Parameterized Complexity of Graph Constraint Logic

Tom C. van der Zanden

Department of Computer Science, Utrecht University, Utrecht, The Netherlands
T.C.vanderZanden@uu.nl

Abstract

Graph constraint logic is a framework introduced by Hearn and Demaine [6], which provides several problems that are often a convenient starting point for reductions. We study the parameterized complexity of Constraint Graph Satisfiability and both bounded and unbounded versions of Nondeterministic Constraint Logic (NCL) with respect to solution length, treewidth and maximum degree of the underlying constraint graph as parameters. As a main result we show that restricted NCL remains *PSPACE*-complete on graphs of bounded bandwidth, strengthening Hearn and Demaine's framework. This allows us to improve upon existing results obtained by reduction from NCL. We show that reconfiguration versions of several classical graph problems (including independent set, feedback vertex set and dominating set) are *PSPACE*-complete on planar graphs of bounded bandwidth and that Rush Hour, generalized to $k \times n$ boards, is *PSPACE*-complete even when k is at most a constant.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Nondeterministic Constraint Logic, Reconfiguration Problems, Parameterized Complexity, Treewidth, Bandwidth

Digital Object Identifier 10.4230/LIPIcs.IPEC.2015.282

1 Introduction

Nondeterministic Constraint Logic (NCL) was introduced by Hearn and Demaine in [6] and extended in [8] to a more general graph constraint logic framework. The framework provides a number of problems complete for various complexity classes, that aim to provide a convenient starting point for reductions proving the hardness of games and puzzles. We study the Constraint Graph Satisfiability problem and Nondeterministic Constraint Logic in a parameterized setting, considering (combinations of) solution length, treewidth and maximum degree of the underlying constraint graph as parameters.

As part of the constraint logic framework [6], Hearn and Demaine provide a restricted variant of Nondeterministic Constraint Logic (restricted NCL), in which the constraint graph is planar, 3-regular, uses only weights in $\{1, 2\}$ and the graph is constructed from only two specific vertex types (AND and OR). Restricted NCL is *PSPACE*-complete, and is (due to the restrictions) a particularly suitable starting point for reductions. Hearn and Demaine's reduction creates graphs of unbounded treewidth. We strengthen their result, by providing a new reduction showing that restricted NCL remains *PSPACE*-complete, even when restricted to graphs of bandwidth at most a given constant (which is a subclass of graphs of treewidth at most a given constant). We show hardness by reduction from H-Word Reconfiguration [14].

The puzzle game Rush Hour, when generalized to $n \times n$ boards, is *PSPACE*-complete [4]. Hearn and Demaine provide a reduction from NCL to Rush Hour [7]. As a consequence of this reduction and our improved hardness result for NCL, we show that Rush Hour is

10th International Symposium on Parameterized and Exact Computation (IPEC 2015).
Editors: Thore Husfeldt and Iyad Kanj; pp. 282–293

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

Table 1 Parameterized complexity of graph constraint logic problems.

		Problem				
		CGS	C2E	C2C	BC2E	BC2C
Parameters	–	*NP*-C	*PSPACE*-C	*PSPACE*-C	*NP*-C	*NP*-C
	l	–	*W*[1]-hard	*W*[1]-hard	*W*[1]-hard	*FPT*
	$l + \Delta$	–	*FPT*	*FPT*	*FPT*	*FPT*
	tw	weakly *NP*-C	*PSPACE*-C	*PSPACE*-C	weakly *NP*-H	weakly *NP*-H
	$tw + \Delta$	*FPT*	*PSPACE*-C	*PSPACE*-C	*FPT*	*FPT*

PSPACE-complete even when played on $k \times n$ boards, where k is a constant. This is in contrast to the result of Ravikumar [12] that Peg Solitaire, a game *NP*-complete on $n \times n$ boards, is linear time solvable on $k \times n$ boards for any fixed k.

NCL is also has applications in showing the hardness of reconfiguration problems [10, 5, 2, 9]. For some reconfiguration problems, their hardness on planar graphs of low maximum degree is known by reduction from NCL [5, 9] while their hardness on bounded bandwidth graphs is known by reduction from H-Word Reconfiguration [14, 5, 11]. Our reduction, which combines techniques from Hearn and Demaine's constraint logic [8] and the reductions from H-Word Reconfiguration in [14, 11] unifies these results: we show that a number of reconfiguration problems (including Independent Set, Vertex Cover and Dominating Set) are *PSPACE*-complete on low-degree, planar graphs of bounded bandwidth. Previously, hardness was known only on graphs that *either* are planar and have low degree *or* have bounded bandwidth - we show that the problems remain hard even when both of these conditions hold simultaneously. Note that while a graph of bounded bandwidth also has bounded degree, the graphs created in these reductions have quite large bandwidth. The degree bounds we obtain are much tighter than what would be obtained from the bandwidth bound alone.

