SIAM J. COMPUT.
Vol. 38, No. 6, pp. 2330–2355

THE CONNECTIVITY OF BOOLEAN SATISFIABILITY: COMPUTATIONAL AND STRUCTURAL DICHOTOMIES*

PARIKSHIT GOPALAN†, PHOKION G. KOLAITIS‡, ELITZA MANEVA‡, AND CHRISTOS H. PAPADIMITRIOU§

Abstract. Boolean satisfiability problems are an important benchmark for questions about complexity, algorithms, heuristics, and threshold phenomena. Recent work on heuristics and the satisfiability threshold has centered around the structure and connectivity of the solution space. Motivated by this work, we study structural and connectivity-related properties of the space of solutions of Boolean satisfiability problems and establish various dichotomies in Schaefer's framework. On the structural side, we obtain dichotomies for the kinds of subgraphs of the hypercube that can be induced by the solutions of Boolean formulas, as well as for the diameter of the connected components of the solution space. On the computational side, we establish dichotomy theorems for the complexity of the connectivity and *st*-connectivity questions for the graph of solutions of Boolean formulas. Our results assert that the intractable side of the computational dichotomies is PSPACE-complete, while the tractable side—which includes but is not limited to all problems with polynomial-time algorithms for satisfiability—is in P for the *st*-connectivity question, and in coNP for the connectivity question. The diameter of components can be exponential for the PSPACE-complete cases, whereas in all other cases it is linear; thus, diameter and complexity of the connectivity problems are remarkably aligned. The crux of our results is an expressibility theorem showing that in the tractable cases, the subgraphs induced by the solution space possess certain good structural properties, whereas in the intractable cases, the subgraphs can be arbitrary.

Key words. Boolean satisfiability, computational complexity, PSPACE, PSPACE-completeness, dichotomy theorems, graph connectivity

AMS subject classifications. 03D15, 68Q15, 68Q17, 68Q25, 05C40

DOI. 10.1137/07070440X

1. Introduction.

In 1978, Schaefer [31] introduced a rich framework for expressing variants of Boolean satisfiability and proved a remarkable *dichotomy theorem*: the satisfiability problem is in P for certain classes of Boolean formulas, while it is NP-complete for all other classes in the framework. This result pinpoints the computational complexity of numerous well-known variants of Boolean SAT, such as 3-SAT, HORN 3-SAT, NOT-ALL-EQUAL 3-SAT, and 1-IN-3 SAT. Schaefer's dichotomy theorem yields a classification of the computational complexity of constraint satisfaction problems (CSPs) over the Boolean domain. Feder and Vardi [15] conjectured that a dichotomy theorem holds for the complexity of CSPs over arbitrary finite domains. This conjecture remains open to date, in spite of concerted efforts and some partial progress, such as the case of domains of size 3 [8].

*Received by the editors October 4, 2007; accepted for publication (in revised form) November 25, 2008; published electronically March 4, 2009. A preliminary version of this article appeared in ICALP'06 [16].

http://www.siam.org/journals/sicomp/38-6/70440.html

†Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195 (parik@cs.washington.edu). This work was done in part while the first author was a summer intern at IBM Almaden.

‡Department of Computer Science Principles and Methodologies, IBM Almaden Research Center, San Jose, CA 95120 (kolaitis@almaden.ibm.com, enmaneva@us.ibm.com). This work was done while the second author was on leave from UC Santa Cruz and while the third author was an intern at IBM Almaden.

§Department of Electrical Engineering and Computer Science, University of California at Berkeley, Berkeley, CA 94720 (christos@cs.berkeley.edu). The fourth author's research was supported by NSF grant CCF0635319, a gift from Yahoo!, a MICRO grant, and a grant from the France-Berkeley Fund.

In this article we concentrate on the Boolean CSPs, but we ask a new set of questions, which was motivated from the study of random instances of satisfiability. In recent years, the structure of the space of solutions for random instances has been the main consideration at the basis of both algorithms for and mathematical analysis of the satisfiability problem [3, 26, 28, 25]. It has been conjectured for 3-SAT [28, 4] and proved for 8-SAT [27, 1] that the solution space fractures as one approaches the *critical region* from below. This apparently leads to performance deterioration of the standard satisfiability algorithms, such as WalkSAT [32] and DPLL [2]. It is also the main consideration behind the design of the survey propagation algorithm, which has far superior performance on random instances of satisfiability [28]. This body of work has served as a motivation for us to pursue the investigation reported here.

Our aim in this article is to carry out a comprehensive exploration of the *connectivity properties* of the space of solutions of Boolean formulas from a worst-case point of view. The solutions (satisfying assignments) of a given n-variable Boolean formula φ induce a subgraph $G(\varphi)$ of the n-dimensional hypercube. Thus, the following two decision problems, called the *connectivity problem* and the *st-connectivity problem*, arise naturally: (i) Given a Boolean formula φ, is $G(\varphi)$ connected? (ii) Given a Boolean formula φ and two solutions $\mathbf{s}$ and $\mathbf{t}$ of φ, is there a path from $\mathbf{s}$ to $\mathbf{t}$ in $G(\varphi)$?

While there has been an intensive study of the structure of the solution space of Boolean satisfiability problems for random instances, our work seems to be the first to explore this issue from a worst-case viewpoint. A priori, it is not clear what to expect. Is the hardness of the satisfiability question for a given CSP at all correlated with the properties of the graphs realizable as $G(\varphi)$ of formulas in the given class? What kinds of graphs are realizable? Are there Boolean CSPs whose connectivity graphs can have connected components with exponential diameter and, if yes, can we characterize these CSPs? What is the complexity of the *st*-connectivity and the connectivity problem, and is it correlated with the answer to the previous question?

In this article, we investigate the above questions for Boolean CSPs and obtain answers for all of them. Our first complexity-theoretic result is a dichotomy theorem for the *st*-connectivity problem. This result reveals that the tractable side is much more generous than the tractable side for satisfiability, while the intractable side is PSPACE-complete. Specifically, Schaefer showed that the satisfiability problem is solvable in polynomial time precisely for formulas built from Boolean relations all of which are bijunctive, or all of which are Horn, or all of which are dual Horn, or all of which are affine. We identify new classes of Boolean relations, called *tight* relations, that properly contain the classes of bijunctive, Horn, dual Horn, and affine relations. We show that *st*-connectivity is solvable in linear time for formulas built from tight relations, and PSPACE-complete in all other cases. Our second main result is a dichotomy theorem for the connectivity problem: it is in coNP for formulas built from tight relations, and PSPACE-complete in all other cases.

In addition to these two complexity-theoretic dichotomies, we establish a structural dichotomy theorem for the diameter of the connected components of the solution space of Boolean formulas. This result asserts that, in the PSPACE-complete cases, the diameter of the connected components can be exponential, but in all other cases it is linear. Thus, small diameter and tractability of the *st*-connectivity problem are remarkably aligned.

To establish our results, the main challenge is to show that for noneasy relations, both the connectivity problem and the *st*-connectivity problem are PSPACE-hard. In Schaefer's dichotomy theorem, NP-hardness of satisfiability was a consequence of

an *expressibility* theorem, which asserted that every Boolean relation can be obtained as a projection over a formula built from clauses in the "hard" relations. Schaefer's notion of expressibility is inadequate for our problem. Instead, we introduce and work with a delicate and stricter notion of expressibility, which we call *structural expressibility*. Intuitively, structural expressibility means that, in addition to definability via a projection, the space of witnesses of the existential quantifiers in the projection has certain strong connectivity properties that allow us to capture the graph structure of the relation that is being defined. It should be noted that Schaefer's dichotomy theorem can also be proved using a Galois connection and Post's celebrated classification of the lattice of Boolean clones (see [5]). This method, however, does not appear to apply to connectivity, as the boundaries discovered here cut across Boolean clones. Thus, the use of structural expressibility or some other refined definability technique seems unavoidable.

The first step towards proving PSPACE-completeness is to show that both connectivity and *st*-connectivity are hard for 3-CNF-formulas; this is proved by a reduction from a generic PSPACE computation. Next, we identify the simplest relations that are not tight: these are ternary relations whose graph is a path of length 4 between assignments at Hamming distance 2. We show that these paths can structurally express all 3-CNF clauses. The crux of our hardness result is an *expressibility* theorem to the effect that one can structurally express such a path from any set of relations which is not tight.

Finally, we show that all *tight* relations have "good" structural properties. Specifically, in a tight relation every component has a unique minimum element, or every component has a unique maximum element, or the Hamming distance coincides with the shortest-path distance in the relation. These properties are inherited by every formula built from tight relations and yield both small diameter and linear algorithms for *st*-connectivity.

Related work. In parallel and independently of our work, a similar dichotomy was found for the k-colorability problem by Bonsma and Cereceda [6] and Cereceda, van den Heuvel, and Johnson [9]. Specifically, Cereceda, van den Heuvel, and Johnson [9] showed that the case of 3-colorability has properties similar to our tight problems—its structure implies (by a proof that is much more intricate than ours for tight problems) that the diameter of connected components of the graph of 3-colorings is at most quadratic in the number of vertices. This implies that the *st*-connectivity question for 3-colorability is in coNP. Furthermore, Bonsma and Cereceda [6] showed that for $k \geq 4$, *st*-connectivity is PSPACE-complete. This result shows an interesting complexity-theoretic difference between 3-colorability and 4-colorability. It also indicates that an extension of our result to larger domains will be quite challenging, since one would have to identify a set of "tight" problems that includes 3-colorability. It is conceivable that characterizing the complexity of the connectivity question is easier than characterizing the complexity of the satisfiability question, but at present there is no conjecture that generalizes both the Boolean case and the case of colorings.

In an earlier piece of related work, Brightwell and Winkler [7] considered the connectivity of the graph of solutions of the graph-homomorphism problems (which are a subclass of CSPs). They characterized those graphs H for which the graph-homomorphism problem with template H, also known as the H-coloring problem, has a graph of solutions that is always connected. Their motivation is the study of uniqueness of Gibbs measures, which is related to the performance of standard Markov chain algorithms for sampling and counting solutions. We note that the state-space

of many of the Markov chains studied in this area is the set of solutions, and steps of the chain are single-variable flips, so that the chain is just a random walk on the graph of solutions considered here.

In a different direction, there has been a substantial body of work on dichotomy theorems for various aspects of constraint satisfaction on the Boolean domain, including optimization [10, 14, 21], counting [12], inverse satisfiability [20], minimal satisfiability [22], lexicographically minimal satisfiability [30], and propositional abduction [13]. Our results contribute to this body of work.

Finally, it is worth mentioning that satisfiability in the context of random instances has also been considered not only for specific CSPs such as k-SAT and k-colorability, but also for general CSPs. Several models of general random CSPs have been studied by Creignou and Daudé [11], and independently by Molloy [29].

