{c}. Without loss of generality, we can assume $c = swt$, where w does not contain s or t. But then $s^\infty.wt^\infty$ is a point in $X_{\hat{H}}$ which is not in the image of $\hat{\Phi}_\infty$. □

We now have that given reducible vertex shifts X_G, X_H and a proposed 1-block conjugacy between them, the shifts can be embedded into irreducible shifts such that the conjugacy or non-conjugacy is preserved. Next we show this embedding can be performed efficiently; the procedure described in the proof is outlined in Algorithm 5.

Theorem 12. *Given reducible vertex shifts X_G, X_H and a 1-block code as $\Phi : V_G \to V_H$, the graphs G^* and H^* can be constructed in $O(|V_G|^3)$ time.*

Proof. Let T be an arbitrary sink component in H and T' be the subgraph $\Phi^{-1}(T)$ of G. We will show the corresponding neighbors of t, t' can be selected in $O(|V_{T'}|^3)$ time. Iterating over all sink components $T \in \mathcal{T}$ and source components $S \in \mathcal{S}$ will give an overall complexity of $O(\sum_{T' \in \mathcal{T}'} |V_{T'}|^3 + \sum_{S' \in \mathcal{S}'} |V_{S'}|^3) = O(|V_G|^3)$ time. (Adding the edges $(t, s), (t', s')$ takes constant time.)

Let v be an arbitrary vertex of T, and let c be the shortest cycle in T through v, which can be computed using breadth-first search in $O(|V_T| + |E_T|) = O(|V_H|^2) = O(|V_G|^2)$ time. Note that $|c| \leq |T|$, so we have completed steps 1 and 2. Step 3 is constant time. The only nontrivial step that remains is step 4, the computation of the set $V' \subseteq V_{T'}$, from which steps 5 and 6 follow trivially in linear time.

Let $C = (V_C, E_C)$ be the subgraph of T corresponding to c, and let $C' = (V_{C'}, E_{C'})$ be the subgraph of T' which maps onto C as follows: $V_{C'} = \Phi^{-1}(V_C)$, and $E_{C'} = \{(u', v') \in E_{T'} : (\Phi(u'), \Phi(v')) \in E_C\}$. The subgraph C' can be constructed in $O(|V_{T'}|^2)$ time. Note that infinite walks in C' starting from any $v' \in \Phi^{-1}(v)$ are precisely the walks in T' that map onto c^∞, and moreover, there is an infinite walk in C' starting from v' if and only if there is a path in C' from v' to a cycle in C'. We therefore define $V' \subseteq \Phi^{-1}(v) \subseteq V_{C'}$ to be the set of vertices v' such that there is a path in C' from v' to a cycle in C'. To compute V', we can simply run breadth-first search from each vertex in $\Phi^{-1}(v)$, in $O(|\Phi^{-1}(v)| \cdot (|V_{C'}| + |E_{C'}|)) = O(|V_{T'}|^3)$ time. □

We have now seen an efficient procedure to embed a pair of reducible graphs into a pair of irreducible graphs, such that the original pair admits a 1-block conjugacy if and only if the embedded pair does. Moreover, the embedded irreducible graphs have at most twice the number of vertices as the original graphs. With this procedure in hand, we can extend our verification algorithm to the reducible case.

Corollary 13. *Given vertex shifts X_G, X_H and a k-block code Φ_∞ as $\Phi : V_G^k \to V_H$, deciding if Φ_∞ is a conjugacy can be determined in $O(|V_G|^{4k})$ time.*

Proof. If G, H are irreducible, Corollary 9 applies immediately. For the reducible case, as in Corollary 9, by passing to the kth higher block shift it suffices to show the case $k = 1$. From Theorem 12, we can embed G, H into the irreducible shifts G^*, H^* in $O(|V_G|^3)$ time. Furthermore, $|V_{G^*}| < 2|V_G|$, so $|V_{G^*}| = O(|V_G|)$. Then by Corollary 9, we can verify if Φ_∞^* (and hence Φ_∞) is a conjugacy in $O(|V_{G^*}|^4) = O(|V_G|^4)$. □

4 Deciding k-Block Conjugacy

We now turn to the question of deciding k-block conjugacy. Specifically, we wish to understand the complexity of the problem k-BC below, to decide whether vertex shifts X_G, X_H are conjugate via a k-block code $\Phi_\infty : X_G \to X_H$. Note that the description size of Φ is polynomial in $|V_G|$ and $|V_H|$, and thus from Corollary 13 we know that a potential k-block conjugacy can be verified in polynomial time; hence, k-BC is in NP. We will show that k-BC is GI-hard for all k, where GI is the class of problems with a polynomial-time Turing reduction to the Graph Isomorphism problem [11]. (A graph isomorphism is bijection between the vertices of two graphs which preserves the edges/non-edge relation; the Graph Isomorphism problem is to decide if two given undirected graphs are isomorphic.)

Definition 14. Given directed graphs G, H, the *k-Block Conjugacy Problem*, denoted k-BC, is to decide if there is a k-block conjugacy $\Phi_\infty : X_G \to X_H$ between the vertex shifts X_G and X_H.

To begin, we give the straightforward result that the case $k = 1$ is GI-hard, essentially because 1-block conjugacies between vertex shifts for equal sized graphs must be isomorphisms. The result appears to be well-known, and follows from e.g. Salo and Törmä [15, Theorem 1]. Our proof shows something slightly stronger, that the problem is hard even when restricting to irreducible shifts.

Theorem 15. *The 1-Block Conjugacy Problem,* 1-BC*, is* GI*-hard.*

Proof. Given strongly connected graphs directed G, H with $|V_G| = |V_H|$, we show that the shifts X_G, X_H are conjugate via 1-block code if and only if the graphs are isomorphic (cf. [12, Ex. 2.2.14]). The result then follows as graph isomorphism between strongly connected directed graphs is GI-hard, by the usual reduction from the undirected case (replace each edge with two directed edges).

First suppose $\Psi : V_G \to V_H$ is a graph isomorphism. As $\Psi(v_1v_2)$ is a legal word in X_H for all words of length 2, by definition of a graph isomorphism, we have that $\Psi_\infty : X_G \to X_H$ is a valid 1-block code. Letting $\Phi = \Psi^{-1} : V_H \to V_G$, we have $\Psi_\infty(\Phi_\infty((x_i)_{i\in\mathbb{Z}})) = (\Psi(\Phi(x_i)))_{i\in\mathbb{Z}} = (x_i)_{i\in\mathbb{Z}}$ for all $x \in X_H$, and $\Phi_\infty(\Psi_\infty((x_i)_{i\in\mathbb{Z}})) = (\Phi(\Psi(x_i)))_{i\in\mathbb{Z}} = (x_i)_{i\in\mathbb{Z}}$ for all $x \in X_G$. Thus, Φ_∞ is the 2-sided inverse of Ψ_∞, and Ψ_∞ is a 1-block conjugacy.

For the other direction, suppose $\Phi_\infty : X_G \to X_H$ is a 1-block conjugacy. Then $\{\Phi(v) : v \in V_G\}$ must be exactly the set of words of length 1 in X_H, i.e., the vertices of H. Since $|V_G| = |V_H|$, $\Phi : V_G \to V_H$ is a bijection. Also, for any edge $(v_1, v_2) \in E_G$, we have $\Phi(v_1v_2) = \Phi(v_1)\Phi(v_2)$, so $(\Phi(v_1), \Phi(v_2)) \in E_H$ as Φ_∞ is a well-defined sliding block code. Even more, consider any pair v_3, v_4 of vertices in V_G such that $(v_3, v_4) \notin E_G$. Noting that $\Phi(v_3)\Phi(v_4)$ has a unique preimage as Φ is a bijection and Φ_∞ is surjective, we have $(\Phi(v_3), \Phi(v_4)) \notin E_H$. Thus $\Phi : V_G \to V_H$ is a bijection on vertices which preserves the edge relationship; that is, Φ is a graph isomorphism. □

(a)
$$\begin{tikzcd}
& & & & & v_k^t \arrow[dr] & \\
v \arrow[r, squiggly] & v_{\text{in}} \arrow[r] & v_1 \arrow[r] & \cdots \arrow[r] & v_{k-1} \arrow[ur] \arrow[dr] & & v_{\text{out}} \\
& & & & & v_k^b \arrow[ur] &
\end{tikzcd}$$

(b)
$$\begin{tikzcd}
& & u_1^t \arrow[r] & \cdots \arrow[r] & u_{k-1}^t \arrow[r] & u_k^t \arrow[dr] & \\
u \arrow[r, squiggly] & u_{\text{in}} \arrow[ur] \arrow[dr] & & & & & u_{\text{out}} \\
& & u_1^b \arrow[r] & \cdots \arrow[r] & u_{k-1}^b \arrow[r] & u_k^b \arrow[ur] &
\end{tikzcd}$$

Figure 3: The vertex gadgets for (a) each vertex v in G, and (b) each vertex u in H.