Our results concerning the hardness of constraint logic problems are summarized in Table 1. This table shows the parameterized complexity of Constraint Graph Satisfiability (CGS), unbounded configuration-to-edge (C2E) and configuration-to-configuration (C2C) variants of Nondeterministic Constraint Logic and their respective bounded counterparts (BC2E and BC2C) with respect to solution length (l), maximum degree (Δ) and treewidth (tw). If a traditional complexity class is listed this means that the problem is hard for this class even when restricted to instances where the parameter is at most a constant.

In this extended abstract, we discuss only the main result (concerning unbounded C2E and C2C NCL on bounded bandwidth graph) and its applications. For a discussion of the other results in Table 1 and for some omitted proofs, we refer to the full version of this paper [13].

2 Preliminaries

2.1 Constraint Logic

▶ **Definition 1** (Constraint Graph). A *constraint graph* is a graph with edge weights and vertex weights. A *legal configuration* for a constraint graph is an assignment of an orientation to each edge such that for each vertex, the total weight of the edges pointing into that vertex is at least that vertex' weight (its *minimum inflow*).

(a) OR vertex **(b)** AND vertex

Figure 1 The two vertex types from which a restricted constraint graph is constructed: (a) OR vertex and (b) AND vertex. Following the convention set in [6], as a mnemonic weight 2 edges are drawn blue (dark grey) and thick, while weight 1 edges are drawn red (light grey) and thinner.

A fundamental decision problem regarding constraint graphs is that of their satisfiability:

Constraint Graph Satisfiablility

Instance: A constraint graph G.

Question: Does G have a legal configuration?

Constraint Graph Satisfiability (CGS) is *NP*-complete [8].

An important problem regarding constraint graph configurations is whether they can be reconfigured into each other:

Nondeterministic Constraint Logic (C2C)

Instance: A constraint graph G and two legal configurations C_1, C_2 for G.

Question: Is there a sequence of legal configurations from C_1 to C_2, where every configuration is obtained from the previous configuration by changing the orientation of one edge?

This problem is called the configuration-to-configuration (C2C) variant of Nondeterministic Constraint Logic . It is *PSPACE*-complete [8]. The configuration-to-edge variant (C2E) is also *PSPACE*-complete [8]:

Nondeterministic Constraint Logic (C2E)

Instance: A constraint graph G, a target edge e from G and an initial legal configuration C_1 for G.

Question: Is there a sequence of legal configurations, starting with C_1, where every configuration is obtained from the previous by changing the orientation of one edge, so that eventually e is reversed?

For the C2C and C2E problems, bC2C and bC2E denote their bounded variants which ask whether there exists a reconfiguration sequence in which each edge is reversed at most once. These problems are *NP*-complete [8].

Hearn and Demaine [8] consider a restricted subset of constraint graphs, which are planar and constructed using only two specific types of vertices: AND and OR vertices (Figure 1).

The OR vertex has minimum inflow 2 and three incident weight 2 edges. Its inflow constraint is thus satisfied if and only if at least one of its incident edges is directed inwards (resembling an OR logic gate). The AND vertex has minimum inflow 2, two incident weight 1 edges and one incident weight 2 edge. Its constraint is thus satisfied if and only if both weight 1 edges are directed inwards or the weight 2 edge is directed inwards (resembling an AND logic gate). Both C2C and C2E NCL remain *PSPACE*-complete under these restrictions.

(a) Constrained blue-edge terminator **(b)** Free red-edge terminator

Figure 2 Gadgets for terminating loose edges. The constrained blue-edge terminator (a) forces the blue edge A to point into the gadget, while the free red-edge terminator (b) allows the red edge A to point out of the gadget.

2.2 Constraint Logic Gadgets

Some constructions in this paper use existing gadgets (such as crossover) for NCL reductions due to Hearn and Demaine [8]. For the purpose of being self-contained, we reproduce these gadgets here and state (without proof) their functionality.

Edge Terminators. The constrained blue-edge (Figure 2a) terminator allows us to have a loose blue edge that is forced to point outwards, effectively removing the edge from the graph while still meeting the requirement that the graph is built from only AND and OR vertices. The free red-edge terminator (Figure 2b) allows us to have a loose red edge whose orientation can be freely chosen, effectively decreasing the minimum inflow of the vertex to which it is incident by one.