2. Basic concepts and statements of results. A CNF-formula is a Boolean formula of the form $C_1 \wedge \cdots \wedge C_n$, where each C_i is a clause, i.e., a disjunction of literals. If k is a positive integer, then a k-CNF-formula is a CNF-formula $C_1 \wedge \cdots \wedge C_n$ in which each clause C_i is a disjunction of at most k literals.

A *logical relation* R is a nonempty subset of $\{0,1\}^k$ for some $k \geq 1$; k is the *arity* of R. Let $\mathcal{S}$ be a finite set of logical relations. A CNF($\mathcal{S}$)*-formula* over a set of variables $V = \{x_1, \ldots, x_n\}$ is a finite conjunction $C_1 \wedge \cdots \wedge C_n$ of clauses built using relations from $\mathcal{S}$, variables from V, and the constants 0 and 1; this means that each C_i is an expression of the form $R(\xi_1, \ldots, \xi_k)$, where $R \in \mathcal{S}$ is a relation of arity k, and each ξ_j is a variable in V or one of the constants 0, 1. Note that the constants 0 and 1 are allowed in CNF($\mathcal{S}$)-formulas; this is equivalent to assuming that the set $\mathcal{S}$ contains the singleton logical relations $\{0\}$ and $\{1\}$. One could also consider CNF($\mathcal{S}$)-formulas without constants. In fact, this class of formulas has already been considered by Schaefer [31], as well as by other researchers in subsequent investigations. In particular, Schaefer [31] also established a dichotomy theorem for the complexity of the satisfiability problem for CNF($\mathcal{S}$)-formulas without constants. A *solution* of a CNF($\mathcal{S}$)-formula φ is an assignment $s = (a_1, \ldots, a_n)$ of Boolean values to the variables that makes every clause of φ true. A CNF($\mathcal{S}$)-formula is *satisfiable* if it has at least one solution.

The *satisfiability problem* SAT($\mathcal{S}$) associated with a finite set $\mathcal{S}$ of logical relations asks the following: Given a CNF($\mathcal{S}$)-formula φ, is it satisfiable? Many well-known restrictions of Boolean satisfiability, such as 3-SAT, NOT-ALL-EQUAL 3-SAT, and POSITIVE 1-IN-3 SAT, can be cast as SAT($\mathcal{S}$) problems, for a suitable choice of $\mathcal{S}$. For instance, let $R_0 = \{0,1\}^3 \backslash \{000\}$, $R_1 = \{0,1\}^3 \backslash \{100\}$, $R_2 = \{0,1\}^3 \backslash \{110\}$, $R_3 = \{0,1\}^3 \backslash \{111\}$. Then 3-SAT is the problem SAT($\{R_0, R_1, R_2, R_3\}$). Similarly, POSITIVE 1-IN-3SAT is SAT($\{R_{1/3}\}$), where $R_{1/3} = \{100, 010, 001\}$.

Schaefer [31] identified the complexity of *every* satisfiability problem SAT($\mathcal{S}$), where $\mathcal{S}$ ranges over all finite sets of logical relations. To state Schaefer's main result, we need to define some basic concepts.

DEFINITION 2.1. *Let R be a logical relation.*

1. *R is* bijunctive *if it is the set of solutions of a* 2-CNF*-formula.*
2. *R is* Horn *if it is the set of solutions of a Horn formula, where a Horn formula is a* CNF*-formula such that each conjunct has at most one positive literal.*
3. *R is* dual Horn *if it is the set of solutions of a dual Horn formula, where a dual Horn formula is a* CNF*-formula such that each conjunct has at most one negative literal.*
4. *R is* affine *if it is the set of solutions of a system of linear equations over $\mathbb{Z}_2$.*

Each of these types of logical relations can be characterized in terms of *closure* properties [31]. In what follows, we use boldface letters to denote vectors of Boolean values or vectors of variables. A relation R is bijunctive if and only if it is closed under the *majority* operation; this means that if $\mathbf{a}, \mathbf{b}, \mathbf{c} \in R$, then $\text{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c}) \in R$, where $\text{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ is the vector whose ith bit is the majority of a_i, b_i, c_i. A relation R is Horn if and only if it is closed under $\wedge$; this means that if $\mathbf{a}, \mathbf{b} \in R$, then $\mathbf{a} \wedge \mathbf{b} \in R$, where $\mathbf{a} \wedge \mathbf{b}$ is the vector whose ith bit is $a_i \wedge b_i$. Similarly, R is dual Horn if and only if it is closed under $\vee$. Finally, R is affine if and only if it is closed under $\mathbf{a} \oplus \mathbf{b} \oplus \mathbf{c}$.

DEFINITION 2.2. *A set $\mathcal{S}$ of logical relations is* Schaefer *if at least one of the following conditions holds:*

1. *Every relation in $\mathcal{S}$ is bijunctive.*
2. *Every relation in $\mathcal{S}$ is Horn.*
3. *Every relation in $\mathcal{S}$ is dual Horn.*
4. *Every relation in $\mathcal{S}$ is affine.*

A logical relation R is Schaefer *if the singleton $\{R\}$ is Schaefer.*

Since the property of a set being Schaefer is characterized in terms of closure under operations that are binary or ternary, it follows that there is a cubic algorithm to determine whether or not a given a finite set $\mathcal{S}$ of logical relations is Schaefer (assuming that each relation in $\mathcal{S}$ is given as a list of Boolean vectors).

THEOREM 2.3 (Schaefer's dichotomy theorem [31]). *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is Schaefer, then* SAT($\mathcal{S}$) *is in* P*; otherwise,* SAT($\mathcal{S}$) *is* NP*-complete.*

Theorem 2.3 is called a dichotomy theorem because Ladner [23] has shown that if P $\neq$ NP, then there are problems in NP that are neither in P nor NP-complete. Thus, Theorem 2.3 asserts that no SAT($\mathcal{S}$) problem is a problem of the kind discovered by Ladner. Note that the aforementioned characterization of Schaefer sets in terms of closure properties yields a cubic algorithm for determining, given a finite set $\mathcal{S}$ of logical relations, whether SAT($\mathcal{S}$) is in P or is NP-complete (here, the input size is the sum of the sizes of the relations in $\mathcal{S}$).

The more difficult part of the original proof of Schaefer's dichotomy theorem is to show that if $\mathcal{S}$ is not Schaefer, then SAT($\mathcal{S}$) is NP-complete. This is a consequence of a powerful result about the expressibility of logical relations. We say that a relation R is *expressible from* a set $\mathcal{S}$ of relations if there is a CNF($\mathcal{S}$)-formula $\varphi(\mathbf{x}, \mathbf{y})$ such that $R = \{\mathbf{a} |\ \exists \mathbf{y} \varphi(\mathbf{a}, \mathbf{y})\}$.

THEOREM 2.4 (Schaefer's expressibility theorem [31]). *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is not Schaefer, then every logical relation is expressible from $\mathcal{S}$.*

In this paper, we are interested in the connectivity properties of the space of solutions of CNF($\mathcal{S}$)-formulas. If φ is a CNF($\mathcal{S}$)-formula with n variables, then the *solution graph* $G(\varphi)$ of φ denotes the subgraph of the n-dimensional hypercube induced by the solutions of φ. This means that the vertices of $G(\varphi)$ are the solutions of φ, and there is an edge between two solutions of $G(\varphi)$ precisely when they differ in exactly one variable.

We consider the following two algorithmic problems for CNF($\mathcal{S}$)-formulas.

Problem 1. *The Connectivity Problem* CONN($\mathcal{S}$). Given a CNF($\mathcal{S}$)-formula φ, is $G(\varphi)$ connected? (If φ is unsatisfiable, then the answer to this problem is "yes.")

Problem 2. *The st-Connectivity Problem* ST-CONN($\mathcal{S}$). Given a CNF($\mathcal{S}$)-formula φ and two solutions $\mathbf{s}$ and $\mathbf{t}$ of φ, is there a path from $\mathbf{s}$ to $\mathbf{t}$ in $G(\varphi)$?

To pinpoint the computational complexity of CONN($\mathcal{S}$) and ST-CONN($\mathcal{S}$), we need to introduce certain new types of relations.

DEFINITION 2.5. *Let $R \subseteq \{0,1\}^k$ be a logical relation.*

1. *R is* componentwise bijunctive *if every connected component of the graph $G(R)$ is a bijunctive relation.*
2. *R is* OR-free *if the relation* $\mathrm{OR} = \{01, 10, 11\}$ *cannot be obtained from R by setting $k-2$ of the coordinates of R to a constant $\mathbf{c} \in \{0,1\}^{k-2}$. In other words, R is* OR*-free if $(x_1 \vee x_2)$ is not definable from R by fixing $k-2$ variables.*
3. *R is* NAND-free *if the relation* $\mathrm{NAND} = \{00, 01, 10\}$ *cannot be obtained from R by setting $k-2$ of the coordinates of R to a constant $\mathbf{c} \in \{0,1\}^{k-2}$. In other words, R is* NAND*-free if $(\bar{x}_1 \vee \bar{x}_2)$ is not definable from R by fixing $k-2$ variables.*

We are now ready to introduce the key concept of a *tight* set of relations.

DEFINITION 2.6. *A set $\mathcal{S}$ of logical relations is* tight *if at least one of the following three conditions holds:*

1. *Every relation in $\mathcal{S}$ is componentwise bijunctive.*
2. *Every relation in $\mathcal{S}$ is* OR*-free.*
3. *Every relation in $\mathcal{S}$ is* NAND*-free.*

A logical relation R is tight *if the singleton $\{R\}$ is tight.*

In section 4, we show that if $\mathcal{S}$ is Schaefer, then it is tight. Moreover, we show that the converse does not hold. It is also easy to see that there is a polynomial-time algorithm (in fact, a cubic algorithm) for testing whether a given relation is tight.

Just as Schaefer's dichotomy theorem follows from an expressibility statement, our dichotomy theorems are derived from the following theorem, which we will call the structural expressibility theorem. The precise definition of the concept of *structural expressibility* is given in section 3. Intuitively, this concept strengthens the concept of expressibility with the requirement that the space of the witnesses to the existentially quantified variables has certain strong connectivity properties.

THEOREM 2.7 (structural expressibility theorem). *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is not tight, then every logical relation is structurally expressible from $\mathcal{S}$.*

Using the structural expressibility theorem, we obtain the following dichotomy theorems for the computational complexity of $\textsc{Conn}(\mathcal{S})$ and $\textsc{st-Conn}(\mathcal{S})$.

THEOREM 2.8. *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is tight, then* $\textsc{Conn}(\mathcal{S})$ *is in* coNP*; otherwise, it is* PSPACE*-complete.*

THEOREM 2.9. *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is tight, then* $\textsc{st-Conn}(\mathcal{S})$ *is in* P*; otherwise,* $\textsc{st-Conn}(\mathcal{S})$ *is* PSPACE*-complete.*

We also show that if $\mathcal{S}$ is tight, but not Schaefer, then $\textsc{Conn}(\mathcal{S})$ is coNP-complete.