Next, we will show k-BC is GI-hard for all k, by reduction to the 1-block case. Specifically, given directed graphs G, H, we will construct graphs G', H' such that there exists a 1-block conjugacy $\Phi_\infty : X_G \to X_H$ if and only if there exists a k-block conjugacy $\Phi'_\infty : X_{G'} \to X_{H'}$ exists. To form G', we replace every vertex $v \in V_G$ with a path $v_{\text{in}}v_1v_2 \cdots v_{k-1}$ followed by the diamond with sides $v_{k-1}v_k^t v_{\text{out}}$ and $v_{k-1}v_k^b v_{\text{out}}$ (Figure 3a). To form H', we replace every vertex $u \in V_H$ with two parallel paths $u_{\text{in}}u_1^t u_2^t \cdots u_k^t u_{\text{out}}$ and $u_{\text{in}}u_1^b u_2^b \cdots u_k^b u_{\text{out}}$ (Figure 3b).

Lemma 16. *Given directed graphs G, H, let G', H' be constructed as above. If there exists a k-block conjugacy $\Phi'_\infty : X_{G'} \to X_{H'}$, then for all $v \in V_G$ there exists $u \in V_H$ such that $\Phi'(v_{in}v_1 \cdots v_{k-1}) = u_{in}$.*

Proof. Suppose for a contradiction that Φ'_∞ is a k-block code such that for some $v \in V_G$ we have $\Phi'(v_{\text{in}}v_1 \cdots v_{k-1}) \neq u_{\text{in}}$ for all $u \in V_H$. We break the argument into two cases.

First, suppose $\Phi'(v_{\text{in}}v_1 \cdots v_{k-1}) = u_i^t$ for $i \leq k$. (The case u_i^b is identical.) Since the shift map commutes with sliding block codes, we must have $\Phi'(v_1 \cdots v_{k-1}v_k^t) = \Phi'(v_1 \cdots v_{k-1}v_k^b) = z$ where $z \in \{u_{i+1}^t, u_{\text{out}}\}$. Picking any edge $(v, \hat{v}) \in E_G$ and continuing to slide the block window, we must have $\Phi'(\hat{v}_{\text{in}}\hat{v}_1 \cdots \hat{v}_{k-1}) \in \{\hat{u}_i^t, \hat{u}_i^b\}$ for some $\hat{u} \in V_H$. Without loss of generality, assume $\Phi'(\hat{v}_{\text{in}}\hat{v}_1 \cdots \hat{v}_{k-1}) = \hat{u}_i^t$. Furthermore, since there is only one word in H' between u_i^t and $\hat{u}_i^t$ of proper length but two words in G' between $\hat{v}_{\text{in}}$ and v_{in}, we have

$$\Phi'(v_{\text{in}} \cdots v_k^t v_{\text{out}}\hat{v}_{\text{in}} \cdots \hat{v}_{k-1}) = u_i^t \cdots u_k^t u_{\text{out}}\hat{u}_{\text{in}}\hat{u}_1^t \cdots \hat{u}_i^t = \Phi'(v_{\text{in}} \cdots v_k^b v_{\text{out}}\hat{v}_{\text{in}} \cdots \hat{v}_{k-1}).$$

That is, Φ' collapses a diamond, so by Lemma 4, Φ' is not a conjugacy.

Second, suppose $\Phi'(v_{\text{in}}v_1 \cdots v_{k-1}) = u_{\text{out}}$. Pick any edge $(\hat{v}, v) \in E_G$. Then without loss of generality, $\Phi'(\hat{v}_{\text{out}}v_{\text{in}} \cdots v_{k-2}) = u_k^t$. Continuing to slide the block window, we have $\Phi'(\hat{v}_{\text{in}} \cdots \hat{v}_{k-1}) = \hat{u}_{\text{out}}$ for some $\hat{u} \in V_H$. Again, there are two words in G' between $\hat{v}_{\text{in}}$ and v_{in} but only one word in H' between $\hat{u}_{\text{out}}$ and u_{out} which passes through u_k^t. Thus, we have

$$\Phi'(\hat{v}_{\text{in}} \cdots \hat{v}_k^t \hat{v}_{\text{out}}v_{\text{in}} \cdots v_{k-1}) = \hat{u}_{\text{out}}u_{\text{in}} \cdots u_k^t u_{\text{out}} = \Phi'(\hat{v}_{\text{in}} \cdots \hat{v}_k^b \hat{v}_{\text{out}}v_{\text{in}} \cdots v_{k-1}),$$

so Φ' again collapses a diamond, and by Lemma 4, Φ' is not a conjugacy. $\square$

We now show that graphs G, H admit a 1-block conjugacy if and only if the graphs G', H' constructed as above admit a k-block conjugacy. To do this, we first introduce a natural operation on shift spaces, which "stretches" each point by a factor N. Given alphabet $\mathcal{A}$, and any point $p = \cdots v.v'v'' \cdots \in \mathcal{A}^{\mathbb{Z}}$, we write $p^{\langle N \rangle} = \cdots vv \cdots v.v'v' \cdots v'v''v'' \cdots v'' \cdots$ to be the point p with each symbol repeated N times. Given a shift X over alphabet $\mathcal{A}$, we define the shift space $X^{\langle N \rangle} = \{\sigma^i(p^{\langle N \rangle}) : p \in X, i \in \mathbb{Z}\}$ where σ is the shift map. In particular, $X^{\langle N \rangle}$ contains all shifts of the Nth expansion of points in X. While in general $X^{\langle N \rangle}$ is not a vertex shift when $N > 1$, it is still structured enough that the following lemma is immediate.

Lemma 17. *Given shifts X, Y, there exists a 1-block conjugacy $\Phi_\infty : X \to Y$ if and only if there exists a 1-block conjugacy $\Phi_\infty^{\langle N \rangle} : X^{\langle N \rangle} \to Y^{\langle N \rangle}$, where $\Phi = \Phi^{\langle N \rangle}$ as block maps.*

To make use of this definition and lemma, we will project points in $X_{G'}, X_{H'}$ to $X_G^{\langle k+2 \rangle}, X_H^{\langle k+2 \rangle}$ by simply erasing the subscript and superscript information. Formally, we define the 1-block map $\Psi^G : V_{G'} \to V_G$ by $\Psi^G(u) = v$ for $u \in \{v_{\text{in}}, v_1, \ldots, v_{k-1}, v_k^t, v_k^b, v_{\text{out}}\}$, and let $\pi^G = \Psi^G_\infty : X_{G'} \to X_G^{\langle k+2 \rangle}$. We define Ψ^H, π^H similarly. Letting $S_p^G := \left(\pi^G\right)^{-1}(p) \subseteq X_{G'}$, we have that $\{S_p^G : p \in X_G^{\langle k+2 \rangle}\}$ is a partition of the points in $X_{G'}$. (Similarly for S_q^H and $X_{H'}$.)

Theorem 18. *Given graphs G, H, construct G', H' as above. Then there exists a 1-block conjugacy $\Phi_\infty : X_G \to X_H$ if and only if there exists a k-block conjugacy $\Phi' : X_{G'} \to X_{H'}$.*

Proof. ($\Rightarrow$) Suppose there exists a 1-block conjugacy $\Phi_\infty : X_G \to X_H$. By Lemma 17, there is a 1-block conjugacy $\Phi_\infty^{\langle k+2 \rangle} : X_G^{\langle k+2 \rangle} \to X_H^{\langle k+2 \rangle}$. Define the k-block code $\Phi'_\infty : X_{G'} \to X_{H'}$ with no memory by

- $\Phi'(v_{\text{in}} \cdots) = \Phi(v)_{\text{in}}$
- $\Phi'(v_i \cdots v_k^t \cdots) = \Phi(v)_i^t$, $i \in \{1, \ldots, k\}$
- $\Phi'(v_i \cdots v_k^b \cdots) = \Phi(v)_i^b$, $i \in \{1, \ldots, k\}$
- $\Phi'(v_{\text{out}} \cdots) = \Phi(v)_{\text{out}}$

To show that Φ'_∞ is a bijection, we will show that for any $p \in X_G^{\langle k+2 \rangle}$ and $q \in X_H^{\langle k+2 \rangle}$ with $\Phi_\infty^{\langle k+2 \rangle}(p) = q$, the map $\Phi'_\infty : S_p^G \to S_q^H$ is a bijection. The result then follows because $\Phi_\infty^{\langle k+2 \rangle}$ is a bijection between $X_G^{\langle k+2 \rangle}$ and $X_H^{\langle k+2 \rangle}$, and the sets $\{S_p^G : p \in X_G^{\langle k+2 \rangle}\}, \{S_q^H : q \in X_H^{\langle k+2 \rangle}\}$ partition $X_{G'}, X_{H'}$.