Red-blue Conversion. It is useful to be able to convert a blue edge to a red edge, i.e. we require a gadget which has a blue edge that can (be reconfigured to) point outwards if and only if its red edge is pointing inwards and vice-versa. Hearn and Demaine [8] provide a construction that allows red-blue conversion in pairs, but also note a simpler construction is possible: an AND vertex, with one of its red edges attached to a free red-edge terminator (Figure 2b) can serve as a red-blue conversion gadget.

We also use the crossover gadget due to Hearn and Demaine [8]. We do not reproduce this gadget here, as it is sufficient to know that given a constraint graph we can eliminate any crossings using the crossover gadget, which can be constructed using only AND and OR vertices. For details of the crossover gadget, we refer to [13, 8].

2.3 H-Word Reconfiguration

To show hardness of NCL on bounded bandwidth graphs, we reduce from the H-Word Reconfiguration problem, introduced in [14].

▶ **Definition 2** (H-word)**.** Let $H = (\Sigma, E)$ where Σ is an alphabet and $E \subseteq \Sigma \times \Sigma$ a relation. An *H-word* is a word over Σ such that every pair of consecutive characters (a, b) is an element of E.

H-Word Reconfiguration

Instance: Two H-words W_s, W_g of equal length

Question: Is there a sequence of H-words $W_1, \ldots, W_n$, so that every pair of consecutive words W_i, W_{i+1} can be obtained from each other by changing one character to another and $W_1 = W_s, W_n = W_g$?

▶ **Theorem 3** (Wrochna [14]). *There exists an H such that H-Word Reconfiguration is PSPACE-complete.*

2.4 Bandwidth and Cutwidth

The main result concerns graphs of bounded bandwidth. The proof of Theorem 11 uses the notion of cutwidth.

▶ **Definition 4** (Bandwidth [1]). Let $G = (V, E)$ be a graph and define a one-to-one correspondence $f : V \to \{1, \ldots, |V|\}$. The *bandwidth* of a graph is the minimum over all such correspondences of $\max_{(u,v)\in E} |f(u) - f(v)|$.

▶ **Definition 5** (Cutwidth [1]). Let $G = (V, E)$ be a graph and define a one-to-one correspondence $f : V \to \{1, \ldots, |V|\}$. The *cutwidth* of a graph is the minimum over all such correspondences of $\max_{w\in V} |\{(u, v) \in E | f(u) \leq f(w) < f(v)\}|$.

3 Hardness of Nondeterministic Constraint Logic on Bounded Bandwidth Graphs

In this section we prove the main result, namely that restricted Nondeterministic Constraint Logic remains *PSPACE*-complete even when restricted to graphs of bounded bandwidth (which is a subclass of graphs of bounded treewidth).

To show *PSPACE*-completeness, we reduce from H-Word Reconfiguration. The bandwidth of the constraint graph created in the reduction will depend only on the size of H. Since H-Word Reconfiguration is *PSPACE*-complete for a fixed (finite) H, we obtain a constant bound on the bandwidth.

▶ **Theorem 6.** *There exists a constant c, such that C2C Nondeterministic Constraint Logic is PSPACE-complete on planar constraint graphs of bandwidth at most c that consist of only AND and OR vertices.*

Proof. Let $H = (\Sigma, E)$ be so that H-Word Reconfiguration is *PSPACE*-complete. We provide a reduction from H-Word Reconfiguration to (unrestricted) NCL and then show how to adapt this reduction to work for restricted NCL. Let W_s, W_g be an instance of H-Word Reconfiguration and let n denote the length of W_s. In the following, all vertices are given minimum inflow 2.

We create a matrix of vertices $X_{i,j}$ for $i \in \{1, \ldots, n\}, j \in \Sigma$, the *character vertices*. The orientation of edges incident to $X_{i,j}$ will correspond to whether the character at position i in the word is character j. For every row i of this matrix we create a *universal vertex* U_i and for every $j \in \Sigma$ we create a blue (weight 2) edge connecting U_i and $X_{i,j}$.

In each row $i \in 1, \ldots, n-1$ and for all pairs $(A, B) \notin E$, we create a *relation vertex* $\Delta_{i,A,B}$. We create a red (weight 1) edge connected to $X_{i,A}$ and pass it through a red-blue conversion gadget and connect the blue edge leaving the conversion gadget to $\Delta_{i,A,B}$. We mirror this construction, creating a red edge connected to $X_{i+1,B}$, converting it to blue, and connecting it to $\Delta_{i,A,B}$.