Example 1. The set $\mathcal{S} = \{R_{1/3}\}$, where $R_{1/3} = \{100, 010, 001\}$, is tight (actually, it is componentwise bijunctive), but not Schaefer. It follows that $\textsc{Sat}(\mathcal{S})$ is NP-complete (recall that this problem is POSITIVE 1-IN-3 SAT), $\textsc{st-Conn}(\mathcal{S})$ is in P, and $\textsc{Conn}(\mathcal{S})$ is coNP-complete.

Example 2. The set $\mathcal{S} = \{R_{\mathrm{NAE}}\}$, where $R_{\mathrm{NAE}} = \{0,1\}^3 \setminus \{000, 111\}$, is not tight; hence $\textsc{Sat}(\mathcal{S})$ is NP-complete (this problem is POSITIVE NOT-ALL-EQUAL 3-SAT), while both $\textsc{st-Conn}(\mathcal{S})$ and $\textsc{Conn}(\mathcal{S})$ are PSPACE-complete.

Example 3. The set $\mathcal{S} = \{R_V\}$, where $R_V = \{110, 100, 000, 001, 011\}$, is not tight. It is OR-free, but not NAND-free or componentwise bijunctive, and it is not Schaefer. Hence $\textsc{Sat}(\mathcal{S})$ is NP-complete, $\textsc{st-Conn}(\mathcal{S})$ is in P, and $\textsc{Conn}(\mathcal{S})$ is in coNP.

Remark. The last example illustrates that OR-free (NAND-free) relations cannot be thought of as componentwise Horn (dual Horn).

The dichotomy in the computational complexity of CONN($\mathcal{S}$) and ST-CONN($\mathcal{S}$) is accompanied by a parallel structural dichotomy in the size of the diameter of $G(\varphi)$ (where, for a CNF($\mathcal{S}$)-formula φ, the *diameter of* $G(\varphi)$ is the maximum of the diameters of the components of $G(\varphi)$).

THEOREM 2.10. *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is tight, then for every* CNF($\mathcal{S}$)*-formula φ, the diameter of $G(\varphi)$ is linear in the number of variables of φ; otherwise, there are* CNF($\mathcal{S}$)*-formulas φ such that the diameter of $G(\varphi)$ is exponential in the number of variables of φ.*

Our results and their comparison to Schaefer's dichotomy theorem are summarized in the table below.

$\mathcal{S}$	SAT($\mathcal{S}$)	ST-CONN($\mathcal{S}$)	CONN($\mathcal{S}$)	Diameter
Schaefer	P	P	coNP	$O(n)$
Tight, non-Schaefer	NP-complete	P	coNP-complete	$O(n)$
Nontight	NP-complete	PSPACE-complete	PSPACE-complete	$2^{\Omega(\sqrt{n})}$

We conjecture that the complexity of CONN($\mathcal{S}$) exhibits a *trichotomy*, that is, for every finite set $\mathcal{S}$ of logical relations, one of the following holds:

1. CONN($\mathcal{S}$) is in P.
2. CONN($\mathcal{S}$) is coNP-complete.
3. CONN($\mathcal{S}$) is PSPACE-complete.

As mentioned above, we will show that if $\mathcal{S}$ is tight but not Schaefer, then CONN($\mathcal{S}$) is coNP-complete. We will also show that if $\mathcal{S}$ is bijunctive or affine, then CONN($\mathcal{S}$) is in P. Hence, to settle the above conjecture, it remains to pinpoint the complexity of CONN($\mathcal{S}$) whenever $\mathcal{S}$ is Horn and whenever $\mathcal{S}$ is dual Horn. In the conference version [16] of the present paper, we further conjectured that if $\mathcal{S}$ is Horn or dual Horn, then CONN($\mathcal{S}$) is in P. In other words, we conjectured that if $\mathcal{S}$ is Schaefer, then CONN($\mathcal{S}$) is in P. This second conjecture, however, was subsequently disproved by Makino, Tamaki, and Yamamoto [24], who discovered a particular Horn set $\mathcal{S}$ such that CONN($\mathcal{S}$) is coNP-complete. Here, we go beyond the results obtained in the conference version of the present paper and identify additional conditions on a Horn set $\mathcal{S}$ implying that CONN($\mathcal{S}$) is in P. These new results suggest a natural dichotomy within Schaefer sets of relations and, thus, provide evidence for the trichotomy conjecture.

The remainder of this paper is organized as follows. In section 3, we prove the structural expressibility theorem, establish the hard side of the dichotomies for CONN($\mathcal{S}$) and for ST-CONN($\mathcal{S}$), and contrast our result to Schaefer's expressibility and dichotomy theorems. In section 4, we describe the easy side of the dichotomy—the polynomial-time algorithms and the structural properties for tight sets of relations. In addition, we obtain partial results towards the trichotomy conjecture for CONN($\mathcal{S}$).

3. The hard case of the dichotomy: Nontight sets of relations. In this section, we address the *hard* side of the dichotomy, where we deal with the more computationally intractable cases. This is also the harder part of our proof. We define the notion of structural expressibility in section 3.1 and prove the structural expressibility theorem in section 3.2. This theorem implies that for all nontight sets $\mathcal{S}$ and $\mathcal{S}'$, the connectivity problems CONN($\mathcal{S}$) and CONN($\mathcal{S}'$) are polynomial-time equivalent; moreover, the same holds for the connectivity problems ST-CONN($\mathcal{S}$) and ST-CONN($\mathcal{S}'$). In addition, the diameters of the solution graphs of CNF($\mathcal{S}$)-formulas and CNF($\mathcal{S}'$)-formulas are also related polynomially. In section 3.3, we prove that for 3-CNF-formulas the connectivity problems are PSPACE-complete, and the diameter can be exponential. This fact combined with the structural expressibility

theorem yields the hard side of all of our dichotomy results, as well as the exponential size of the diameter.

We will use $\mathbf{a}, \mathbf{b}, \ldots$ to denote Boolean vectors, and $\mathbf{x}$ and $\mathbf{y}$ to denote vectors of variables. We write $|\mathbf{a}|$ to denote the Hamming weight (number of 1's) of a Boolean vector $\mathbf{a}$. Given two Boolean vectors $\mathbf{a}$ and $\mathbf{b}$, we write $|\mathbf{a}-\mathbf{b}|$ to denote the Hamming distance between $\mathbf{a}$ and $\mathbf{b}$. Finally, if $\mathbf{a}$ and $\mathbf{b}$ are solutions of a Boolean formula φ and lie in the same component of $G(\varphi)$, then we write $d_\varphi(\mathbf{a}, \mathbf{b})$ to denote the shortest-path distance between $\mathbf{a}$ and $\mathbf{b}$ in $G(\varphi)$.

3.1. Structural expressibility. As stated in the previous section, in his dichotomy theorem, Schaefer [31] used the following notion of expressibility: a relation R is *expressible from* a set $\mathcal{S}$ of relations if there is a CNF($\mathcal{S}$)-formula φ so that $R = \{\mathbf{a}|\ \exists \mathbf{y} \varphi(\mathbf{a}, \mathbf{y})\}$. This notion is not sufficient for our purposes. Instead, we introduce a more delicate notion which we call *structural expressibility*. Intuitively, we view the relation R as a subgraph of the hypercube, rather than just a subset, and require that this graph structure also be captured by the formula φ.

DEFINITION 3.1. *A relation R is* structurally expressible *from a set of relations $\mathcal{S}$ if there is a* CNF($\mathcal{S}$)*-formula φ such that the following conditions hold:*

1. $R = \{\mathbf{a}|\ \exists \mathbf{y} \varphi(\mathbf{a}, \mathbf{y})\}$.
2. *For every $\mathbf{a} \in R$, the graph $G(\varphi(\mathbf{a}, \mathbf{y}))$ is connected.*
3. *For $\mathbf{a}, \mathbf{b} \in R$ with $|\mathbf{a} - \mathbf{b}| = 1$, there exists $\mathbf{w}$ such that $(\mathbf{a}, \mathbf{w})$ and $(\mathbf{b}, \mathbf{w})$ are solutions of φ.*

For $\mathbf{a} \in R$, the *witnesses* of $\mathbf{a}$ are the $\mathbf{y}$'s such that $\varphi(\mathbf{a}, \mathbf{y})$ is true. The last two conditions say that the witnesses of $\mathbf{a} \in R$ are connected, and that neighboring $\mathbf{a}, \mathbf{b} \in R$ have a common witness. This allows us to simulate an edge $(\mathbf{a}, \mathbf{b})$ in $G(R)$ by a path in $G(\varphi)$, and thus relate the connectivity properties of the solution spaces. There is, however, a price to pay: it is much harder to come up with formulas that structurally express a relation R. An example is when $\mathcal{S}$ is the set of all paths of length 4 in $\{0,1\}^3$, a set that plays a crucial role in our proof. While 3-SAT relations are easily expressible from $\mathcal{S}$ in Schaefer's sense, the CNF($\mathcal{S}$)-formulas that structurally express 3-SAT relations are fairly complicated and have a large witness space.

An example of the difference between a structural and a nonstructural expression is shown in Figure 3.1. Consider the logical relation given by the formula $(x_1 \vee x_2 \vee x_3)$; the graph of this logical relation is depicted in Figure 3.1(a). Consider also the NOT-ALL-EQUAL relation $R_{\text{NAE}} = \{0,1\}^3 \setminus \{000, 111\}$. Figure 3.1(b) depicts the graph of the expression $\varphi(x_1, x_2, x_3, y_1, y_2) = R_{\text{NAE}}(x_1, x_2, y_1) \wedge R_{\text{NAE}}(x_2, x_3, y_2) \wedge R_{\text{NAE}}(y_1, y_2, 1)$. This is a structural expression because $(x_1 \vee x_2 \vee x_3) \equiv \exists y_1, y_2 \varphi(x_1, x_2, x_3, y_1, y_2)$ and connectivity is preserved. Finally, Figure 3.1(c) depicts the graph of the expression $\psi(x_1, x_2, x_3, y_1) = R_{\text{NAE}}(x_1, x_2, y_1) \wedge R_{\text{NAE}}(\bar{y}_1, x_3, 0) \wedge R_{\text{NAE}}(y_1, x_2, 1)$. Even though $(x_1 \vee x_2 \vee x_3) \equiv \exists y_1 \psi(x_1, x_2, x_3, y_1)$, this is *not* a structural expression because connectivity is not preserved.