We first claim that $\Phi'_\infty(S_p^G) \subseteq S_q^H$, which is to say, for every $p' \in X_{G'}$ such that $\pi^G(p') = p$, we have $\pi^H(\Phi'_\infty(p')) = q$. To see this, note that by construction of Φ', for all $p' \in X_{G'}$ and all $i \in \mathbb{Z}$, we have $\Psi^H(\Phi'_\infty(p')_i) = \Phi(\Psi^G(p'_i)) = \Phi^{\langle k+2 \rangle}(\Psi^G(p'_i))$. The condition $\pi^G(p') = \Psi^G_\infty(p') = p$ implies $\Psi^G(p'_i) = p_i$ for all $i \in \mathbb{Z}$. Combining the above with the observation that $\Phi^{\langle k+2 \rangle}(p_i) = q_i$ gives $\Psi^H(\Phi'_\infty(p')_i) = q_i$, as desired.

We now show $\Phi'_\infty : S_p^G \to S_q^H$ is a bijection. Note that, by definition, any point in S_p^G and S_q^H can be uniquely identified with a point in $\{b, t\}^{\mathbb{Z}}$ by reading in order the superscripts of each v_k or u_k in the point. The map Φ'_∞, together with this identification, induces a map $\iota : \{b, t\}^{\mathbb{Z}} \to \{b, t\}^{\mathbb{Z}}$. By construction of Φ' above, ι is the identity map. Thus Φ'_∞ is a bijection from S_p^G to S_q^H.

$$\begin{tikzcd} X_{G'} \arrow[r, "\Phi'_\infty"] \arrow[d, "\pi^G=\Psi^G_\infty"'] & X_{H'} \arrow[d, "\pi^H=\Psi^H_\infty"] \\ X_G^{\langle k+2\rangle} \arrow[r, "\Phi_\infty^{\langle k+2\rangle}"] & X_H^{\langle k+2\rangle} \end{tikzcd} \qquad \begin{tikzcd} p' \arrow[r, mapsto, "\Phi'_\infty"] \arrow[d, mapsto, "\pi^G"'] & q' \arrow[d, mapsto, "\pi^H"] \\ p \arrow[r, mapsto, "\Phi_\infty^{\langle k+2\rangle}"] & q \end{tikzcd}$$

Figure 4: Given Φ' or Φ, one can construct the other such that this diagram commutes.

($\Leftarrow$) Suppose there exists a k-block conjugacy $\Phi'_\infty : X_{G'} \to X_{H'}$. By Lemma 16, for every $v \in V_G$, there exists $u \in V_H$ such that $\Phi'(v_{\text{in}} \cdots) = u_{\text{in}}$. Define the 1-block code $\Phi_\infty : X_G \to X_H$ by $\Phi(v) = \Psi^H(\Phi'(v_{\text{in}} v_1 \cdots v_{k-1}))$. We claim Φ_∞ is a conjugacy. To show this, we instead will show $\Phi_\infty^{\langle k+2\rangle}$ defined by the same block map is a conjugacy. To see $\Phi_\infty^{\langle k+2\rangle}$ is surjective, consider any $q \in X_H^{\langle k+2\rangle}$. Picking any $q' \in S_q^H$, set $p' = \Phi'^{-1}_\infty(q')$ and $p = \pi^G(p')$. We will now show $\Phi_\infty^{\langle k+2\rangle}(p) = q$, so the diagram in Figure 4 is commutative and Φ_∞ is surjective. For all $i \in \mathbb{Z}$,

$$\Phi(p_i) = \Psi^H(\Phi'((p_i)_{\text{in}} \cdots (p_i)_{k-1})) = \Psi^H((q_i)_{\text{in}}) = q_i. \tag{1}$$

To see $\Phi_\infty^{\langle k+2\rangle}$ is injective, consider distinct $p_1, p_2 \in X_G^{\langle k+2\rangle}$. Then $S_{p_1}^G \neq S_{p_2}^G$. Since Φ'_∞ is a conjugacy, $\Phi'_\infty(S_{p_1}^G) \cap \Phi'_\infty(S_{p_2}^G) = \emptyset$. By Lemma 16, there exist $q_1, q_2 \in X_H^{\langle k+2\rangle}$ such that $S_{q_1}^H = \Phi'_\infty(S_{p_1}^G)$ and $S_{q_2}^H = \Phi'_\infty(S_{p_2}^G)$. By the construction of Φ (and shown in (1)), $\Phi_\infty^{\langle k+2\rangle}(p_1) = q_1 \neq q_2 = \Phi_\infty^{\langle k+2\rangle}(p_2)$, so $\Phi_\infty^{\langle k+2\rangle}$ is injective. $\square$

The construction in Theorem 18 gives a polynomial-time reduction from 1-BC to k-BC for all k. (The same construction could plausibly give a reduction from m-BC to ℓ-BC where $\ell = (m-1)(k+2) + k$, though if true the proof would be much more involved.) Combining this reduction with Theorem 15 therefore gives GI-hardness for all k.

Corollary 19. *k-BC is GI-hard for all k.*

5 Reducing Representation Size

Thus far we have addressed two problems. We first gave an efficient algorithm, given directed graphs G, H and k-block map Φ, to verify whether $\Phi_\infty : X_G \to X_H$ is a conjugacy. We then showed that the problem of deciding whether X_G and X_H are conjugate via a k-block map, given only G and H, is GI-hard. We now address a problem given only G and an integer ℓ: whether we can find a k-block code which reduces the size of G by ℓ vertices while preserving conjugacy.

Definition 20. Given a directed graph G and integer ℓ, the *k-Block Reduction Problem*, denoted k-BR, is to decide if there exists a directed graph H with $|V_H| = |V_G| - \ell$ such that the vertex shifts X_G and X_H are conjugate via a k-block code.

We will show this problem is NP-complete for the case $k = 1$, by modifying the hardness proof of the State Amalgamation Problem (SAP), which asks if ℓ consecutive amalgamations can be performed on a graph G [5]. The proof that SAP is NP-hard shows that the set of graphs satisfying a certain structure property is closed under the amalgamation operation. This structure is then leveraged to encode an NP-complete

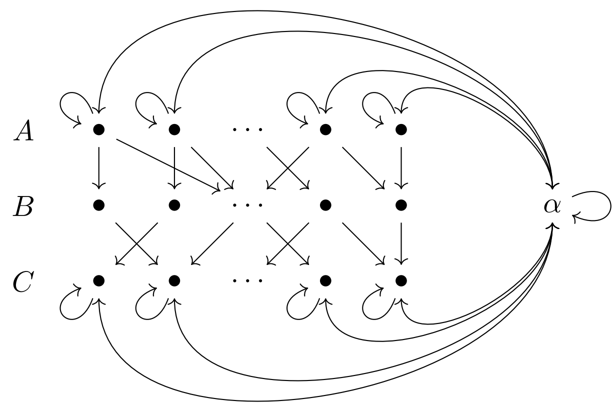


Figure 5: A graph which satisfies the structure property

problem (Hitting Set). While 1-block codes are more general than sequences of amalgamations (Figure 1), we find that, surprisingly, the same set of graphs is also closed under 1-block conjugacy. In fact, the rest of the construction of [5] suffices as well, though much of the argument needs to be strengthened to the general 1-block case.

We begin by recalling the structure property.

Definition 21 ([5])**.** A directed graph G satisfies the *structure property* if it is essential and there exists a partition $\{\{\alpha\}, A, B, C\}$ of V_G such that the following four conditions hold.

1. $N^+(\alpha) = N^-(\alpha) = \{\alpha\} \cup A \cup C$.
2. For each $a \in A$, $N^-(a) = \{a, \alpha\}$ and $\{a, \alpha\} \subseteq N^+(a) \subseteq \{a, \alpha\} \cup B$.
3. For each $c \in C$, $N^+(c) = \{c, \alpha\}$ and $\{c, \alpha\} \subseteq N^-(c) \subseteq \{c, \alpha\} \cup B$.
4. For each $b \in B$, $N^-(b) \subseteq A$ and $N^+(b) \subseteq C$.

See Figure 5 for an example. We now show that the structure property is preserved under 1-block conjugacy.

Lemma 22. *Let G be a graph with the structure property having $\{\{\alpha\}, A, B, C\}$ as the partition of V_G, and let $\Phi_\infty : X_G \to X_H$ be a 1-block conjugacy. Then $\Phi(v_1) = \Phi(v_2)$ implies $v_1 = v_2$ or $v_1, v_2 \in B$. In particular, H also satisfies the structure property with vertex partition $\{\{\Phi(\alpha)\}, \Phi(A), \Phi(B), \Phi(C)\}$.*

Proof. First note that if $\Phi : X_G \to X_H$ is a 1-block conjugacy from a graph G with vertex partition $\{\{\alpha\}, A, B, C\}$ such that $\Phi(v_1) = \Phi(v_2)$ implies $v_1 = v_2$ or $v_1, v_2 \in B$, then the fact that H satisfies the structure property with vertex partition $\{\{\Phi(\alpha)\}, \Phi(A), \Phi(B), \Phi(C)\}$ follows immediately. Now suppose for a contradiction that $v_1 \neq v_2 \in V_G$ and $\Phi(v_1) = \Phi(v_2)$; we proceed in cases.
Case 1: $v_1, v_2 \in \{\alpha\} \cup A \cup C$. Then $\Phi_\infty(v_1^\infty) = \Phi_\infty(v_2^\infty)$ and Φ_∞ is not a conjugacy.

