Journal of Computer and System Sciences 93 (2018) 1–10

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Reconfiguration in bounded bandwidth and tree-depth ☆

Marcin Wrochna

University of Warsaw, Poland

ARTICLE INFO

Article history:
Received 23 August 2016
Received in revised form 8 September 2017
Accepted 21 November 2017
Available online 5 December 2017

Keywords:
Reconfiguration
Recoloring
Treewidth
Bandwidth
Tree-depth
Graph homomorphism

ABSTRACT

We show that several reconfiguration problems known to be PSPACE-complete remain so even when limited to graphs of bounded bandwidth (and hence pathwidth and treewidth). The essential step is noticing the similarity to very limited string rewriting systems, whose ability to directly simulate Turing Machines is classically known. On the other hand, we show that a large class of natural reconfiguration problems (coming from graph homomorphisms) becomes tractable on graphs of bounded tree-depth, and prove a dichotomy showing this to be in some sense tight.

1. Introduction

In the *reconfiguration* framework one studies how combinatorial objects can be transformed into one another by sequences of small transformations. Usually the set of objects considered is the solution space of a known combinatorial problem and the transformations allowed are changes to a single element of the solution. For example, the problem k-Coloring Reachability [1–7] is defined as follows: given two (proper) k-colorings of a graph, can one be transformed into the other by changing one color at a time (maintaining a proper coloring throughout). Another well studied example is *independent set reconfiguration* [8–11], where given a set of tokens placed on vertices of a graph, one asks whether it is possible to reach another configuration of the tokens by moving one at a time, so that no two tokens are ever adjacent.

The main motivation for studying reconfiguration is to investigate the solution space of combinatorial problems, knowledge of which is applied in local search heuristics and random solution sampling, see e.g. [12]. In some settings reconfiguration problems arise directly, e.g. a construction by Hearn and Demaine [8] allowed to show that many popular puzzles are PSPACE-complete [13]. More interestingly, Heijltjes and Houston used the construction to prove that deciding the equivalence of proofs in a certain proof system is PSPACE-complete [14], answering a question about normal forms of proofs that arose in this context.

Several results suggested that if the graph underlying the combinatorial problem is assumed to have some structure that allows one to find solutions in polynomial time, then questions about reconfiguring solutions can also be answered in polynomial time. For Independent Set, deciding the existence of any solution of some size is a classic NP-complete problem. It remains NP-complete even when limited to cubic planar graphs [15], but can be solved in polynomial time for bipartite graphs, for claw-free graphs [16–18], and for graphs of bounded treewidth, among others (see [19]). In the reconfiguration variants, the reachability problem is PSPACE-complete, even for subcubic planar graphs, by a general result of Hearn and

☆ This work was supported by the Foundation for Polish Science (HOMING PLUS/2011-4/8).
E-mail address: m.wrochna@mimuw.edu.pl.

https://doi.org/10.1016/j.jcss.2017.11.003
0022-0000/

Demaine [8]. Recently it has been shown to be solvable in polynomial time for claw-free graphs by Bonsma et al. [20] and for cographs by Bonsma [11].

Simple algorithms for Independent Set, k-Coloring and many other problems are known for graphs of bounded treewidth (see [21] for an overview and definitions). This motivated the question of determining the complexity of reconfiguration problems in graphs of bounded treewidth, first posed by Bonsma [22]. He further motivated the question by showing that the techniques used for such classes—dynamic programming—apply to reconfiguration, by using them to show polynomial algorithms for Shortest Path Reachability in planar graphs and for TAR Reachability in cographs (i.e., P_4-free graphs) [11]. We answer it in the negative, showing that several such problems are PSPACE-complete even when limited to graphs of bounded bandwidth, a notion strictly stronger than treewidth or pathwidth.

To rigorously explore possible patterns in the complexity of reconfiguration problems, we introduce reconfiguration of homomorphisms, or H-colorings for digraphs H. On one hand, this naturally generalizes k-Coloring Reachability. On the other hand, this extends the work of Gopalan et al. [23] on reconfiguration of generalized satisfiability problems to constraint satisfaction problems (by allowing variables to take more than two different values, but restricting our attention to a single binary relation). H-colorings provide a special—though very expressive—case of constraint satisfaction problems. Indeed, Feder and Vardi [24] proved that every constraint satisfaction problem (with a fixed template, see [24]) is polynomial-time equivalent to some H-coloring problem, for some digraph H. For these reasons, the author believes them to be a good setting for formally describing patterns arising in reconfiguration problems.