LEMMA 3.2. *Let $\mathcal{S}$ and $\mathcal{S}'$ be sets of relations such that every $R \in \mathcal{S}'$ is structurally expressible from $\mathcal{S}$, and, moreover, there is a polynomial-time algorithm that produces a structural expression from $\mathcal{S}$ for every $R \in \mathcal{S}'$. Given a* CNF($\mathcal{S}'$)*-formula $\psi(\mathbf{x})$, one can efficiently construct a* CNF($\mathcal{S}$)*-formula $\varphi(\mathbf{x}, \mathbf{y})$ such that*

1. *$\psi(\mathbf{x}) \equiv \exists \mathbf{y}\ \varphi(\mathbf{x}, \mathbf{y})$;*
2. *if $(\mathbf{s}, \mathbf{w^s}), (\mathbf{t}, \mathbf{w^t}) \in \varphi$ are connected in $G(\varphi)$ by a path of length d, then there is a path from $\mathbf{s}$ to $\mathbf{t}$ in $G(\psi)$ of length at most d;*
3. *if $\mathbf{s}, \mathbf{t} \in \psi$ are connected in $G(\psi)$, then for every witness $\mathbf{w^s}$ of $\mathbf{s}$, and every witness $\mathbf{w^t}$ of $\mathbf{t}$, there is a path from $(\mathbf{s}, \mathbf{w^s})$ to $(\mathbf{t}, \mathbf{w^t})$ in $G(\varphi)$.*

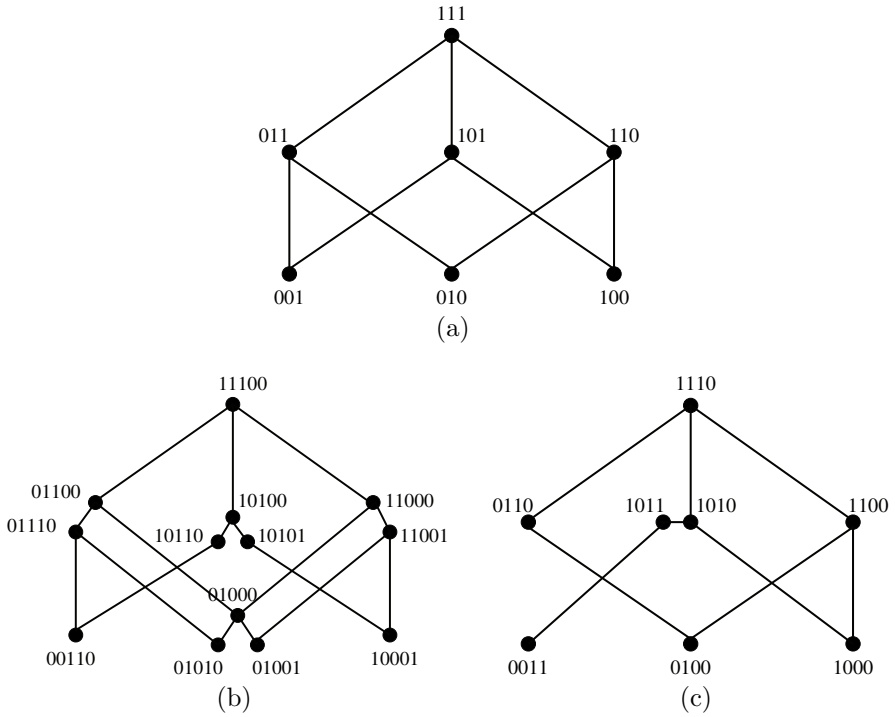

FIG. 3.1. *Expressing the relation* $(x_1 \vee x_2 \vee x_3)$ *from the* R_{NAE} *relation.*

Proof. Suppose ψ is a formula on n variables that consists of m clauses $C_1, \ldots, C_m$. For clause C_j, assume that the set of variables is $V_j \subseteq [n] = \{1, \ldots, n\}$, and that it involves relation $R_j \in \mathcal{S}$. Thus, $\psi(\mathbf{x})$ is $\wedge_{j=1}^{m} R_j(\mathbf{x}_{V_j})$. Let φ_j be the structural expression for R_j from $\mathcal{S}'$, so that $R_j(\mathbf{x}_{V_j}) \equiv \exists \mathbf{y}_j \ \varphi_j(\mathbf{x}_{V_j}, \mathbf{y}_j)$. Let $\mathbf{y}$ be the vector $(\mathbf{y}_1, \ldots, \mathbf{y}_m)$ and let $\varphi(\mathbf{x}, \mathbf{y})$ be the formula $\wedge_{j=1}^{m} \varphi_j(\mathbf{x}_{V_j}, \mathbf{y}_j)$. Then $\psi(\mathbf{x}) \equiv \exists \mathbf{y}\ \varphi(\mathbf{x}, \mathbf{y})$.

Statement 2 follows from 1 by projection of the path on the coordinates of $\mathbf{x}$. For statement 3, consider $\mathbf{s}, \mathbf{t} \in \psi$ that are connected in $G(\psi)$ via a path $\mathbf{s} = \mathbf{u^0} \to \mathbf{u^1} \to \cdots \to \mathbf{u^r} = \mathbf{t}$. For every $\mathbf{u^i}, \mathbf{u^{i+1}}$, and clause C_j, there exists an assignment $\mathbf{w^i}_j$ to $\mathbf{y}_j$ such that both $(\mathbf{u^i}_{V_j}, \mathbf{w^i}_j)$ and $(\mathbf{u^{i+1}}_{V_j}, \mathbf{w^i}_j)$ are solutions of φ_j, by condition 2 of structural expressibility. Thus $(\mathbf{u^i}, \mathbf{w^i})$ and $(\mathbf{u^{i+1}}, \mathbf{w^i})$ are both solutions of φ, where $\mathbf{w^i} = (\mathbf{w^i}_1, \ldots, \mathbf{w^i}_m)$. Further, for every $\mathbf{u^i}$, the space of solutions of $\varphi(\mathbf{u^i}, \mathbf{y})$ is the product space of the solutions of $\varphi_j(\mathbf{u^i}_{V_j}, \mathbf{y}_j)$ over $j = 1, \ldots, m$. Since these are all connected by condition 3 of structural expressibility, $G(\varphi(\mathbf{u^i}, \mathbf{y}))$ is connected. The following describes a path from $(\mathbf{s}, \mathbf{w^s})$ to $(\mathbf{t}, \mathbf{w^t})$ in $G(\varphi)$: $(\mathbf{s}, \mathbf{w^s}) \rightsquigarrow (\mathbf{s}, \mathbf{w^0}) \to (\mathbf{u^1}, \mathbf{w^0}) \rightsquigarrow (\mathbf{u^1}, \mathbf{w^1}) \to \cdots \rightsquigarrow (\mathbf{u^{r-1}}, \mathbf{w^{r-1}}) \to (\mathbf{t}, \mathbf{w^{r-1}}) \rightsquigarrow (\mathbf{t}, \mathbf{w^t})$. Here $\rightsquigarrow$ indicates a path in $G(\varphi(\mathbf{u^i}, \mathbf{y}))$. □

COROLLARY 3.3. *Suppose* $\mathcal{S}$ *and* $\mathcal{S}'$ *are sets of relations such that every* $R \in \mathcal{S}'$ *is structurally expressible from* $\mathcal{S}$*, and, moreover, there is a polynomial-time algorithm that produces a structural expression from* $\mathcal{S}$ *for every* $R \in \mathcal{S}'$*.*

1. *There are polynomial-time reductions from* CONN($\mathcal{S}'$) *to* CONN($\mathcal{S}$)*, and from* ST-CONN($\mathcal{S}'$) *to* ST-CONN($\mathcal{S}$)*.*
2. *If there exists a* CNF($\mathcal{S}'$)*-formula* $\psi(\mathbf{x})$ *with* n *variables,* m *clauses, and diameter* d*, then there exists a* CNF($\mathcal{S}$)*-formula* $\varphi(\mathbf{x}, \mathbf{y})$*, where* $\mathbf{y}$ *is a vector of* $O(m)$ *variables, such that the diameter of* $G(\phi)$ *is at least* d*.*

3.2. The structural expressibility theorem. In this subsection, we prove the structural expressibility theorem. The main step in the proof is Lemma 3.4, which shows that if $\mathcal{S}$ is not tight, then we can structurally express the 3-clause relations from the relations in $\mathcal{S}$. If $k \geq 2$, then a *k-clause* is a disjunction of k variables or negated variables. For $0 \leq i \leq k$, let D_i be the set of all satisfying truth assignments of the k-clause whose first i literals are negated, and let $\mathcal{S}_k = \{D_0, D_1, \ldots, D_k\}$. Thus, $\mathrm{CNF}(\mathcal{S}_k)$ is the collection of k-CNF-formulas.

LEMMA 3.4. *If set $\mathcal{S}$ of relations is not tight, $\mathcal{S}_3$ is structurally expressible from $\mathcal{S}$.*

Proof. First, observe that all 2-clauses are structurally expressible from $\mathcal{S}$. There exists $R \in \mathcal{S}$ which is not OR-free, so we can express $(x_1 \vee x_2)$ by substituting constants in R. Similarly, we can express $(\bar{x}_1 \vee \bar{x}_2)$ using a relation that is not NAND-free. The last 2-clause $(x_1 \vee \bar{x}_2)$ can be obtained from OR and NAND by a technique that corresponds to reverse resolution. $(x_1 \vee \bar{x}_2) = \exists y\ (x_1 \vee y) \wedge (\bar{y} \vee \bar{x}_2)$. It is easy to see that this gives a structural expression. From here onwards we assume that $\mathcal{S}$ contains all 2-clauses. The proof now proceeds in four steps. First, we will express a relation in which there exist two elements that are at graph distance larger than their Hamming distance. Second, we will express a relation that is just a single path between such elements. Third, we will express a relation which is a path of length 4 between elements at Hamming distance 2. Finally, we will express the 3-clauses.

Step 1. *Structurally expressing a relation in which some distance expands.* For a relation R, we say that the distance between $\mathbf{a}$ and $\mathbf{b}$ *expands* if $\mathbf{a}$ and $\mathbf{b}$ are connected in $G(R)$, but $d_R(\mathbf{a}, \mathbf{b}) > |\mathbf{a} - \mathbf{b}|$. In section 4.2, Lemma 4.3, we will show that no distance expands in componentwise bijunctive relations. The same also holds true for the relation $R_{\mathrm{NAE}} = \{0,1\}^3 \setminus \{000, 111\}$, which is not componentwise bijunctive. Nonetheless, we show here that if R is not componentwise bijunctive, then, by adding 2-clauses, we can structurally express a relation Q in which some distance expands. For instance, when $R = R_{\mathrm{NAE}}$, then we can take $Q(x_1, x_2, x_3) = R_{\mathrm{NAE}}(x_1, x_2, x_3) \wedge (\bar{x_1} \vee \bar{x}_3)$, as shown in Figure 3.2. The distance between $\mathbf{a} = 100$ and $\mathbf{b} = 001$ in Q expands. Similarly, in the general construction, we identify $\mathbf{a}$ and $\mathbf{b}$ on a cycle, and add 2-clauses that eliminate all the vertices along the shorter arc between $\mathbf{a}$ and $\mathbf{b}$.