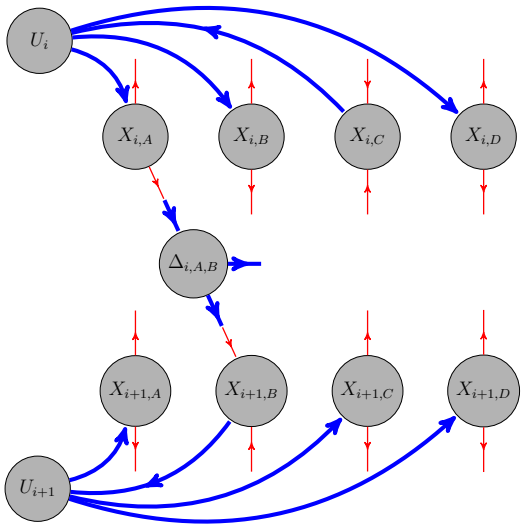


Figure 3 Slice of two rows from the matrix of vertices (over an alphabet of $\Sigma = \{A, B, C, D\}$), and the construction for enforcing non-adjacency of A and B.

Finally, we connect one additional blue edge to $\Delta_{i,A,B}$ and connect it to a constrained blue-edge terminator. This edge serves no purpose (its inflow can never count towards $\Delta_{i,A,B}$) but to make $\Delta_{i,A,B}$ an OR vertex as claimed.

A single instance of this construction is shown in Figure 3, which depicts a slice of two rows from the matrix of vertices. Having created an instance of this construction for (A, B), we might need to create an instance for (A, C). Rather than attaching another red edge to $X_{i,A}$, we instead split the existing red edge leaving $X_{i,A}$ by connecting it to a red-blue conversion gadget, and attaching the resulting blue edge to an AND vertex. The edges leaving the AND vertex may be oriented outward if and only if the red edge leaving $X_{i,A}$ is oriented into the AND vertex, effectively splitting it. We then convert the two red edges of the AND vertex to blue and attach them to $\Delta_{i,A,B}$ and $\Delta_{i,A,C}$ as before. To create further copies of this construction we can repeat the splitting process, so that $X_{i,A}$ eventually has two incident red edges, one connecting to gadgets in row $i-1$ and one connecting to row $i+1$ (the excess red edges in rows 1 and n may instead be connected to a free red edge terminator).

We define a character j to be *in* the word at position i if the edge $(U_i, X_{i,j})$ is oriented towards U_i.

▶ **Claim.** *In any legal configuration, at least one character is in the word at each position.*

Proof. U_i has minimum inflow 2, so at least one of its incident edges $(U_i, X_{i,j})$ must be oriented towards it and hence j is in the word at position i. ◀

▶ **Claim.** *In any legal configuration, if $(A, B) \notin E$ then A can be in the word at position i only if B is not in the word at position $i+1$.*

Proof. If A is in the word at position i, then $X_{i,A}$ is not receiving inflow from U_i, so both of the red edges incident to $X_{i,A}$ most point in towards $X_{i,A}$. On the path from $X_{i,A}$ through $\Delta_{i,A,B}$ to $X_{i+1,B}$ we will first pass through a red-blue conversion gadget. Since the red edge is oriented out of the conversion gadget (and towards $X_{i,A}$), the blue edge must be oriented

into the conversion gadget. We may then encounter several AND vertices (that are used for the "splitting" of red edges incident to $X_{i,A}$), note that since the blue edge is oriented towards the conversion gadget and away from the AND vertex, both red edges must be oriented into the AND vertex. This means that when we encounter another red-blue conversion gadget its blue edge must be oriented into the AND vertex and in turn the red edge incident to the conversion gadget has to be oriented into the conversion gadget. Ultimately, when we arrive at the vertex $\Delta_{i,A,B}$ we find that one of its edges must be oriented out of it (to point into the previously described AND vertices and help satisfy the minimum inflow of $X_{i,A}$). Suppose by contradiction that B is in the word at position $i+1$. By a similar argument, we find that the second edge incident to $\Delta_{i,A,B}$ also has to be oriented out of it. This is impossible, since the third blue edge incident to $\Delta_{i,A,B}$ is also oriented out of it (since that edge is attached to a constrained blue-edge terminator gadget) and thus its minimum inflow constraint is violated. Therefore B can not be in the word at position $i+1$. ◀

Note that this claim implies that if for each position we pick *some* character that is in the position at that word, we end up with a valid H-Word. We say that a configuration for the constraint graph *encodes* a word W if each of that word's characters is in the word at the appropriate position.