Case 2: $v_1 = \alpha, v_2 \in B$. Let $a \in A, c \in C$ be such that av_2c is a word in X_G. (Such a, c exist as G is essential.) Then $\Phi_\infty((av_2c\alpha)^\infty) = \Phi_\infty((a\alpha c\alpha)^\infty)$ and Φ_∞ is not a conjugacy.
Case 3a: $v_1 \in A, v_2 \in B, (v_1, v_2) \notin E_G$. Let $a \in A$ be such that $(a, v_2) \in E_G$. Note that $a \neq v_1$. Consider the point

$$p = (\Phi(a)\Phi(v_2)\Phi(\alpha))^\infty = (\Phi(a)\Phi(v_1)\Phi(\alpha))^\infty$$

in X_H of period 3. Due to G having the structure property and our assumption that Φ_∞ is a 1-block conjugacy, the preimage of p must be defined by a 3-cycle whose vertices are contained in $\{\alpha\} \cup A \cup C$. In particular, the preimage must trace a self-loop, so we know $\Phi(a) = \Phi(\alpha)$ or $\Phi(a) = \Phi(v_1)$ or $\Phi(v_1) = \Phi(\alpha)$. Since we know Φ is injective on $\{\alpha\} \cup A \cup C$ by Case 1, none of these are possible.
Case 3b: $v_1 \in A, v_2 \in B, (v_1, v_2) \in E_G$. Let $c \in C$ be such that $(v_2, c) \in E_G$. Consider the point

$$p = (\Phi(v_2)\Phi(c)\Phi(\alpha))^\infty = (\Phi(v_1)\Phi(c)\Phi(\alpha))^\infty$$

in X_H of period 3. Again by the requirement that the preimage of p traces a self-loop, we know $\Phi(v_1) = \Phi(c)$ or $\Phi(v_1) = \Phi(\alpha)$ or $\Phi(\alpha) = \Phi(c)$. However, all of these situations violate the injectivity of Φ on $\{\alpha\} \cup A \cup C$.
Case 4: $v_1 \in C, v_2 \in B$. This is identical to Case 3 where the edges in the graph have been reversed. □

As in [5], we will need a "weight widget" which acts as a weighted switch, using the following notation. Let v be a vertex with $N^-(v) = D$ and $N^+(v) = E$. We will write $v : [D, E]$ in this situation, and as a slight abuse of notation, we will drop the curly brackets if E or D is a singleton and write $v : [u, E]$. Additionally, we extend this notation to sets S of vertices, so that $S : [D, E]$ means $N^-(S) = D$ and $N^+(S) = E$, where $N^-(S) = \bigcup_{v \in S} N^-(v)$ and similarly for N^+.

Definition 23 ([5])**.** Let G satisfy the structure property with $V_G = \{\alpha\} \cup A \cup B \cup C$, and let $K > 0$ be a fixed even integer. Then for nonempty subsets $A_* \subseteq A, C_* \subseteq C$, the *weight widget* $w = \mathsf{weight}_K[A_*, C_*]$ is the following collection of vertices.

- $A_w = \{a_1^w, \ldots, a_{K/2}^w\}$
- $B_w = \{b_1^w \ldots, b_K^w\}$
- $C_w = \{c_1^w, \ldots, c_{K/2}^w\}$

where $A_w \cap A_* = \emptyset = C_w \cap C_*$, and for each $i \in \{1, \ldots, K/2\}$ we have

- $b_{2i-1} : [A_* \cup \{a_1^w, \ldots, a_{i-1}^w\}, c_i^w]$
- $b_{2i} : [a_i^w, C_* \cup \{c_1^w, \ldots, c_i^w\}]$.

Moreover, we require these to be the only out-neighbors of A_w in B, i.e., $B \cap N^+(a_i^w) \subseteq B_w$ for all $a_i^w \in A_w$, and similarly for the in-neighbors of C_w. For a given 1-block conjugacy Φ_∞, letting $S = \Phi^{-1}(\Phi(b_1^w)) \setminus B_w = \{b \in B : \Phi(b) = \Phi(b_1^w)\} \setminus B_w$, we say w is *activated* if $S : [A_*, C_*]$.

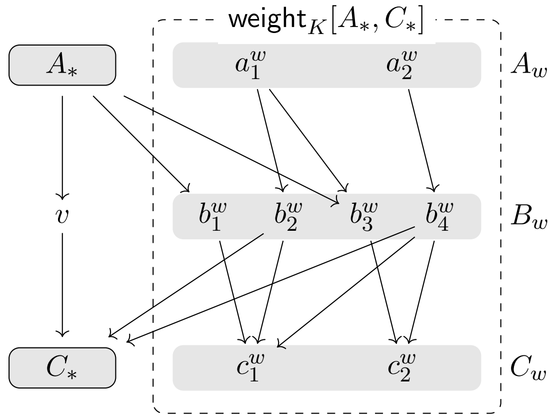


Figure 6: The weight widget $\mathsf{weight}_K[A_*, C_*]$ with $K = 4$.

In other words, when the vertex b_j is added, if j is odd we add edges from A_* and $a^w_1, \ldots, a^w_{(j-1)/2}$, and an edge to $c^w_{(j+1)/2}$, and when j is even, we add an edge from $a^w_{j/2}$ and edges to C_* and $c^w_1, \ldots, c^w_{j/2}$. See Figure 6 for an example. The term "activate" comes from the following fact, which we show below in Lemma 25(1): if $S = \{v\}$ is a singleton, then the construction of the weight widget allows v to amalgamate with $b^w_1, \ldots, b^w_K$ in order, thus amalgamating $\{v\} \cup B_w$ into a single vertex. For example, the vertex v in Figure 6 can activate the weight widget shown. The next two lemmas show that these amalgamations cannot be performed if the widget is not activated.

Lemma 24. *Let $w = \mathsf{weight}_K[A_*, C_*]$ be a weight widget in G. If $\Phi_\infty : X_G \to X_H$ is a 1-block conjugacy between graphs with the structure property, then for any $v \in V_G$ and $\ell > 1$, the statement $b^w_\ell \in \Phi^{-1}(\Phi(v))$ implies $b^w_{\ell-1} \in \Phi^{-1}(\Phi(v))$ or $|\Phi^{-1}(\Phi(v))| = 1$.*

Proof. By contrapositive, suppose $|\Phi^{-1}(\Phi(v))| > 1$ and there exists $b^w_\ell \in \Phi^{-1}(\Phi(v))$ such that $b^w_{\ell-1} \notin \Phi^{-1}(v)$. Without loss of generality, let ℓ be the largest such subscript. We have two cases; in each we will find a word in X_H with no Φ-preimage in X_G.

Case 1: ℓ is even. We will show there exists $a \in A \setminus \{a^w_{\ell/2}, \ldots, a^w_{K/2}\}$ such that $\Phi(a)\Phi(v)\Phi(c^w_{\ell/2})$ is a word in X_H but has no preimage in X_G. To show the word has no preimage, first recall from Lemma 22 that $\Phi^{-1}(\Phi(a)) = a$ and $\Phi^{-1}(\Phi(c)) = c$ for any $a \in A$ and $c \in C$, and $c = c^w_{\ell/2}$ in particular. A preimage of this word therefore takes the form $abc^w_{\ell/2}$ for some $b \in B$, but the only such word in X_G has $b = b^w_{\ell-1} \notin \Phi^{-1}(\Phi(v))$.

The word $\Phi(v)\Phi(c^w_{\ell/2})$ is in X_H, as $(b^w_\ell, c^w_{\ell/2})$ is an edge in G by definition of the weight widget, and $b^w_\ell \in \Phi^{-1}(\Phi(v))$ by assumption. We next claim that there exists $a \in A \setminus \{a^w_{\ell/2}, \ldots, a^w_{K/2}\}$ such that $\Phi(a) \in N^-(\Phi(v))$. If $\Phi^{-1}(\Phi(v)) \subseteq \{b^w_2, b^w_4, \ldots, b^w_K\}$, recall that $|\Phi^{-1}(\Phi(v))| > 1$ and ℓ is the largest such subscript such that there exists $b^w_{2j} \in \Phi^{-1}(\Phi(v))$ with $2j < \ell$; now a^w_j satisfies our claim as by construction (a^w_j, b^w_{2j}) is an edge of G. Otherwise, either $v = b^w_j$ for j odd, or $v \notin B_w$. In the former, we have $\emptyset \neq A_* \subseteq N^{-1}(v)$. In the latter, we have $N^{-1}(v) \neq \emptyset$ since G is essential, and $N^{-1}(v) \cap A_w = \emptyset$ by construction.