Since $\mathcal{S}$ is not tight, it contains a relation R which is not componentwise bijunctive. If R contains $\mathbf{a}, \mathbf{b}$ where the distance between them expands, we are done. So assume that for all $\mathbf{a}, \mathbf{b} \in G(R)$, $d_R(\mathbf{a}, \mathbf{b}) = |\mathbf{a} - \mathbf{b}|$. Since R is not componentwise bijunctive, there exists a triple of assignments $\mathbf{a}, \mathbf{b}, \mathbf{c}$ lying in the same component such that $\mathrm{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ is not in that component. Choose the triple such that the sum of pairwise distances $d_R(\mathbf{a}, \mathbf{b}) + d_R(\mathbf{b}, \mathbf{c}) + d_R(\mathbf{c}, \mathbf{a})$ is minimized. Let $U = \{i | a_i \neq b_i\}$, $V = \{i | b_i \neq c_i\}$, and $W = \{i | c_i \neq a_i\}$. Since $d_R(\mathbf{a}, \mathbf{b}) = |\mathbf{a} - \mathbf{b}|$, a shortest path does not flip variables outside of U, and each variable in U is flipped exactly once. The same holds for V and W. We note some useful properties of the sets U, V, W:

1. *Every index $i \in U \cup V \cup W$ occurs in exactly two of U, V, W.* Consider going by a shortest path from $\mathbf{a}$ to $\mathbf{b}$ to $\mathbf{c}$ and back to $\mathbf{a}$. Every $i \in U \cup V \cup W$ is seen an even number of times along this path since we return to $\mathbf{a}$. It is seen at least once, and at most thrice, so in fact it occurs twice.
2. *Every pairwise intersection $U \cap V, V \cap W$, and $W \cap U$ is nonempty.* Suppose the sets U and V are disjoint. From property 1, we must have $W = U \cup V$. But then it is easy to see that $\mathrm{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c}) = \mathbf{b}$. This contradicts the choice of $\mathbf{a}, \mathbf{b}, \mathbf{c}$.

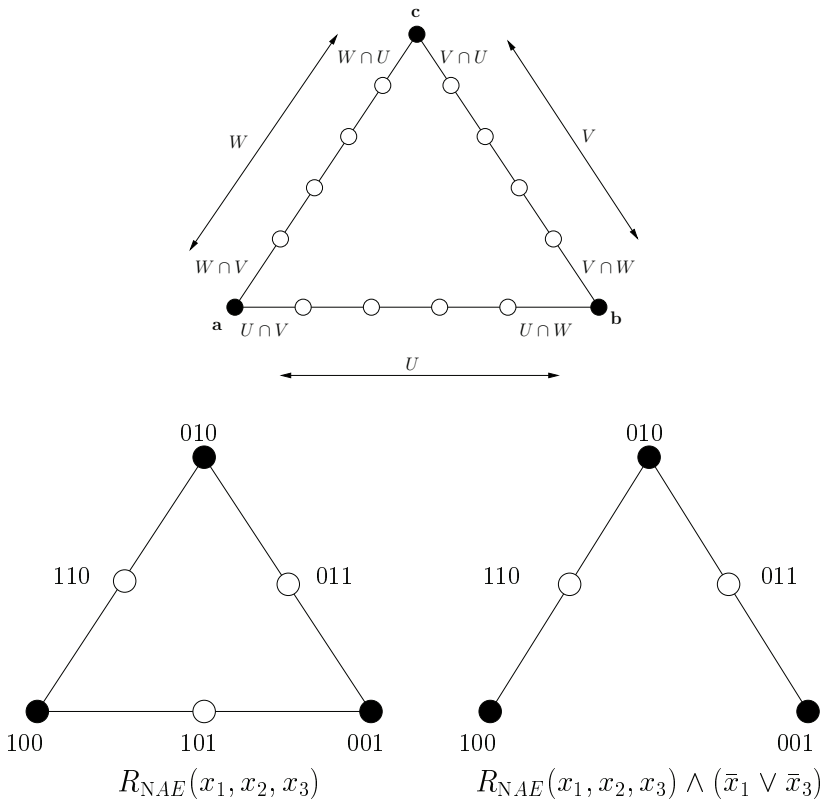

FIG. 3.2. *Step* 1 *of the proof of Lemma* 3.4*, and an example.*

3. *The sets* $U \cap V$ *and* $U \cap W$ *partition the set* U. By property 1, each index of U occurs in one of V and W as well. Also since no index occurs in all three sets U, V, W this is in fact a disjoint partition.
4. *For each index* $i \in U \cap W$*, it holds that* $\mathbf{a} \oplus \mathbf{e}_i \notin R$. Assume for the sake of contradiction that $\mathbf{a}' = \mathbf{a} \oplus \mathbf{e}_i \in R$. Since $i \in U \cap W$ we have simultaneously moved closer to both $\mathbf{b}$ and $\mathbf{c}$. Hence we have $d_R(\mathbf{a}', \mathbf{b}) + d_R(\mathbf{b}, \mathbf{c}) + d_R(\mathbf{c}, \mathbf{a}') < d_R(\mathbf{a}, \mathbf{b}) + d_R(\mathbf{b}, \mathbf{c}) + d_R(\mathbf{c}, \mathbf{a})$. Also $\mathrm{maj}(\mathbf{a}', \mathbf{b}, \mathbf{c}) = \mathrm{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c})$. But this contradicts our choice of $\mathbf{a}, \mathbf{b}, \mathbf{c}$.

Property 4 implies that the shortest paths to $\mathbf{b}$ and $\mathbf{c}$ diverge at $\mathbf{a}$, since for any shortest path to $\mathbf{b}$ the first variable flipped is from $U \cap V$, whereas for a shortest path to $\mathbf{c}$ it is from $W \cap V$. Similar statements hold for the vertices $\mathbf{b}$ and $\mathbf{c}$. Thus along the shortest path from $\mathbf{a}$ to $\mathbf{b}$ the first bit flipped is from $U \cap V$ and the last bit flipped is from $U \cap W$. On the other hand, if we go from $\mathbf{a}$ to $\mathbf{c}$ and then to $\mathbf{b}$, all the bits from $U \cap W$ are flipped before the bits from $U \cap V$. We use this crucially to define Q. We will add a set of 2-clauses that enforce the following rule on paths starting at $\mathbf{a}$: *Flip variables from* $U \cap W$ *before variables from* $U \cap V$. This will eliminate all shortest paths from $\mathbf{a}$ to $\mathbf{b}$ since they begin by flipping a variable in $U \cap V$ and end with $U \cap W$. The paths from $\mathbf{a}$ to $\mathbf{b}$ via $\mathbf{c}$ survive since they flip $U \cap W$ while going from $\mathbf{a}$ to $\mathbf{c}$ and $U \cap V$ while going from $\mathbf{c}$ to $\mathbf{b}$. However, all remaining paths have length at least $|\mathbf{a} - \mathbf{b}| + 2$ since they flip twice some variables not in U.

Take all pairs of indices $\{(i, j) | i \in U \cap W, j \in U \cap V\}$. The following conditions

hold from the definition of U, V, W: $a_i = \bar{c}_i = \bar{b}_i$ and $a_j = c_j = \bar{b}_j$. Add the 2-clause C_{ij} asserting that the pair of variables $x_i x_j$ must take values in $\{a_i a_j, c_i c_j, b_i b_j\} = \{a_i a_j, \bar{a}_i a_j, \bar{a}_i \bar{a}_j\}$. The new relation is $Q = R \wedge_{i,j} C_{ij}$. Note that $Q \subset R$. We verify that the distance between $\mathbf{a}$ and $\mathbf{b}$ in Q expands. It is easy to see that for any $j \in U$, the assignment $\mathbf{a} \oplus \mathbf{e}_j \notin Q$. Hence there are no shortest paths left from $\mathbf{a}$ to $\mathbf{b}$. On the other hand, it is easy to see that $\mathbf{a}$ and $\mathbf{b}$ are still connected, since the vertex $\mathbf{c}$ is still reachable from both.

Step 2. *Isolating a pair of assignments whose distance expands.* The relation Q obtained in Step 1 may have several disconnected components. This *cleanup* step isolates a single pair of assignments whose distance expands. By adding 2-clauses, we show that one can express a path of length $r+2$ between assignments at distance r.

Take $\mathbf{a}, \mathbf{b} \in Q$ whose distance expands in Q and $d_Q(\mathbf{a}, \mathbf{b})$ is minimized. Let $U = \{i | a_i \neq b_i\}$ and $|U| = r$. Shortest paths between $\mathbf{a}$ and $\mathbf{b}$ have certain useful properties:

1. *Each shortest path flips every variable from U exactly once.* Observe that each index $j \in U$ is flipped an odd number of times along any path from $\mathbf{a}$ to $\mathbf{b}$. Suppose it is flipped thrice along a shortest path. Starting at $\mathbf{a}$ and going along this path, let $\mathbf{b}'$ be the assignment reached after flipping j twice. Then the distance between $\mathbf{a}$ and $\mathbf{b}'$ expands, since j is flipped twice along a shortest path between them in Q. Also $d_Q(\mathbf{a}, \mathbf{b}') < d_Q(\mathbf{a}, \mathbf{b})$, contradicting the choice of $\mathbf{a}$ and $\mathbf{b}$.
2. *Every shortest path flips exactly one variable $i \notin U$.* Since the distance between $\mathbf{a}$ and $\mathbf{b}$ expands, every shortest path must flip some variable $i \notin U$. Suppose it flips more than one such variable. Since $\mathbf{a}$ and $\mathbf{b}$ agree on these variables, each of them is flipped an even number of times. Let i be the first variable to be flipped twice. Let $\mathbf{b}'$ be the assignment reached after flipping i the second time. It is easy to verify that the distance between $\mathbf{a}$ and $\mathbf{b}'$ also expands, but $d_Q(\mathbf{a}, \mathbf{b}') < d_Q(\mathbf{a}, \mathbf{b})$.
3. *The variable $i \notin U$ is the first and last variable to be flipped along the path.* Assume the first variable flipped is not i. Let $\mathbf{a}'$ be the assignment reached along the path before we flip i the first time. Then $d_Q(\mathbf{a}', \mathbf{b}) < d_Q(\mathbf{a}, \mathbf{b})$. The distance between $\mathbf{a}'$ and $\mathbf{b}$ expands since the shortest path between them flips the variables i twice. This contradicts the choice of $\mathbf{a}$ and $\mathbf{b}$. Assume $j \in U$ is flipped twice. Then as before we get a pair $\mathbf{a}', \mathbf{b}'$ that contradict the choice of $\mathbf{a}, \mathbf{b}$.

Every shortest path between $\mathbf{a}$ and $\mathbf{b}$ has the following structure: first a variable $i \notin U$ is flipped to $\bar{a}_i$, then the variables from U are flipped in some order, and finally the variable i is flipped back to a_i.