▶ **Claim.** *If a legal configuration for the constraint graph encoding W_s may be reconfigured into a legal configuration encoding only W_g (i.e. the configuration encodes no other word), then W_s may be reconfigured into W_g.*

Proof. Suppose we have a reconfiguration sequence of legal configurations $C_1, \ldots, C_m$ so that C_1 encodes W_s and C_m encodes only W_g. We define a reconfiguration sequence of words $W_1, \ldots, W_m$ which has the property that C_i encodes W_i. Let $W_1 = W_s$, we recursively define $W_i, 1 < i \leq m$ as follows: Since C_{i+1} differs from C_i in the orientation of only one edge, the set of words encoded by C_{i+1} differs only from the set of words encoded by C_i by making one character at a position allowed (in the word) or disallowed (was in the word in C_i but not in the word in C_{i+1}). If a character at a position becomes allowed in C_{i+1}, let $W_{i+1} = W_i$ (since C_i encodes W_i, C_{i+1} also encodes W_i). If a character at a position becomes disallowed (i.e. is no longer in the word), we obtain W_{i+1} from W_i by changing the character at that position to some allowed character (of which there is at least one). Following these steps, since the final configuration encodes only W_g, we obtain a reconfiguration sequence from W_s to W_g as claimed. ◀

Note that we may have multiple choices for W_i because C_i can encode more than one word. It suffices to simply pick one of the words it encodes, while ensuring it differs in at most one character from the previous and next.

▶ **Claim.** *Given a H-word W of length n, there exists a legal configuration for the constraint graph encoding only W.*

Proof. Pick the orientation of edge $(U_i, X_{i,j})$ to be towards U_i if the character in W at position i is j, and towards $X_{i,j}$ otherwise. Clearly this constraint graph encodes only W. This configuration can be extended to a legal configuration for the remaining (relation) vertices by noticing the following: if there is a relation vertex $\Delta_{i,A,B}$ then either A is not in the word at position i or B is not in the word at position j. Suppose w.l.g. that B is not in the word, then $X_{i+1,B}$ is receiving inflow from U_{i+1} and hence its incident red edges may be pointing outwards, which (after passing through the red-blue conversion gadgets and

splitting AND vertices as described before) allows us to satisfy the inflow requirement of $\Delta_{i,A,B}$. ◀

▶ **Claim.** *If two H-Words W_1, W_2 differ by only one character, then a constraint graph G encoding only W_1 can be reconfigured into one encoding only W_2.*

Proof. Suppose that W_1 and W_2 differ by changing the character at position i from A to B. We may first reconfigure G from encoding only W_1 to encoding both W_1 and W_2, and then reconfigure it to encode only W_2. This temporarily increases the inflow U_i receives from 2 (receiving inflow only from $X_{i,A}$) to 4 (receiving inflow from both $X_{i,A}$ and $X_{i,B}$) back to 2 (receiving inflow from only $X_{i,A}$). It may be required to reorient some edges to satisfy the inflows of the relation vertices, but this is always possible because neither A nor B conflicts with any preceding or succeeding characters. ◀

Note that this claim implies that if W_s can be reconfigured into W_g, then a constraint graph encoding only W_s can be reconfigured into one encoding only W_g. This completes the reduction.

All that remains to show is that this graph can be adapted so that it only uses AND and OR vertices and becomes planar, and that the resulting graph has bounded bandwidth. The only vertices that are not already AND or OR vertices are the universal vertices U_i. Note that taking a single OR vertex (that has 3 incident edges at least one of which has to be oriented inwards) we can attach another OR vertex to one of its edges to obtain a structure that has 4 external edges, at least one of which has to be oriented inwards. Repeating this procedure n times we obtain a tree of OR vertices with $n+3$ external edges, at least one of which has to be oriented inwards, which can replace the universal vertex.

To make the graph planar, we can use Hearn and Demaine's crossover gadget [8, 13]. While this may increase the graph's bandwidth, the following argument that it remains bounded by a constant:

We create a number of bags $B_1, \dots B_{n-1}$ that are subsets of vertices, with the property that the size of each bag depends only on H (which is fixed) and that edges connect only vertices that are both in the same bag, or a vertex in bag B_i with a vertex in bag B_{i+1}. This is achieved by taking in bag i the vertices $\{X_{i,j} : j \in \Sigma\}, \{\Delta_{i,A,B}\,(A,B) \notin E\}, \{X_{i+1,j} : j \in \Sigma\}$, the vertices in gadgets between them (i.e. the red-blue conversion gadgets and AND vertices used in the splitting process) and the construction replacing the universal vertices U_i and U_{i+1}. This shows the bandwidth of the graph is bounded by a constant c, since we can order the vertices so that a vertex in B_i precedes a vertex in B_{i+1} (and pick an arbitrary order for the vertices in the same bag).