Case 2: ℓ is odd. Using an argument symmetric to Case 1, there exists $c \in C \setminus \{\Phi(c^w_{(\ell+1)/2}), \ldots, \Phi(c^w_{K/2})\}$ such that the word $\Phi(a^w_{(\ell-1)/2})\Phi(v)\Phi(c)$ is a word in X_H without a preimage in X_G. $\square$

Lemma 25. *Suppose $w = \mathsf{weight}_K[A_*, C_*]$ is a weight widget in G. Then*

1. *Suppose V_G contains $v : [A_*, C_*]$. Define $\Phi_\infty : X_G \to X_H$ by*

$$\Phi(u) = \begin{cases} v, & \text{if } u \in \{v\} \cup B_w \\ u, & \text{else} \end{cases}$$

where H is the minimal graph induced by G and Φ. Then Φ is a 1-block conjugacy with $|V_H| = |V_G| - K$.

2. *If $w = \mathsf{weight}_K[A_*, C_*]$ is not activated and $\Phi_\infty : X_G \to X_H$ is a 1-block conjugacy, then $\Phi^{-1}(\Phi(b_i^w))$ is a singleton for every b_i^w with $i > 1$.*

Proof. (1) Note that by the construction of the weight widget, $\{v : [A_*, C_*]\} \cup B_w$ can be amalgamated sequentially for a total of K amalgamations.

(2) Suppose $b_1^w \in \Phi^{-1}(v)$ for some $v \in V_H$ and consider $V = \Phi^{-1}(v) \setminus B_w$. By Lemma 24 it suffices to show $b_2^w \notin \Phi^{-1}(v)$. By definition of w not being activated, we have two cases.

Case 1: $N^-(V) \neq A_*$. If there is some $a \in N^-(V) \setminus A_*$, then $\Phi(a)v\Phi(c_1^w)$ is a word in X_H. Since there is no vertex in G connecting a with c_1^w, the word has no preimage in X_G and Φ_∞ is not a conjugacy. Otherwise, there is some $a \in A_* \setminus N^-(V)$. By contrapositive, suppose $b_2^w \in \Phi^{-1}(v)$. Picking any $c \in C_*$, we have $\Phi(a)v\Phi(c)$ is a word in X_H. Since there is no vertex in $\Phi^{-1}(v)$ connecting a with c, the word has no preimage and Φ_∞ is not a conjugacy.

Case 2: $N^+(V) \neq C_*$. By contrapositive, suppose $b_2^w \in \Phi^{-1}(v)$. If there is some $c \in N^+(V) \setminus C_*$, then $\Phi(a_1^w)v\Phi(c)$ is a word in X_H. Since there is no vertex in G connecting a_1^w with c, the word has no preimage and Φ_∞ is not a conjugacy. Otherwise, there is some $c \in C_* \setminus N^+(V)$. Considering any $a \in N^-(V)$, we have $\Phi(a)v\Phi(c)$ is a word in X_H. Since there is no vertex in $\Phi^{-1}(v)$ connecting a with c, the word has no preimage and Φ_∞ is not a conjugacy. $\square$

We now define the Hitting Set problem, which is NP-complete [9], and state a lemma which we will need in the proof.

Definition 26. Let $\mathcal{Z} = \{S_1, \ldots, S_m\}$ be a collection of sets with $\bigcup_i S_i = U$. Given a subset $S \subseteq U$, we define its *hit set* as $\mathrm{hit}(S) = \{S_i : S \cap S_i \neq \emptyset\}$. Given $\mathcal{Z}, U$, and an integer t, the *hitting set problem*, denoted $\mathsf{HittingSet}$, is to decide whether there is a set H of cardinality t such that $\mathrm{hit}(H) = \mathcal{Z}$. We will also overload this notation, and write $\mathrm{hit}(s)$ to mean $\mathrm{hit}(\{s\})$ for $s \in U$.

Lemma 27 ([5])**.** *Let $(\mathcal{Z}, U, t)$ be an instance of* $\mathsf{HittingSet}$*. Suppose for some $t \leq |\mathcal{Z}|$ there is no H with $|H| \leq t$ and $hit(H) = \mathcal{Z}$. Then for all $H \subseteq U$, $|hit(H)| - |H| < |\mathcal{Z}| - t$.*

We now show that $\mathsf{1\text{-}BR}$ is NP-complete, by reduction from $\mathsf{HittingSet}$. Given the lemmas developed above, the result essentially follows from the argument in [5], with minor modifications for the 1-block case; for completeness, we give the full proof.

Theorem 28. $\mathsf{1\text{-}BR}$ *is* NP*-complete.*

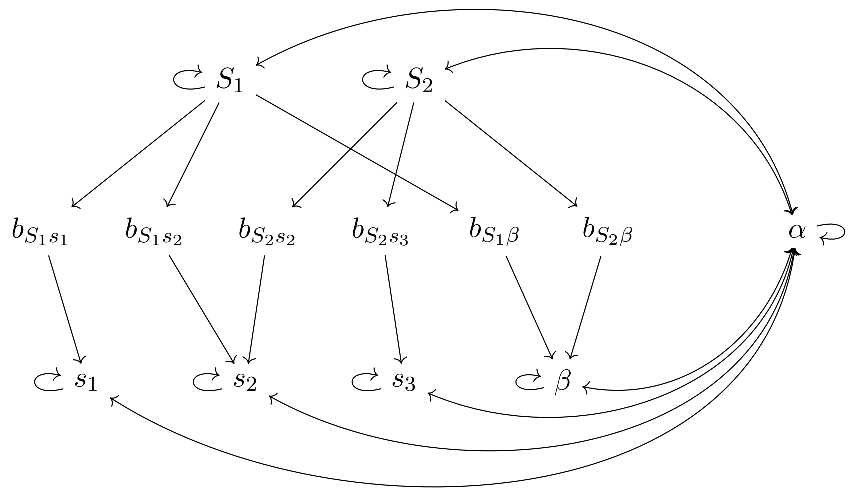


Figure 7: The graph constructed in Theorem 28 for the HittingSet instance with $\mathcal{Z} = \{\{u_1, u_2\}, \{u_2, u_3\}\}$, without any weight widgets attached.

Proof. First we show 1-BR is in NP. Given a vertex shift X_G and $\Phi : V_G \to \{1, 2, \ldots, |V_G| - n\}$ from a proposed 1-block conjugacy Φ_∞, we construct the minimal image graph G' such that $\Phi_\infty : X_G \to X_{G'}$ is well-defined. In particular, $V_{G'} = \{\Phi^{-1}(u) : u \in V_G\}$ and $E_{G'} = \{(\Phi^{-1}(v_1), \Phi^{-1}(v_2)) : (v_1, v_2) \in E_G\}$. By Corollary 13, we can determine if Φ_∞ is a conjugacy in $O(|V_G|^4)$ time.

To show hardness, we reduce from HittingSet; let $\mathcal{Z} = \{S_1, \ldots, S_m\}$ be the collection of sets and t the given integer. Defining $n = |U|$ for $U = \bigcup_i S_i$, we set the parameter $K = 6mn$ for the weight widgets. Then, as in [5], we build the following graph $G = (V_G, E_G)$ with the structure property $V_G = A \cup B \cup C \cup \{\alpha\}$.

1. Start with $A = \mathcal{Z}, B = \emptyset, C = U \cup \{\beta\}$, where β is a new vertex.
2. For each $S_i \in \mathcal{Z}$, and $c \in S_i \cup \{\beta\}$, add $b_{S_ic} : [S_i, c]$. That is, add the vertex b_{S_is} to B and the path $S_i \to b_{S_is} \to s$ for every $s \in S_i$, and add the vertex $b_{S_i\beta}$ to B and path $S_i \to b_{S_i\beta} \to \beta$.
3. For each (s, S_i) with $s \in S_i$, add the weight widget $w = \mathsf{weight}_K[S_i, \{s, \beta\}] = (A_w, B_w, C_w)$. Note that A_w will added to A, B_w to B, and C_w to C.
4. For each $s \in U$, add the weight widget $\mathsf{weight}_K[\mathrm{hit}(s), \{s\}]$.
5. Finally, add the vertex α and the necessary edges for G to have the structure property, i.e., add the edges $\{(a, \alpha), (\alpha, a) : a \in A\} \cup \{(b, \alpha), (\alpha, b) : b \in B\} \cup \{(v, v) : v \in A \cup B \cup \{\alpha\}\}$.

Summarizing, if W is the collection of weight widgets added above, $A = \mathcal{Z} \cup \bigcup_{w \in W} A_w$, $B = \{b_{S_is} : S_i \in \mathcal{Z}, s \in S_i\} \cup \{b_{S_i\beta} : S_i \in \mathcal{Z}\} \cup \bigcup_{w \in W} B_w$, and $C = U \cup \{\beta\} \cup \bigcup_{w \in W} C_w$. See Figure 7 for an example with $\mathcal{Z} = \{\{u_1, u_2\}, \{u_2, u_3\}\}$ where only steps (1), (2), and (5) have been performed.