Different shortest paths may vary in the choice of $i \notin U$ in the first step and in the order in which the variables from U are flipped. Fix one such path $T \subseteq Q$. Assume that $U = \{1, \ldots, r\}$ and the variables are flipped in this order, and the additional variable flipped twice is $r+1$. Denote the path by $\mathbf{a} \to \mathbf{u^0} \to \mathbf{u^1} \to \cdots \to \mathbf{u^r} \to \mathbf{b}$. Next we prove that we cannot flip the $r+1$th variable at an intermediate vertex along the path.

4. *For $1 \leq j \leq r-1$ the assignment $\mathbf{u^j} \oplus \mathbf{e_{r+1}} \notin Q$.* Suppose that for some j we have $\mathbf{c} = \mathbf{u^j} \oplus \mathbf{e_{r+1}} \in Q$. Then $\mathbf{c}$ differs from $\mathbf{a}$ on $\{1, \ldots, i\}$ and from $\mathbf{b}$ on $\{i+1, \ldots, r\}$. The distance from $\mathbf{c}$ to at least one of $\mathbf{a}$ or $\mathbf{b}$ must expand, or else we get a path from $\mathbf{a}$ to $\mathbf{b}$ through $\mathbf{c}$ of length $|\mathbf{a}-\mathbf{b}|$, which contradicts the fact that this distance expands. However, $d_Q(\mathbf{a}, \mathbf{c})$ and $d_Q(\mathbf{b}, \mathbf{c})$ are strictly less than $d_Q(\mathbf{a}, \mathbf{b})$, so we get a contradiction to the choice of $\mathbf{a}, \mathbf{b}$.

We now construct the path of length $r+2$. For all $i \geq r+2$ we set $x_i = a_i$ to get a relation on $r+1$ variables. Note that $\mathbf{b} = \bar{a}_1 \dots \bar{a}_r a_{r+1}$. Take $i < j \in U$. Along the path T the variable i is flipped before j so the variables $x_i x_j$ take one of three values $\{a_i a_j, \bar{a}_i a_j, \bar{a}_i \bar{a}_j\}$. So we add a 2-clause C_{ij} that requires $x_i x_j$ to take one of these values and take $T = Q \wedge_{i,j} C_{ij}$. Clearly, every assignment along the path lies in T. We claim that these are the only solutions. To show this, take an arbitrary assignment $\mathbf{c}$ satisfying the added constraints. If for some $i < j \leq r$ we have $c_i = a_i$ but $c_j = \bar{a}_j$, this would violate C_{ij}. Hence the first r variables of $\mathbf{c}$ are of the form $\bar{a}_1 \dots \bar{a}_i a_{i+1} \dots a_r$ for $0 \leq i \leq r$. If $c_{r+1} = \bar{a}_{r+1}$, then $\mathbf{c} = \mathbf{u}^i$. If $c_{r+1} = a_{r+1}$, then $\mathbf{c} = \mathbf{u}^i \oplus \mathbf{e}_{r+1}$. By property 4 above, such a vector satisfies Q if and only if $i = 0$ or $i = r$, which correspond to $\mathbf{c} = \mathbf{a}$ and $\mathbf{c} = \mathbf{b}$, respectively.

Step 3. *Structurally expressing paths of length* 4. Let $\mathcal{P}$ denote the set of all ternary relations whose graph is a path of length 4 between two assignments at Hamming distance 2. Up to permutations of coordinates, there are 6 such relations. Each of them is the conjunction of a 3-clause and a 2-clause. For instance, the relation $M = \{100, 110, 010, 011, 001\}$ can be written as $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3)$. (It is named so because its graph looks like the letter M on the cube.) These relations are "minimal" examples of relations that are not componentwise bijunctive. By projecting out intermediate variables from the path T obtained in Step 2, we structurally express one of the relations in $\mathcal{P}$. We structurally express other relations in $\mathcal{P}$ using this relation.

We will write all relations in $\mathcal{P}$ in terms of $M(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3)$ by negating variables. For example, $M(\bar{x}_1, x_2, x_3) = (\bar{x}_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_3) = \{000, 010, 110, 111, 101\}$.

Define the relation $P(x_1, x_{r+1}, x_2) = \exists x_3 \dots x_r \; T(x_1, \dots, x_{r+1})$. The table below, listing all tuples in P and their witnesses, shows that the conditions for structural expressibility are satisfied, and $P \in \mathcal{P}$.

x_1, x_2, x_{r+1}	$x_3, \dots, x_r$
$a_1 a_2 a_{r+1}$	$a_3 \dots a_r$
$a_1 a_2 \bar{a}_{r+1}$	$a_3 \dots a_r$
$\bar{a}_1 a_2 \bar{a}_{r+1}$	$a_3 \dots a_r$
$\bar{a}_1 \bar{a}_2 \bar{a}_{r+1}$	$a_3 \dots a_k,\ \bar{a}_3 a_4 \dots a_r,\ \bar{a}_3 \bar{a}_4 a_5 \dots a_r,\ \dots,\ \bar{a}_3 \bar{a}_4 \dots \bar{a}_r$
$\bar{a}_1 \bar{a}_2 a_{r+1}$	$\bar{a}_3 \bar{a}_4 \dots \bar{a}_r$

Let $P(x_1, x_2, x_3) = M(l_1, l_2, l_3)$, where l_i is one of $\{x_i, \bar{x}_i\}$. We can now use P and 2-clauses to express every other relation in $\mathcal{P}$. Given $M(l_1, l_2, l_3)$, every relation in $\mathcal{P}$ can be obtained by negating some subset of the variables. Hence it suffices to show that we can express structurally $M(\bar{l}_1, l_2, l_3)$ and $M(l_1, \bar{l}_2, l_3)$ (M is symmetric in x_1 and x_3). In the following let λ denote one of the literals $\{y, \bar{y}\}$, such that it is $\bar{y}$ if and only if l_1 is $\bar{x}_1$.

$$\begin{aligned} M(\bar{l}_1, l_2, l_3) &= (\bar{l}_1 \vee l_2 \vee l_3) \wedge (l_1 \vee \bar{l}_3) \\ &= \exists y \; (\bar{l}_1 \vee \bar{\lambda}) \wedge (\lambda \vee l_2 \vee l_3) \wedge (l_1 \vee \bar{l}_3) \\ &= \exists y \; (\bar{l}_1 \vee \bar{\lambda}) \wedge (\lambda \vee l_2 \vee l_3) \wedge (l_1 \vee \bar{l}_3) \wedge (\bar{\lambda} \vee \bar{l}_3) \\ &= \exists y \; (\bar{l}_1 \vee \bar{\lambda}) \wedge (l_1 \vee \bar{l}_3) \wedge M(\lambda, l_2, l_3) \\ &= \exists y \; (\bar{l}_1 \vee \bar{\lambda}) \wedge (l_1 \vee \bar{l}_3) \wedge P(y, x_2, x_3). \end{aligned}$$

In the second step the clause $(\bar{\lambda} \vee \bar{l}_3)$ is implied by the resolution of the clauses $(\bar{l}_1 \vee \bar{\lambda}) \wedge (l_1 \vee \bar{l}_3)$.

For the next expression let λ denote one of the literals $\{y, \bar{y}\}$, such that it is negated if and only if l_2 is $\bar{x}_2$.

$$\begin{aligned} M(l_1, \bar{l}_2, l_3) &= (l_1 \vee \bar{l}_2 \vee l_3) \wedge (\bar{l}_1 \vee \bar{l}_3) \\ &= \exists y\ (l_1 \vee l_3 \vee \lambda) \wedge (\bar{\lambda} \vee \bar{l}_2) \wedge (\bar{l}_1 \vee \bar{l}_3) \\ &= \exists y\ (\bar{\lambda} \vee \bar{l}_2) \wedge M(l_1, \lambda, l_3) \\ &= \exists y\ (\bar{\lambda} \vee \bar{l}_2) \wedge P(x_1, y, x_3). \end{aligned}$$

The above expressions are both based on resolution, and it is easy to check that they satisfy the properties of structural expressibility.

Step 4. *Structurally expressing* $\mathcal{S}_3$. We structurally express $(x_1 \vee x_2 \vee x_3)$ from M using a formula derived from a gadget in [17]. This gadget expresses $(x_1 \vee x_2 \vee x_3)$ in terms of "Protected OR," which corresponds to our relation M.

$$\begin{aligned} (x_1 \vee x_2 \vee x_3) = \exists y_1 \dots y_5\ &(x_1 \vee \bar{y}_1) \wedge (x_2 \vee \bar{y}_2) \wedge (x_3 \vee \bar{y}_3) \wedge (x_3 \vee \bar{y}_4) \\ &\wedge M(y_1, y_5, y_3) \wedge M(y_2, \bar{y}_5, y_4). \end{aligned} \tag{3.1}$$

The table below, listing the witnesses of each assignment for (x_1, x_2, x_3), shows that the conditions for structural expressibility are satisfied.

x_1, x_2, x_3	$y_1 \dots y_5$											
111	00011	00111	00110	00100	01100	01101	01001	11001	11000	10000	10010	10011
110							01001	11001	11000	10000		
100										10000		
101	00011	00111	00110	00100						10000	10010	10011
001	00011	00111	00110	00100								
011	00011	00111	00110	00100	01100	01101	01001					
010							01001					

From the relation $(x_1 \vee x_2 \vee x_3)$ we derive the other 3-clauses by reverse resolution, for instance,

$$(\bar{x}_1 \vee x_2 \vee x_3) = \exists y\ (\bar{x}_1 \vee \bar{y}) \wedge (y \vee x_2 \vee x_3). \qquad \square$$

To complete the proof of the structural expressibility theorem, we show that an arbitrary relation can be expressed structurally from $\mathcal{S}_3$.

LEMMA 3.5. *Let* $R \subseteq \{0,1\}^k$ *be any relation of arity* $k \geq 1$. R *is structurally expressible from* $\mathcal{S}_3$.

Proof. If $k \leq 3$, then R can be expressed as a formula in $\mathrm{CNF}(\mathcal{S}_3)$ with constants, without introducing witness variables. This kind of expression is always structural.

If $k \geq 4$, then R can be expressed as a formula in $\mathrm{CNF}(\mathcal{S}_k)$, without witnesses (i.e., structurally). We will show that every k-clause can be expressed structurally from $\mathcal{S}_{k-1}$. Then, by induction, it can be expressed structurally from $\mathcal{S}_3$. For simplicity we express a k-clause corresponding to the relation D_0. The remaining relations are expressed equivalently. We express D_0 in a way that is standard in other complexity reductions and turns out to be structural:

$$(x_1 \vee x_2 \vee \dots \vee x_k) = \exists y\ (x_1 \vee x_2 \vee y) \wedge (\bar{y} \vee x_3 \vee \dots \vee x_k).$$

This is the reverse operation of resolution. For any satisfying assignment for $\mathbf{x}$, its witness space is either $\{0\}$, $\{1\}$, or $\{0,1\}$, so in all cases it is connected. Furthermore, the only way two neighboring satisfying assignments for x can have no common witness

is if one of them has witness set $\{0\}$ and the other one has witness set $\{1\}$. This implies that the first one has $(x_3, \ldots, x_k) = (0, \ldots, 0)$, and the other one has $(x_1, x_2) = (0, 0)$, and thus they differ in the assignments of at least two variables: one from $\{x_1, x_2\}$ and one from $\{x_3, \ldots, x_k\}$. In that case they cannot be neighboring assignments. Therefore all requirements of structural expressibility are satisfied. □

3.3. Hardness results for 3-CNF-formulas. From Lemma 3.4 and Corollary 3.3, it follows that, to prove the hard side of our dichotomy theorems, it suffices to focus on 3-CNF-formulas.