We have thus shown how to reduce H-Word Reconfiguration to NCL, shown how to adapt our reduction to use only AND and OR vertices and make the resulting graph planar and that the resulting graph has bounded bandwidth and have thus shown Theorem 6. ◀

Theorem 6 also holds for C2E Nondeterministic Constraint Logic:

▶ **Theorem 7.** *There is a constant c, such that C2E* Nondeterministic Constraint Logic *is PSPACE-complete, even on planar constraint graphs of treewidth (bandwidth) at most c that use only AND and OR vertices.*

Proof. H-Word Reconfiguration remains *PSPACE*-complete, even when instead of requiring that one H-word is reconfigured in to another, we ask whether it is possible to reconfigure a given H-word so that a given character appears at a specific position (without requiring anything regarding the remaining characters in the word). This can be seen by

examining the proof in [14], noting that we may modify the Turing machine to, upon reaching an accepting state, move the head to the start of the tape and write a specific symbol there. The proof of Theorem 6 can then easily be adapted to work for C2E NCL (since a character appearing at a specific position corresponds to reversing a specific edge). ◀

We note that Theorems 6 and 7 may be further strengthened by requiring that all OR vertices in the graph are *protected*, i.e., in any legal configuration at least one of its incident edges is directed outwards. Hearn and Demaine [8] show how to construct an OR vertex using only AND vertices and protected OR vertices.

4 Applications

As a result of our hardness proof for NCL, we (nearly) automatically obtain tighter bounds for other problems that reduce from NCL in their hardness proofs. If the reduction showing *PSPACE*-hardness for a problem is bandwidth-preserving in the sense that the bandwidth of the resulting graph only depends on the bandwidth of the original constraint graph, then that problem remains *PSPACE*-hard even when limited to instances of bounded bandwidth. Since reductions from NCL usually work by locally replacing AND and OR vertices in the graph with gadgets that simulate their functionality, such reductions are often bandwidth-preserving.

Independent Set Reconfiguration (IS-R)

Instance: Graph $G = (V, E)$, independent sets $S_s, S_g \subseteq V$ of G, integer threshold k.

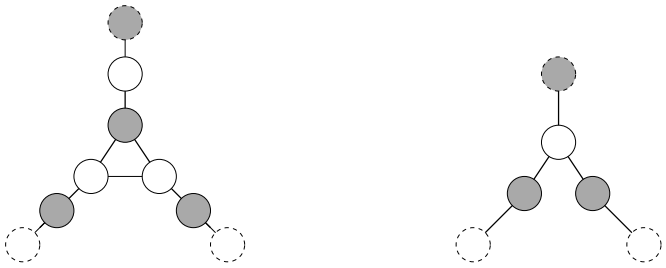
Question: Is there a sequence of independent sets $S_1, \ldots, S_n$, so that for all i, S_{i+1} is obtained from S_i by the addition or removal of one vertex, $|S_i| \geq k$ and $S_1 = S_s, S_n = S_g$?

This is the *token addition-and-removal (TAR)* version of IS-R. It is also possible to define a *token jumping (TJ)* variant (in which we obtain one independent set from the next by removing and immediately replacing a vertex) and a *token sliding* variant (in which when we remove a vertex, we must replace it with an incident vertex, i.e. "sliding" the token/vertex along an edge).

▶ **Theorem 8.** *TAR, TJ and TS versions of* Independent Set Reconfiguration *are PSPACE-complete on planar, maximum degree 3 graphs of bounded bandwidth.*

Proof. Hearn and Demaine [7] provide a reduction from IS-R to TJ and TS versions of IS-R. The gadgets used in their reduction are reproduced in Figure 4. Since the reduction works by replacing every vertex with a construction of at most 6 other vertices, the bandwidth of the graph created in the reduction is at most 6 times the bandwidth of the original constraint graph. The theorem thus follows immediately from our hardness result and Hearn and Demaine's reduction [7] and the fact that the TAR version is equivalent to TJ [10]. ◀

Similarly to Independent Set Reconfiguration, we can define reconfiguration versions of Vertex Cover (VC-R), Feedback Vertex Set (FVS-R), Induced Forest (IF-R), Odd Cycle Transversal (OCT-R) and Induced Bipartite Subgraph (IBS-R). Note that for IS-R, IF-R, IBS-R the size of the vertex subset is never allowed to drop below the threshold, while for VC-R, FVS-R and OCT-R the size is never allowed to exceed the threshold. These problems are *PSPACE*-complete on bounded bandwidth graphs by reduction from H-Word Reconfiguration [11]. We strengthen this result, noting that our proof follows the same reasoning as in [11]:



(a) OR vertex **(b)** AND vertex

Figure 4 Gadgets used in the reduction from IS-R to NCL. Dashed vertices represent vertices that are part not of the gadget itself, but of other gadgets. The dark gray vertices represent an example independent set.