We will show there is a hitting set of size t if and only if there is a 1-block conjugacy $\Phi_\infty : X_G \to X_{G'}$ such that $|V_{G'}| \leq |V_G| - (m+n-t)K$. The idea behind the reduction is that s can either choose to be in the hitting set by combining some $b_{S_i s}$ with the appropriate $b_{S_i \beta}$ to activate some of the $\mathsf{weight}_K[S_i, \{s, \beta\}]$, or choose not to be in the hitting set by combining all $b_{S_i s}$ for $S_i \in \mathrm{hit}(s)$ to activate $\mathsf{weight}_K[\mathrm{hit}(s), \{s\}]$. We will be able to activate $|\mathrm{hit}(H)| + |U \setminus H| = m + (n-t)$ weight widgets if there is a hitting set of size t and strictly fewer if no such set exists. Finally, we show that our choice of K is larger than any reduction in the number of vertices not caused by activating weight widgets.

First, suppose there is a hitting set H for $\mathcal{Z}$ of size t. We will give a sequence of $(m+n-t)K$ consecutive amalgamations, which together constitute a 1-block reducing the number of vertices by $(m+n-t)K$. For each $S_i \in \mathcal{Z}$, pick some $s \in H$ such that $S_i \in \mathrm{hit}(s)$. Amalgamating $b_{S_i s}$ with $b_{S_i \beta}$, we obtain a vertex $v : [S_i, \{s, \beta\}]$, which by Lemma 25 can activate the weight widget $w = \mathsf{weight}_K[S_i, \{s, \beta\}]$ for K additional amalgamations. Performing the above steps for each $S_i \in \mathcal{Z}$ gives a total of $m(K+1) \geq mK$ consecutive amalgamations.

As the above amalgamations only affected the vertices in B associated with H, next consider any $s \in U \setminus H$. We can amalgamate the vertices $\{b_{S_i s} : S_i \in \mathrm{hit}(s)\}$ in any order to form $b_{\mathrm{hit}(s)s} : [\mathrm{hit}(s), \{s\}]$ which by Lemma 25 can then activate $\mathsf{weight}_K[\mathrm{hit}(s), \{s\}]$, for an additional K amalgamations. For $U \setminus H$, we thus have at least $(n-t)K$ total amalgamations. Putting all of these steps together, we can perform at least $mK + (n-t)K = (m+n-t)K$ consecutive amalgamations, so there is a 1-block conjugacy $\Phi_\infty : X_G \to X_{G'}$ such that $|V_G| \geq |V_{G'}| - (m+n-t)K$.

Next suppose there is no hitting set H of size t. Let $\Phi : X_G \to X_{G'}$ be a 1-block conjugacy such that $N = |V_G| - |V_{G'}|$ is as large as possible. Define

$$\begin{aligned} \overline{H} &= \{s \in U : \mathsf{weight}_K[\mathrm{hit}(s), \{s\}] \text{ is activated}\}, \\ F &= \{S_i : \mathsf{weight}_K[S_i, \{s, \beta\}] \text{ is activated for some } s \in S_i\}, \\ H &= U \setminus \overline{H}. \end{aligned}$$

Note that there is a single path in G from S_i to s, through the vertex $b_{S_i s}$, which is required to activate both $\mathsf{weight}_K[\mathrm{hit}(s), \{s\}]$ and $\mathsf{weight}_K[S_i, \{s, \beta\}]$. Thus for every $b_{S_i s}$ we have that if $S_i \in F$, then $s \in H$. That is,

$$F \subseteq \{S_i : s \in H \text{ for some } b_{S_i s}\}. \tag{2}$$

We now count how much smaller $|V_{G'}|$ could be than $|V_G|$. From Lemma 22, Φ is injective on $A \cup C \cup \{\alpha\}$, and thus the only reduction could come from combining vertices in B. By construction, each activated widget can lead to reducing the number of vertices by at most K. Let $B_{\text{non-weight}} = \{b_{S_i s} : S_i \in \mathcal{Z}, s \in S_i\} \cup \{b_{S_i \beta} : S_i \in \mathcal{Z}\}$ be the vertices in B not in weight widgets. By Lemma 25, if $u \in \Phi^{-1}(v)$ with $|\Phi^{-1}(v)| > 1$ for some u not in an activated widget, then $u \in B_{\text{non-weight}} \cup \bigcup_{w \in W} w_1$. Thus V_G can be reduced by at most $(|F| + |\overline{H}|)K + |B_{\text{non-weight}}| + |W|$. Since

$$|B_{\text{non-weight}}| + |W| = (mn + m) + (mn + n) < K,$$

we have

$$\begin{aligned}
N &\le (|F| + |\overline{H}|)K + |B_{\text{non-weight}}| + |W| \\
&< (|F| + |\overline{H}|)K + K \\
&\le (|\text{hit}(H)| + (n - |H|) + 1)K && \text{(by (2) and } H = U \setminus \overline{H}) \\
&\le (m + n - t)K && \text{(by Lemma 27).}
\end{aligned}$$

□

6 Edge Shifts

Thus far we have restricted our attention to vertex shifts, rather than edge shifts, though the latter are perhaps more commonly used in the literature. For various reasons, the problems we consider are in general more appropriate for vertex shifts, as we discuss in the following section. (Vertex shifts are also motivated by applications (§ 7).) Nonetheless, we now give some results for edge shifts, for the first two problems: verifying k-block conjugacies, and testing pairs of shifts for conjugacy. (The third problem remains open.)

In the following, we will leverage our results for vertex shifts, using the standard conversion from edge shifts to vertex shifts: edges become vertices, and pairs of adjacent edges become edges [12, Proposition 2.3.9]. More formally, we recall that given edge shift X^e_G, its vertex shift representation is the shift $X_{G'}$ where $V_{G'} = E_G$ and $E_{G'} = \{(e_i, e_j) : e_i e_j \text{ is a word in } X^e_G\}$. Thus, for any edge shifts X^e_G, X^e_H, there exists a k-block conjugacy $\Phi_\infty : X^e_G \to X^e_H$ if and only if there exists a k-block conjugacy $\Phi'_\infty : X_{G'} \to X_{H'}$ between the vertex shift representations of X_G and X_H.

First, we observe that our verification algorithm for vertex shifts immediately applies to edge shifts.

Theorem 29. *Given directed multigraphs G, H and a proposed k-block code $\Phi_\infty : X^e_G \to X^e_H$, deciding if Φ_∞ is a conjugacy can be determined in $O(|E_G|^{4k})$.*

Proof. Given edge shifts X^e_G, X^e_H, we first construct their vertex shift representations $X_{G'}, X_{H'}$ as above. Letting Φ'_∞ be the corresponding block code between the vertex shifts, by Corollary 13, we can determine if Φ'_∞ is a conjugacy in $O(|V_{G'}|^{4k}) = O(|E_G|^{4k})$ time. □

We now turn to the k-block conjugacy problem for edge shifts, where we again show GI-hardness.

Definition 30. Given directed multigraphs G, H, the *k-Block Conjugacy Problem*, denoted k-BC^e, is to decide is there is a k-block conjugacy $\Phi_\infty : X^e_G \to X^e_H$ between the edge shifts X^e_G, X^e_H.

Theorem 31. *k-BC^e is GI-hard.*

Proof. We first show that 1-BC^e is GI-hard. Given directed graphs G, H with $|E_G| = |E_H|$, as in the vertex shift case, we will argue that there exists a 1-block conjugacy between the edge shifts if and only if the graphs are isomorphic. Suppose first that G, H are isomorphic. Let G', H' be the directed graphs for vertex shifts, as described above, so that $X_{G'} = X^e_G$ and $X_{H'} = X^e_H$. Since G, H are isomorphic and G', H' are created

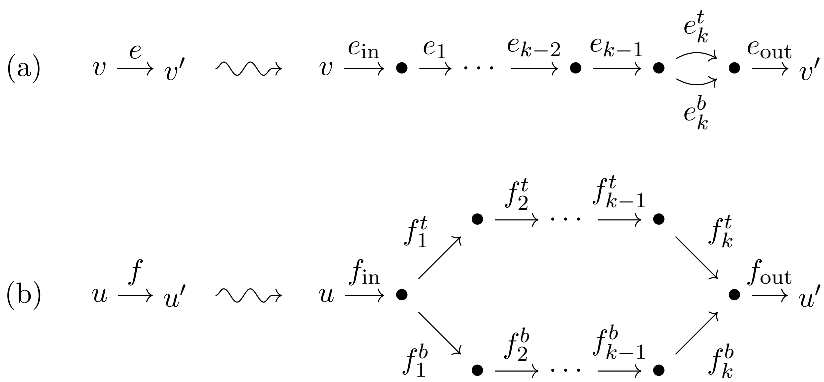


Figure 8: (a) The edge gadget for each pre-image graph. (b) The edge gadget for each image graph.

by the same (deterministic) procedure, G', H' are isomorphic. By Theorem 15, there exists a 1-block conjugacy $\Phi_\infty : X_{G'} \to X_{H'}$, so $X_{G'} = X_G^e$ is conjugate to $X_{H'} = X_H^e$ via a 1-block code.