The proof that CONN($\mathcal{S}_3$) and ST-CONN($\mathcal{S}_3$) are PSPACE-complete is fairly intricate; it entails a direct reduction from the computation of a space-bounded Turing machine. The result for ST-CONN can also be proved easily using results of Hearne and Demaine on nondeterministic constraint logic [17]. However, it does not appear that completeness for CONN follows from their results.

LEMMA 3.6. ST-CONN($\mathcal{S}_3$) *and* CONN($\mathcal{S}_3$) *are* PSPACE*-complete.*

Proof. Given a CNF($\mathcal{S}_3$)-formula φ and satisfying assignments $\mathbf{s}, \mathbf{t}$, we can guess a path of length at most 2^n between them and verify that each vertex along the path is indeed a solution to φ. Hence ST-CONN($\mathcal{S}_3$) is in NPSPACE, which equals PSPACE by Savitch's theorem. Similarly for CONN($\mathcal{S}_3$), by reusing space we can check for all pairs of assignments whether they are satisfying and, if they both are, whether they are connected in $G(\varphi)$. It follows that both problems are in PSPACE.

Next we show that CONN($\mathcal{S}_3$) and ST-CONN($\mathcal{S}_3$) are PSPACE-hard. Let A be a language decided by a deterministic Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ in space n^k for some constant $k \geq 1$, where n is the length of the input. We give a polynomial-time reduction from A to ST-CONN($\mathcal{S}_3$) and CONN($\mathcal{S}_3$).

The reduction maps a string w (with $|w| = n$) to a 3-CNF-formula φ and two satisfying assignments for the formula, which are connected in $G(\varphi)$ if and only if M accepts w. Furthermore, all satisfying assignments of φ are connected to one of these two assignments, so that $G(\varphi)$ is connected if and only if M accepts w.

Before we show how to construct φ, we modify M in several ways to obtain a new machine M' which depends on w:

1. We define the tape of M' to be cyclical of length $n^k + 1$ with a special symbol # written in cell 0 which cannot be overwritten. The input w is placed after this symbol. Notice that the machine M accepts or rejects w within n^k space, and therefore the # symbol is never read when the machine is initialized in a legal way (at the state q_0, input from Σ^*, and the head at the initial position). The # symbol may be read if the machine is initialized at a different configuration. We modify the transitions of M so that if the # symbol is read, the machine M' goes into the state q_{reject}.
2. We add to M' a clock that counts from 0 to $n^k \times |Q| \times |\Gamma|^{n^k} = 2^{O(n^{k+1})}$, which is the total number of possible distinct configurations of M when it uses only n^k space. For this, M' uses a separate tape of length $O(n^{k+1})$ with the alphabet $\{0, 1\}$. Before a transition happens, control is passed on to the clock, its counter is incremented, and finally the transition is completed.
3. We define a standard accepting configuration. Whenever q_{accept} is reached, the clock is stopped and set to zero, the original tape is erased (except for #) and the head is placed in the initial position, always in state q_{accept}.
4. Whenever q_{reject} is reached the machine goes into its initial configuration. First w is written back on the input tape after the # symbol. This step requires adding n states to the machine in order to write the n letters of w.

This increases the number of states of M' to $O(n)$. Next, the rest of the tape is erased, the clock is set to zero, the head is placed in the initial position, and the state is set to q_0 (and thus the computation resumes).

5. Whenever the clock overflows, the machine goes into q_{reject}.

The new machine M' runs forever if w is not in A, and it accepts if w is in A. It also has the property that every configuration having # in position 0 leads either to the accepting configuration or to the initial configuration with input w. Therefore the space of such configurations is connected if and only if $w \in A$. Let's denote by Q' the states of M' and by δ' its transitions. As mentioned earlier, $|Q'| = O(n)$, and M' runs on two tapes, one of size $N = n^k$ and the other (for the clock) of size $N_c = O(n^{k+1})$. The alphabet of M' on one tape is Γ, and on the other $\{0,1\}$. For simplicity we can also assume that at each transition the machine uses only one of the two tapes and moves its head either left or right.

Next, we construct an intermediate CNF-formula ψ whose solutions are the configurations of M'. However, the space of solutions of ψ is disconnected.

For each $i \in [N]$ and $a \in \Gamma$, we have a variable $x(i,a)$. If $x(i,a) = 1$, this means that the ith tape cell contains symbol a. For every $i \in [N]$ there is a variable $y(i)$ which is 1 if the head is at position i. For every $q \in Q'$, there is a variable $z(q)$ which is 1 if the current state is q. Similarly for every $j \in [N_c]$ and $a \in \{0,1\}$ we have variables $x_c(j,a)$ and a variable $y_c(j)$ which is 1 if the head of the clock tape is at position j.

We enforce the following conditions:

1. Every cell contains some symbol and cell 0 contains #:

$$\psi_1 = \bigwedge_{i\in[N]} \left(\vee_{a\in\Gamma} \;\; x(i,a)\right) \bigwedge_{j\in[N_c]} \left(\vee_{a\in\{0,1\}} \;\; x_c(j,a)\right) \wedge x(0,\#).$$

2. No cell contains two symbols:

$$\psi_2 = \bigwedge_{i\in[N]} \bigwedge_{a\neq a'\in\Gamma} \left(\overline{x(i,a)} \vee \overline{x(i,a')}\right) \bigwedge_{j\in[N_c]} \left(\overline{x_c(j,0)} \vee \overline{x_c(j,1)}\right).$$

3. The head is in some position, the clock head is in some position, and the machine is in some state:

$$\psi_3 = \left(\vee_{i\in[N]} \;\; y(i)\right) \bigwedge \left(\vee_{j\in[N_c]} \;\; y_c(j)\right) \bigwedge \left(\vee_{q\in Q'} \;\; z(q)\right).$$

4. The main tape head is in a unique position, the clock head is in a unique position, and the machine is in a unique state:

$$\psi_4 = \bigwedge_{i\neq i'\in[N]} \left(\overline{y(i)} \vee \overline{y(i')}\right) \bigwedge_{j\neq j'\in[N_c]} \left(\overline{y_c(j)} \vee \overline{y_c(j')}\right) \bigwedge_{q\neq q'\in Q'} \left(\overline{z(q)} \vee \overline{z(q')}\right).$$

Solutions of $\psi = \psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \psi_4$ are in 1-1 correspondence with configurations of M'. Furthermore, the assignments corresponding to any two distinct configurations differ in at least two variables (hence the space of solutions is totally disconnected).

Next, to connect the solution space along valid transitions of M', we relax conditions 2 and 4 by introducing new transition variables, which allow the head to have two states or a cell to have two symbols at the same time. This allows us to go from one configuration to the next.

Consider a transition $\delta'(q,a) = (q', b, R)$, which operates on the first tape, for example. Fix the position of the head of the first tape to be i. The variables that are

changed by the transition are $x(i,a)$, $y(i)$, $z(q)$, $x(i,b)$, $y(i+1)$, $z(q')$ (some of these may be the same variable, such as if $a = b$, but at least two are guaranteed to be different since the head has to move left or right). Before the transition the first three are set to 1, the second three are set to 0, and after the transition they are all flipped. Corresponding to this transition (which is specified by i, q, and a), we introduce a transition variable $t(i,q,a)$. We now relax conditions 2 and 4 as follows:

- Replace $\big(\overline{x(i,a)} \vee \overline{x(i,b)}\big)$ by $\big(\overline{x(i,a)} \vee \overline{x(i,b)} \vee t(i,q,a)\big)$.
- Replace $\big(\overline{y(i)} \vee \overline{y(i+1)}\big)$ by $\big(\overline{y(i)} \vee \overline{y(i+1)} \vee t(i,q,a)\big)$.
- Replace $\big(\overline{z(q)} \vee \overline{z(q')}\big)$ by $\big(\overline{z(q)} \vee \overline{z(q')} \vee t(i,q,a)\big)$.

This is done for every value of q, a, and i (and also for transitions acting on the clock tape). We add the transition variables to the corresponding clauses so that, for example, the clause $\big(\overline{x(i,a)} \vee \overline{x(i,b)}\big)$ could potentially become very long, such as

$$\Big(\overline{x(i,a)} \vee \overline{x(i,b)} \vee t(i,q_1,a) \vee t(i,q_2,a) \vee \cdots\Big).$$

However, the total number of transition variables is only polynomial in n. We also add a constraint for every pair of transition variables $t(i,q,a)$, $t(i',q',a')$, saying they cannot be 1 simultaneously: $(\overline{t(i,q,a)} \vee \overline{t(i',q',a')})$. This ensures that only one transition can be happening at any time. The effect of adding the transition variables to the clauses of ψ_2 and ψ_4 is that by setting $t(i,q,a)$ to 1, we can simultaneously set $x(i,a)$ and $x(i,b)$ to 1, and so on. This gives a path from the initial configuration to the final configuration as follows: Set $t(i,q,a) = 1$, set $x(i,b) = 1$, $y(i+1) = 1$, $z(q') = 1$, $x(i,a) = 0$, $y(i) = 0$, $z(q) = 0$, then set $t(i,q,a) = 0$. Thus consecutive configurations are now connected. To avoid connecting to other configurations, we also add an expression to ensure that these are the only assignments the 6 variables can take when $t(i,q,a) = 1$:

$$\psi_{i,q,a} = \overline{t(i,q,a)} \vee ((x(i,a), y(i), z(q), x(i,b)), y(i+1), z(q')) \in \{111000, 111100, 111110, 111111, 011111, 001111, 000111\}.$$

This expression can of course be written in conjunctive normal form.

Call the resulting CNF-formula $\varphi(\mathbf{x}, \mathbf{x_c}, \mathbf{y}, \mathbf{y_c}, \mathbf{z}, \mathbf{t})$. Note that $\varphi(\mathbf{x}, \mathbf{x_c}, \mathbf{y}, \mathbf{y_c}, \mathbf{z}, \mathbf{0}) = \psi(\mathbf{x}, \mathbf{x_c}, \mathbf{y}, \mathbf{y_c}, \mathbf{z})$, so a solution where all transition variables are 0 corresponds to a configuration of M'. To see that we have not introduced any shortcut between configurations that are not valid machine transitions, notice that in any solution of φ, at most a single transition variable can be 1. Therefore none of the transitional solutions belonging to different transitions can be adjacent. Furthermore, out of the solutions that have a transition variable set to 1, only the first and the last correspond to a valid configuration. Therefore none of the intermediate solutions (between the starting and the ending configuration of a transition) can be adjacent to a solution with all transition variables set to 0.