▶ **Theorem 9.** *TAR, TJ and TS versions of* Independent Set *(IS-R),* Vertex Cover *(VC-R),* Feedback Vertex Set *(FVS-R),* Induced Forest *(IF-R),* Odd Cycle Transversal *(OCT-R) and* Induced Bipartite Subgraph *(IBS-R) are PSPACE-complete on planar graphs of bounded bandwidth and low maximum degree.*

Proof. By Theorem 8, IS-R is *PSPACE*-complete on planar graphs of bounded bandwidth and maximum degree 3. Since an independent set is the complement of a vertex cover, the theorem holds for VC-R on maximum degree 3 graphs.

By replacing every edge in the graph by a triangle, we can reduce VC-R to FVS-R (since picking at least one vertex of every edge is equivalent to picking at least one vertex of every triangle), thus showing the theorem for FVS-R on maximum degree 6 graphs. We note that in such a graph a feedback vertex set is also an odd cycle transversal, thus also showing the theorem for OCT-R on maximum degree 6 graphs.

Finally, the theorem holds for IF-R and IBS-R on maximum degree 6 graphs by considering complements of solutions for FVS-R and OCT-R. ◀

Another related reconfiguration problem is that of Dominating Set Reconfiguration (DS-R). In [5], the authors show that DS-R is *PSPACE*-complete on planar graphs of maximum degree six by reduction from NCL and *PSPACE*-complete on graphs of bounded bandwidth by a reduction from VC-R. The following theorem that unifies these results follows immediately from our improved result concerning VC-R:

▶ **Theorem 10.** *The TAR version of* Dominating Set Reconfiguration *is PSPACE-complete, even on planar, maximum degree six graphs of bounded bandwidth.*

The puzzle game Rush Hour, in which the player moves cars horizontally and vertically with the goal to free a specific car from the board, is *PSPACE*-complete [4] when played on an $n \times n$ board. Another consequence of our result is that Rush Hour is *PSPACE*-complete even on rectangular boards where one of the dimensions of the board is constant:

▶ **Theorem 11.** *There exists a constant k, so that Rush Hour is PSPACE-complete when played on boards of size $k \times n$.*

Proof. Hearn and Demaine [7] provide a reduction from restricted NCL, showing how to construct AND and OR vertices as gadgets in Rush Hour and how to connect them together using straight line and turn gadgets. Given a planar constraint graph, it can be drawn in the grid (with vertices on points of the grid and edges running along the lines of the grid), after

which vertices can be replaced by their appropriate gadgets and edges with the necessary line and turn gadgets. However, starting with a graph of bounded bandwidth does not immediately give a suitable drawing in a grid of which one of the dimensions is bounded (from which the theorem would follow, since all of the gadgets have constant size).

Given a restricted NCL graph of bounded bandwidth, we note that this graph also has bounded cutwidth [1]. Given such a graph with n vertices and cutwidth at most c, it is possible to arrange it in a $(c+1) \times 3n$ grid, so that vertices of the graph are mapped to vertices of the grid, the edges of the graph run along edges of the grid, and no two edges or vertices get mapped to the same edge or vertex in the grid. This is achieved by placing all of the vertices of the graph in a single column (noting that 3 rows are required for each vertex since it has 3 incident edges) and (due to the graph having cutwidth c) the edges can be placed in the remaining c columns (placing each edge in a distinct column). However, even when starting with a planar NCL graph, the resulting embedding may have crossings. These can be eliminated using the crossover gadget, noting that since the crossover gadget has constant size, the resulting graph can still be drawn in a $O(c) \times O(n)$ grid. We can now use the existing gadgets from [7] to finish the reduction, showing Rush Hour *PSPACE*-complete on boards of which one of the dimensions is bounded by a constant. ◀

Note that this result is likely to carry over to show other board games *PSPACE*-complete on $n \times k$ boards, such as Sokoban or Plank Puzzles, which also reduce from NCL in their hardness proofs [8].