Now suppose $\Phi_\infty : X_G^e \to X_H^e$ is a 1-block conjugacy. Noting that $\Phi : E_G \to E_H$ is a map on edges, we show that Φ can be realized as a map on V_G. To do this, it suffices to show (i) for any two edges $(v_1, v_2), (v_1, v_3)$ starting at the same vertex, $\Phi((v_1, v_2)), \Phi((v_1, v_3))$ also start at the same vertex and (ii) for any two edges $(u_1, u_2), (u_3, u_2)$ ending at the same vertex, $\Phi((u_1, u_2)), \Phi((u_3, u_2))$ also end at the same vertex. To see condition (i), consider any $(v_4, v_1) \in E_G$. As $\Phi((v_4, v_1)(v_1, v_2))$ and $\Phi((v_4, v_1)(v_1, v_3))$ must both be words in X_H^e, we must have that $\Phi((v_1, v_2)), \Phi((v_1, v_3))$ both start at the same vertex. Similarly, for condition (ii), consider any $(u_2, u_4) \in E_G$, and note that $\Phi((u_1, u_2)(u_2, u_4)), \Phi((u_3, u_2)(u_2, u_4))$ are both words in X_H^e, so $\Phi(u_1, u_2), \Phi(u_3, u_2)$ must end at the same vertex. Thus Φ can be realized as a map $\Psi : V_G \to V_H$ on vertices which is surjective and preserves the edge/non-edge relation. To show Ψ is actually a graph isomorphism, consider the inverse Φ_∞^{-1}. Since Φ_∞ is 1-block conjugacy and $|E_G| = |E_H|$, Φ_∞^{-1} is a 1-block code. Again, Φ_∞^{-1} can be realized as a surjective vertex map Ψ' which preserves the edge/non-edge relation. Since both $\Psi : V_G \to V_H$ and $\Psi' : V_H \to V_G$ are surjective maps between finite sets, we actually have Ψ, Ψ' are bijections. Thus Ψ is a graph isomorphism from G to H.

We now reduce k-BC^e to 1-BC^e, as we did with vertex shifts. Given edge shifts X_G^e, X_H^e, construct $\hat{G}, \hat{H}$ as follows. To form $\hat{G}$, substitute each edge in G with a path of length k followed by two parallel edges and a final edge (Figure 8a). Construct $\hat{H}$ by substituting each edge in H with a single edge followed by two parallel paths of length k followed by a single edge (Figure 8b). Then construct the vertex shift representations $G', H', \hat{G}', \hat{H}'$ of $G, H, \hat{G}, \hat{H}$. By construction of the edge gadget, $\hat{G}', \hat{H}'$ can be formed from G', H' by using the vertex gadget in Figure 3. Thus by Theorem 18, there exists a 1-block conjugacy $\Phi'_\infty : X_{G'} \to X_{H'}$ if and only if there exists a k-block conjugacy $\hat{\Phi}'_\infty : X_{\hat{G}'} \to X_{\hat{H}'}$. Since there exists a k-block conjugacy $\Phi_\infty : X_G^e \to X_H^e$ between edge shifts if and only if there exists a k-block conjugacy $\Phi'_\infty : X_{G'} \to X_{H'}$ between the vertex representations, there exists a 1-block conjugacy $\Phi_\infty : X_G \to X_H$ if and only if there exists a k-block conjugacy $\hat{\Phi} : X_{\hat{G}} \to X_{\hat{H}}$. Thus k-BC^e is 1-BC^e-hard and, in particular, GI-hard. $\square$

	Block size	Verif. (G, H, Φ)	Conjugacy (G, H)	Reduction (G, ℓ)
Vertex	$k = 1$	1-BV: P	1-BC: GI-hard, NP	1-BR: NP-complete
	$k > 1$	k-BV: P	k-BC: GI-hard, NP	k-BR: NP-complete ?
Edge	$k = 1$	1-BV^e: P	1-BC^e: GI-hard, NP*	1-BR^e: NP-complete ?
	$k > 1$	k-BV^e: P	k-BC^e: GI-hard, NP*	k-BR^e: NP-complete ?

Table 1: Summary of results and open questions, for vertex and edge shifts. Question marks denote conjectures, and BV refers to the verification problem (§ 3). The asterisk (*) denotes a subtlety in how edge shifts are represented: the k-block conjugacy problem is in NP when the representation size is considered to be the number of edges (i.e., a unary representation), but membership in NP is not clear when the shift is given as an adjacency matrix (i.e., a binary representation).

7 Discussion

We have addressed several variants of the conjugacy problem restricted to k-block codes, with new algorithms to verify a proposed conjugacy, and hardness results for k-block conjugacy and representation reduction via 1-block codes (Table 1). Below we discuss subtleties of input representation, followed by applications and open problems.

Representations of SFTs. When considering how to describe a subshift of finite type (SFT), three representations come to mind: a vertex shift, an edge shift, and a list of forbidden words $\mathcal{F}$. As our results pertain to vertex and edge shifts, we now discuss some nuances in these two representations, leaving lists of forbidden words to future work.

Perhaps the central advantage of edge shifts over vertex shifts is their compact representation size: a shift on n symbols can be represented in size as small as $O(\log n)$ by writing the multigraph as a integer adjacency matrix, as opposed to $\Omega(n)$ for vertex shifts. This compact representation size can have important implications on the computational complexity. In the verification problem, for example, writing down a k-block code Φ naïvely takes $\Omega(n) = \Omega(|E_G|)$ space, which can be exponential in the size of the graphs G, H. (One can improve this by encoding Φ as a integer $|V_G| \times |V_G| \times |E_H|$ tensor, specifying how many $(u, v) \in E_G$ edges map to a given $e \in E_H$, but this can still be exponential.) Thus, while our algorithm remains polynomial-time in n, it would not be for cases allowing a compact representation of Φ.

Similarly, for the conjugacy problem, we only know k-BC^e to be in NP if we consider the graphs G, H to be represented in adjacency list form, which takes $\Omega(|E_G|)$ space, rather than the typically more compact integer adjacency matrix form taking $O(|V_G| \log |E_G|)$ space, as the natural certificate is the block map Φ witnessing the conjugacy. For the matrix representation of edge shifts, membership in NP would require a certificate exponentially smaller than the naïve representation of the block map Φ.

Finally, what "size reduction" means for edge shifts depends on the choice of adjacency list or matrix above. For the adjacency list, we have that the problem of reducing the number of vertices in the graph is in NP, but it is less motivated, as the size is dominated by $|E_G|$, not $|V_G|$. On the other hand, while the adjacency matrix representation size is dominated by $|V_G|$, it is not clear whether the problem of reducing the number of vertices is in NP, for the same reason as above.

Motivation from Markov partitions. As noted in [5], variants of the conjugacy problem for vertex shifts have applications in simplifying Markov partitions, a tool to study discrete-time dynamical systems via symbolic dynamics. Briefly, a Markov partition is a collection C of regions of the phase space, satisfying certain properties, which induces a conjugacy to a vertex shift X_G where $V_G = C$, i.e., the vertices are labeled with the regions of the phase space [6, Proposition 5.3.4]. (Our discussion also applies to the weaker definition yielding only an almost 1-to-1 map [2, § 6.1].) In applications, one can encounter Markov partitions with thousands of regions, thus motivating the problem of simplifying the partition. Without additional information about the dynamical system, essentially the only way to do this while preserving the relevant geometric information is to *coarsen* the partition, by replacing sets of regions with a single region which is their union. This operation is exactly a 1-block code. Our results therefore give an efficient algorithm to test whether a proposed coarsening (1-block code) is valid (yields a conjugacy). Our results also imply that the problem of minimizing the partition size is NP-complete. (Previous work [5] only showed the latter for the case where the 1-block code was a sequence of amalgamations.)

Open problems. Our work leaves several open problems, such as those implied by Table 1: resolving the complexity of the k-block conjugacy problem, and showing NP-hardness of the size reduction problem. The complexity of deciding k-block conjugacy between edge shifts represented as integer matrices is especially interesting, as membership in NP is perhaps unlikely (see above). Regarding the k-block conjugacy problem and resolving where it lies on the spectrum between GI-complete and NP-complete, we conjecture that, similar to the induced subgraph isomorphism problem [17], it is an NP-complete problem which happens to be GI-complete when $|V_G| = |V_H|$. Beyond these questions, it would be interesting to address the complexity of k-block conjugacy between SFTs given as lists of forbidden words, and the natural variants of the problem for that input (for example, reducing the representation size of the list).