Results

We show that there exist integers k, b such that reachability in reconfiguration variants of k-Coloring, Independent Set and Shortest Path is PSPACE-complete even when limited to graphs of bandwidth b. As intermediary steps that highlight where the hardness comes from, we define a simple problem on words and use it to show that reconfiguring k-list-colorings is PSPACE-complete even for very specific graphs of pathwidth 2, and that there is a fixed digraph H such that reconfiguring H-colorings of an input graph G is PSPACE-complete even if G is limited to be a directed path. We also argue that edge orientation is not crucial by showing the same result for undirected cycles.

Finally, we give an algorithm for H-coloring reconfiguration in graphs of bounded tree-depth. This being very restrictive, the algorithm is not very surprising nor practical, but by connecting the fact with the PSPACE-hardness reductions we show the following: for a class of graphs $\mathcal{C}$ closed under subgraphs, H-Coloring Reachability problems have polynomial algorithms in $\mathcal{C}$ for each digraph H if and only if $\mathcal{C}$ has bounded tree-depth. This strongly suggest that no better meta-algorithmic result is possible for reconfiguration problems, unless the constraints themselves are somehow limited or additional parameters that could be small in practice are considered.

Definitions of the problems and graph parameters are, because of their number, only recalled in the section concerning them. For other, standard definitions we refer to the book of Diestel [25]. We consider digraphs (graphs) as representing arbitrary relations, that is, we allow graphs and digraphs to have loops, unless stated otherwise; we don't allow multiple edges, but digraphs can have edges in both directions between two vertices.

2. String rewriting systems

The general idea in our reductions is to construct one arbitrarily complicated set of local rules with a fixed instance of the problem; connecting such instances in a path then allows to simulate the tape of a Turing Machine in a graph of bandwidth bounded by this instance's size. To formalize this into clearly delineated parts we give reductions from the word problem in string rewriting systems (also known as *semi-Thue systems* and essentially equivalent to *unrestricted grammars* and *finitely presented monoids*), whose ability to directly simulate Turing Machines is a well-known, classical result. We construct a very limited PSPACE-complete string rewriting system and later interpret it as an intermediary reconfiguration problem, from which reductions to other problems are easy.

A *string rewriting system* (SRS for short) is a pair (Σ, R) where Σ is a finite alphabet and R is a set of rules, where each rule is an ordered pair of words $(\alpha, \beta) \in \Sigma^* \times \Sigma^*$. A rule can be applied to a word by replacing one subword by the other, that is, for two words $s, t \in \Sigma^*$, we write $s \to_R t$ if $s = u\alpha v$ and $t = u\beta v$ for some rule $(\alpha, \beta) \in R$ and words $u, v \in \Sigma^*$. The reflexive transitive closure of this relation defines a reachability relation $\to_R^*$, where a word t can be reached from another s iff it can be obtained from s by repeated application of rules from R. The *word problem* of R is the problem of deciding, given two word $s, t \in \Sigma^*$, whether $s \to_R^* t$.

A string rewriting system is called *symmetric* when $(\alpha, \beta) \in R \iff (\beta, \alpha) \in R$, in other words, rules are unordered pairs and the reachability relation is symmetric (this is also known as a Thue system). An SRS is called *balanced* if for each rule $(\alpha, \beta) \in R$ we have $|\alpha| = |\beta|$, and 2-*balanced* if for each rule $(\alpha, \beta) \in R$, $|\alpha| = |\beta| = 2$. In a balanced system, only words of the same length can be equivalent.

The word problem of certain 2-balanced symmetric SRSs is known to be PSPACE-complete. This fact is a folklore variant of the undecidability of general SRSs, whose proof by Post [26] (and independently by A.A. Markov [27]) was described as "the first unsolvability proof for a problem from classical mathematics". The essential steps are: encoding the configurations of a Turing machine (TM) as a string so that a transition corresponds to string rewriting; padding the encoding so that the corresponding system is balanced; and noticing that the non-reversibility of a TM transition (the asymmetry of the