This result is in contrast to [12], where it is shown that Peg Solitaire, when played on $k \times n$ boards, is linear time solvable for any fixed k. An important distinction here is that the length of a solution in a Peg Solitaire game is *bounded* by the number of pegs (since every move removes one peg from play) whereas Rush Hour games are *unbounded*: any length move sequence is possible, though obviously after an exponential number of moves, positions would be repeated.

5 Conclusions

We have studied the parameterized complexity of constraint logic problems with regards to (combinations of) solution length, maximum degree and treewidth as parameters. As a main result, we showed that RESTRICTED NONDETERMINISTIC CONSTRAINT LOGIC remains *PSPACE*-complete on graphs of bounded bandwidth, strengthening Hearn and Demaine's framework [8].

By combining Wrochna's proof [14] and Hearn and Demaine's [8] constraint logic techniques, we have managed to get the best of both worlds: for several reconfiguration problems, we showed hardness for graphs that are not only planar and have low maximum degree, but that also have bounded bandwidth. This not only strengthens Wrochna's results, but also makes it easier to prove hardness for reconfiguration on bounded bandwidth graphs by merging H-WORD RECONFIGURATION into the more convenient NCL framework.

Note that the constant c in Theorem 6 has not been calculated precisely, but an informal analysis of Wrochna's proof [14] and our reduction suggests it is very large (growing with the 12^{th} power of the original instance size). This raises two open questions: one is to determine a tighter bound on the value of c, and the other is to determine whether efficient algorithms exist for solving reconfiguration problems when the graph's bandwidth is bounded by more practical values. Some progress in this direction has already been made [3].

We have studied the parameterized complexity of CGS and NCL. Hearn and Demaine [8] have defined several other problems related to constraint graphs, including two player

and multiple player constraint graph games, complete for classes such as *EXPTIME* and *NEXPTIME*. Studying these problems in a parameterized setting gives rise to several interesting open problems.

Acknowledgement. I thank my advisor, Hans Bodlaender, for his guidance and useful comments and suggestions.

References

1. Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical computer science*, 209(1):1–45, 1998.
2. Paul Bonsma and Luis Cereceda. Finding paths between graph colourings: Pspace-completeness and superpolynomial distances. *Theoretical Computer Science*, 410(50):5215–5226, 2009.
3. Erik D. Demaine, Martin L. Demaine, Eli Fox-Epstein, Duc A Hoang, Takehiro Ito, Hirotaka Ono, Yota Otachi, Ryuhei Uehara, and Takeshi Yamada. Polynomial-time algorithm for sliding tokens on trees. In *Algorithms and Computation*, pages 389–400. Springer, 2014.
4. Gary William Flake and Eric B. Baum. Rush hour is pspace-complete, or "why you should generously tip parking lot attendants". *Theoretical Computer Science*, 270(1):895–911, 2002.
5. Arash Haddadan, Takehiro Ito, Amer E. Mouawad, Naomi Nishimura, Hirotaka Ono, Akira Suzuki, and Youcef Tebbal. The complexity of dominating set reconfiguration. *arXiv preprint arXiv:1503.00833*, 2015.
6. Robert A. Hearn and Erik D. Demaine. The nondeterministic constraint logic model of computation: Reductions and applications. In *Automata, Languages and Programming*, pages 401–413. Springer, 2002.
7. Robert A. Hearn and Erik D. Demaine. Pspace-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science*, 343(1):72–96, 2005.
8. Robert A. Hearn and Erik D. Demaine. *Games, puzzles, and computation*. CRC Press, 2009.
9. Takehiro Ito, Erik D. Demaine, Nicholas J.A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the complexity of reconfiguration problems. In *Algorithms and Computation*, pages 28–39. Springer, 2008.
10. Marcin Kamiński, Paul Medvedev, and Martin Milanič. Complexity of independent set reconfigurability problems. *Theoretical computer science*, 439:9–15, 2012.
11. Amer E. Mouawad, Naomi Nishimura, Venkatesh Raman, and Marcin Wrochna. Reconfiguration over tree decompositions. In *Parameterized and Exact Computation*, pages 246–257. Springer, 2014.
12. Bala Ravikumar. Peg-solitaire, string rewriting systems and finite automata. In *Algorithms and Computation*, pages 233–242. Springer, 1997.
13. Tom C. van der Zanden. Parameterized complexity of graph constraint logic. *arXiv preprint:1509.02683*, 2015.
14. Marcin Wrochna. Reconfiguration in bounded bandwidth and treedepth. *arXiv preprint arXiv:1405.0847*, 2014.