Acknowledgments

We thank Luke Meszar for valuable contributions to the early stages of this work. We also thank Mike Boyle, Josh Grochow, Doug Lind, and Brian Marcus, for several helpful conversations, references, and insights.

References

[1] Robert Berger. *The undecidability of the domino problem*. Number 66. American Mathematical Society, 1966.

[2] Jan De Vries. *Topological dynamical systems: an introduction to the dynamics of continuous mappings*. de Gruyter, 2014.

[3] Emilie Delnieppe. N-block presentations and decidability of direct conjugacy between subshifts of finite type. *arXiv preprint arXiv:1604.01721*, 2016.

[4] Jean-Guillaume Dumas, Clément Pernet, and Zhendong Wan. Efficient computation of the characteristic polynomial. In *Proceedings of the 2005 international symposium on Symbolic and algebraic computation*, pages 140–147, 2005.

[5] Rafael M. Frongillo. Optimal state amalgamation is NP-hard. *Ergodic Theory and Dynamical Systems*, 39(7):1857–1869, 2019. doi: 10.1017/etds.2017.105.

[6] John Guckenheimer and Philip Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* Springer, New York, 1983. ISBN 978-0-387-90819-9.

[7] Emmanuel Jeandel and Pascal Vanier. Hardness of conjugacy, embedding and factorization of multidimensional subshifts. *Journal of Computer and System Sciences*, 81(8):1648–1664, 2015. URL http://www.sciencedirect.com/science/article/pii/S0022000015000604.

[8] Natasa Jonoska and Brian Marcus. Minimal presentations for irreducible sofic shifts. *IEEE transactions on information theory*, 40(6):1818–1825, 1994. URL http://cat.inist.fr/?aModele=afficheN&cpsidt=3457216.

[9] Richard M. Karp. *Reducibility among combinatorial problems.* Springer, 1972.

[10] Walter Keller-Gehrig. Fast algorithms for the characteristics polynomial. *Theoretical computer science*, 36:309–317, 1985.

[11] Johannes Kobler, Uwe Schöning, and Jacobo Torán. *The Graph Isomorphism problem: Its Structural Complexity.* Berkhauser Verlag, Basel, Switzerland, 1993.

[12] Douglas Lind and Brian Marcus. *An introduction to symbolic dynamics and coding.* Cambridge University Press, 1999. URL https://books.google.com/books?hl=en&lr=&id=qSkNs3jr-DIC.

[13] Douglas Lind and Klaus Schmidt. Symbolic and algebraic dynamical systems. *Handbook of dynamical systems*, 1:765–812, 2002.

[14] Andreas Malcher. Minimizing finite automata is computationally hard. *Theoretical Computer Science*, 327(3):375 – 390, 2004. ISSN 0304-3975. doi: https://doi.org/10.1016/j.tcs.2004.03.070. URL http://www.sciencedirect.com/science/article/pii/S0304397504004864. Developments in Language Theory.

[15] Ville Salo and Ilkka Törmä. Complexity of conjugacy, factoring and embedding for countable sofic shifts of rank 2. In *International Workshop on Cellular Automata and Discrete Complex Systems*, pages 121–134. Springer, 2014.

[16] Klaus Sutner. Linear cellular automata and de bruijn automata. In Maxence Delorme and Jacques Mazoyer, editors, *Cellular Automata: A Parallel Model*, pages 303–319, Dordrecht, 1999. Springer Netherlands. ISBN 978-94-015-9153-9. doi: 10.1007/978-94-015-9153-9_12. URL https://doi.org/10.1007/978-94-015-9153-9_12.

[17] Maciej M. Sysło. The subgraph isomorphism problem for outerplanar graphs. *Theoretical Computer Science*, 17(1):91–97, 1982. ISSN 0304-3975. doi: https://doi.org/10.1016/0304-3975(82)90133-5. URL http://www.sciencedirect.com/science/article/pii/0304397582901335.

[18] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, 1972.

[19] Robert F. Williams. Classification of subshifts of finite type. *Annals of Mathematics*, 98(1):120–153, 1973. URL http://link.springer.com/content/pdf/10.1007/BFb0061747.pdf.

[20] Charalampos Zinoviadis. Hierarchy and expansiveness in 2d subshifts of finite type. In *International Conference on Language and Automata Theory and Applications*, pages 365–377. Springer, 2015.

A Algorithms

Function `IsInjective`(G, H, Φ)**:**

> **Input:** irreducible graphs G, H and a 1-block code Φ
> **Output:** true, if $\Phi_{\mathsf{c}} : \bigcup_n C_n(G) \to \bigcup_n C_n(H)$ is injective; false, otherwise
> `/* Construct the meta-graph` M `*/`
> $V_M \leftarrow V_G \times V_G$;
> $E_M \leftarrow \{((u_1, v_1), (u_2, v_2)) : \Phi(u_1) = \Phi(v_1), \Phi(u_2) = \Phi(v_2), \text{ and } (u_1, u_2), (v_1, v_2) \in E_G\}$;
> `/* Decide if` M `has a cycle passing through` (v_1, v_2) `with` $v_1 \neq v_2$ `*/`
> $\mathcal{S} \leftarrow$ `GetStronglyConnectedComponents`(M) ; `/* Tarjan's */`
> **foreach** *subgraph s in $\mathcal{S}$* **do**
>   **if** *s is a singleton* **then**
>   **continue**;
>   **end**
>   **foreach** *vertex (v_1, v_2) in s* **do**
>   **if** $v_1 \neq v_2$ **then**
>   **return** *true*;
>   **end**
>   **end**
> **end**
> **return** *false*;

Algorithm 1: Determine if Φ_{c} is injective

Function `IsConjugacyIrreducible`(G, H, Φ)**:**

> **Input:** irreducible graphs G, H and a 1-block code Φ
> **Output:** true, if Φ_∞ is a conjugacy; false, otherwise
> **if** *not* `IsInjective`(G, H, Φ) **then**
>   **return** *false*;
> **end**
> **for** $i \in \{1, \ldots, |V_G|\}$ **do**
>   **if** $tr(A(G)^i) \neq tr(A(H)^i)$ **then**
>   **return** *false*;
>   **end**
> **end**
> **return** *true*;

Algorithm 2: Determine if Φ_∞ between irreducible graphs is a conjugacy

```
Function AddSinkVertices(G, H, Φ):
    Input: reducible graphs G, H and a 1-block code Φ
    Result: (1) alters G, H by adding a new sink vertex to each graph, and
            (2) extends Φ to the new graphs so Φ∞ : X_G → X_H is a
            conjugacy if and only if the original 1-block code was a conjugacy
    /* Add the new vertex t and 't                                    */
    V_H.Add(t);
    N⁺(t) ← {t};
    N⁻(t) ← {t};
    V_G.Add(t');
    N⁺(t') ← {t'};
    N⁻(t') ← {t'};
    𝒯 ← GetSinkComponents(H);
    foreach subgraph T in 𝒯 do
        T' ← Φ⁻¹(T);
        /* Find the subgraphs C and C'                                */
        v ← GetRandomVertex(T);
        c ← GetShortestCycleStartingAt(v);
        V_C ← {u ∈ V_T : u ∈ c};
        E_C ← {(u, u') : uu' is a word of length 2 contained in c^∞};
        V_C' ← {u ∈ V_T' : Φ(u) ∈ V_C};
        E_C' ← {(u, u') ∈ E_T' : (Φ(u), Φ(u')) ∈ E_C};
        foreach vertex u in V_C' do
            if Φ(u) = v ∧ there is a path in C' from u to a cycle then
                N⁻(t').Add(u);
            end
        end
    end
    Φ(t') ← t;
```

Algorithm 3: Turn every sink component into a single vertex

```
Function AddSourceVertices(G, H, Φ):
    Input: reducible graphs G, H and a 1-block code Φ
    Result: (1) alters G, H by adding a new source vertex to each graph, and
            (2) extends Φ to the new graphs so Φ∞ : X_G → X_H is a
            conjugacy if and only if the original 1-block code was a conjugacy
    G.ReverseEdges();
    H.ReverseEdges();
    AddSinkVertices(G, H, Φ);
    G.ReverseEdges();
    H.ReverseEdges();
```

Algorithm 4: Turn every source component into a single vertex

Function `IsConjugacyReducible`(G, H, Φ)**:**

Input: reducible graphs G, H and a 1-block code Φ

Output: true, if Φ_∞ is a conjugacy; false, otherwise

`AddSinkVertices`(G, H, Φ);

`AddSourceVertices`(G, H, Φ);

`/* Connect each new sink vertices to its respective new source vertex */`

$N^-(s) \leftarrow t$;

$N^-(s') \leftarrow t'$;

return `IsConjugacyIrreducible`(G, H, Φ);

Algorithm 5: Determine if Φ_∞ between reducible graphs is a conjugacy