Finally, we define the assignment $\mathbf{s}$ to be the one corresponding to the initial configuration of M' with w on the tape, and $\mathbf{t}$ to be the assignment corresponding to the accepting configuration—where the state is q_{accept}, and the tape has only the # symbol at position 0. This completes the reduction.

The formula φ is a CNF-formula where clause size is unbounded. We use the same algorithm for producing structural expressions for k-clauses as in the proof of Lemma 3.5 to get a 3-CNF-formula. By Lemma 3.2 and Corollary 3.3, ST-CONN and CONN for $\mathcal{S}_3$ are PSPACE-complete. ∎

By Lemma 3.4 and Corollary 3.3, this completes the proof of the hardness part of the dichotomies for CONN and ST-CONN (Theorems 2.8 and 2.9).

Finally, we show that 3-CNF-formulas can have exponential diameter by inductively constructing a path of length at least $2^{\frac{n}{2}}$ on n variables and then identifying it with the solution space of a 3-CNF-formula with $O(n^2)$ clauses. By Lemma 3.4 and Corollary 3.3, this implies the hardness part of the diameter dichotomy (Theorem 2.10).

LEMMA 3.7. *For n even, there is a* 3-CNF*-formula φ_n with n variables and $O(n^2)$ clauses, such that $G(\varphi_n)$ is a path of length greater than $2^{\frac{n}{2}}$.*

Proof. The construction is in two steps: We first exhibit an induced subgraph G_n of the n-dimensional hypercube with large diameter. We then construct a 3-CNF-formula φ_n so that $G_n = G(\varphi_n)$.

The graph G_n is a path of length $2^{\frac{n}{2}}$. We construct it using induction. For $n = 2$, we take $V(G_2) = \{(0,0),(0,1),(1,1)\}$ which has diameter 2. Assume that we have constructed G_{n-2} with $2^{\frac{n-2}{2}}$ vertices, and with distinguished vertices $\mathbf{s_{n-2}}, \mathbf{t_{n-2}}$ such that the shortest path from $\mathbf{s_{n-2}}$ to $\mathbf{t_{n-2}}$ in G_{n-2} has length $2^{\frac{n-2}{2}}$. We now describe the set $V(G_n)$. For each vertex $\mathbf{v} \in V(G_{n-2})$, $V(G_n)$ contains two vertices $(\mathbf{v},0,0)$ and $(\mathbf{v},1,1)$. Note that the subgraph induced by these vertices alone consists of two disconnected copies of G_{n-2}. To connect these two components, we add the vertex $\mathbf{m} = (\mathbf{t},0,1)$ (which is connected to $(\mathbf{t},0,0)$ and $(\mathbf{t},1,1)$ in the induced subgraph). Note that the resulting graph G_n is connected, but any path from $(\mathbf{u},0,0)$ to $(\mathbf{v},1,1)$ must pass through $\mathbf{m}$. Further note that by induction the graph G_n is also a path. The vertices $\mathbf{s_n} = (\mathbf{s_{n-2}},0,0)$ and $\mathbf{t_n} = (\mathbf{s_{n-2}},1,1)$ are diametrically opposite ends of this path. The path length is at least $2 \cdot 2^{\frac{n-2}{2}} + 2 > 2^{\frac{n}{2}}$. Also $\mathbf{s_2} = (0,0)$, $\mathbf{s_n} = (\mathbf{s_{n-2}},0,0)$, $\mathbf{t_n} = (\mathbf{s_{n-2}},1,1)$ and hence $\mathbf{s_n} = (0,\ldots,0), \mathbf{t_n} = (0,\ldots,0,1,1)$.

We construct a sequence of 3-CNF-formulas $\varphi_n(x_1,\ldots,x_n)$ so that $G_n = G(\varphi_n)$. Let $\varphi_2(x_1,x_2) = \bar{x}_1 \vee x_2$. Assume we have $\varphi_{n-2}(x_1,\ldots,x_{n-2})$. We add two variables x_{n-1} and x_n and the clauses

$$\varphi_{n-2}(x_1,\ldots,x_{n-2}), \quad \bar{x}_{n-1} \wedge x_n,$$

$$x_{n-1} \vee \bar{x}_n \vee \bar{x}_i \qquad \text{for } \ i \le n-4, \tag{3.2}$$

$$x_{n-1} \vee \bar{x}_n \vee x_i \qquad \text{for } \ i = n-3, n-2. \tag{3.3}$$

Note that a clause in (3.2) is just the implication $(\bar{x}_{n-1} \wedge x_n) \to \bar{x}_i$. Thus clauses (3.2), (3.3) enforce the condition that $x_{n-1} = 0, x_n = 1$ implies that $(x_1,\ldots,x_{n-2}) = \mathbf{t_{n-2}} = (0,\ldots,0,1,1)$. ∎

4. The easy case of the dichotomy: Tight sets of relations.

4.1. Schaefer sets of relations. We begin by showing that all Schaefer sets of relations are tight. Schaefer relations are characterized by closure properties. We say that an r-ary relation R is closed under some k-ary operation $\alpha : \{0,1\}^k \to \{0,1\}$ if for every $\mathbf{a^1}, \mathbf{a^2}, \ldots, \mathbf{a^k} \in R$, the tuple $(\alpha(a_1^1, a_1^2, \ldots, a_1^k), \ldots, \alpha(a_r^1, \ldots, a_r^k))$ is in R. We denote this tuple by $\alpha(\mathbf{a^1}, \ldots, \mathbf{a^k})$.

We will use the following lemma about closure properties on several occasions.

LEMMA 4.1. *If a logical relation R is closed under an operation $\alpha : \{0,1\}^k \to \{0,1\}$ such that $\alpha(1,1,\ldots,1) = 1$ and $\alpha(0,0,\ldots,0) = 0$ (a.k.a. an idempotent operation), then every connected component of $G(R)$ is closed under α.*

Proof. Consider $\mathbf{a^1}, \ldots, \mathbf{a^k} \in R$, such that they all belong to the same connected component of $G(R)$. It suffices to prove that $\mathbf{a} = \alpha(\mathbf{a^1}, \ldots, \mathbf{a^k})$ is in the same connected component of $G(R)$. To that end we will first prove that for any $\mathbf{s}, \mathbf{t} \in R$ if there

is a path from $\mathbf{s}$ to $\mathbf{t}$ in $G(R)$, then there is a path from $\alpha(\mathbf{b^1},\dots,\mathbf{b^{i-1}},\mathbf{s},\mathbf{b^{i+1}},\dots,\mathbf{b^k})$ to $\alpha(\mathbf{b^1},\dots,\mathbf{b^{i-1}},\mathbf{t},\mathbf{b^{i+1}},\dots,\mathbf{b^k})$ for any $\mathbf{b^1},\dots,\mathbf{b^k} \in R$. This observation implies that there is a path from $\mathbf{a^1} = \alpha(\mathbf{a^1},\mathbf{a^1},\dots,\mathbf{a^1})$ to $\alpha(\mathbf{a^1},\mathbf{a^2},\mathbf{a^1},\dots,\mathbf{a^1})$, from there to $\alpha(\mathbf{a^1},\mathbf{a^2},\mathbf{a^3},\mathbf{a^1},\dots,\mathbf{a^1})$ and so on, to $\alpha(\mathbf{a^1},\mathbf{a^2},\dots,\mathbf{a^k}) = \mathbf{a}$. Thus $\mathbf{a}$ is in the same connected component of $G(R)$ as $\mathbf{a^1}$.

Let the path from $\mathbf{s}$ to $\mathbf{t}$ be $\mathbf{s} = \mathbf{s^1} \to \mathbf{s^2} \to \dots \mathbf{s^m} = \mathbf{t}$. For every $j \in \{1,2,\dots,m-1\}$, the tuples $\alpha(\mathbf{b^1},\dots,\mathbf{b^{i-1}},\mathbf{s^j},\mathbf{b^{i+1}},\dots,\mathbf{b^m})$ and $\alpha(\mathbf{b^1},\dots,\mathbf{b^{i-1}},\mathbf{s^{j+1}},\mathbf{b^{i+1}},\dots,\mathbf{b^m})$ differ in at most one position (the position in which $\mathbf{s^j}$ and $\mathbf{s^{j+1}}$ are different); therefore they belong to the same component of $G(R)$. Thus $\alpha(\mathbf{b^1},\dots,\mathbf{b^{i-1}},\mathbf{s^1},\mathbf{b^{i+1}},\dots,\mathbf{b^m})$ and $\alpha(\mathbf{b^1},\dots,\mathbf{b^{i-1}},\mathbf{s^m},\mathbf{b^{i+1}},\dots,\mathbf{b^m})$ belong to the same component. □

We are ready to prove that all Schaefer relations are tight.

LEMMA 4.2. *Let R be a logical relation.*

1. *If R is bijunctive, then R is componentwise bijunctive.*
2. *If R is Horn, then R is* OR*-free.*
3. *If R is dual Horn, then R is* NAND*-free.*
4. *If R is affine, then R is componentwise bijunctive,* OR*-free, and* NAND*-free.*

Proof. The case of bijunctive relations follows immediately from Lemma 4.1 and the fact that a relation is bijunctive if and only if it is closed under the ternary majority operation maj, which is idempotent.

The cases of Horn and dual Horn are symmetric. Suppose an r-ary Horn relation R is not OR-free. Then there exist $i,j \in \{1,\dots,r\}$ and constants $t_1,\dots,t_r \in \{0,1\}$ such that the relation $R(t_1,\dots,t_{i-1},x,t_{i+1},\dots,t_{j-1},y,t_{j+1},\dots,t_r)$ on variables x and y is equivalent to $x \vee y$, i.e.,

$$R(t_1,\dots,t_{i-1},x,t_{i+1},\dots,t_{j-1},y,t_{j+1},\dots,t_r) = \{01,11,10\}.$$

Thus the tuples $\mathbf{t^{00}},\mathbf{t^{01}}\mathbf{t^{10}},\mathbf{t^{11}}$ defined by $(t_i^{ab},t_j^{ab}) = (a,b)$ and $t_k^{ab} = t_k$ for every $k \notin \{i,j\}$, where $a,b, \in \{0,1\}$ satisfy $\mathbf{t^{10}},\mathbf{t^{11}},\mathbf{t^{01}} \in R$ and $\mathbf{t^{00}} \notin R$. However, since every Horn relation is closed under $\wedge$, it follows that $\mathbf{t^{01}} \wedge \mathbf{t^{10}} = \mathbf{t^{00}}$ must be in R, which is a contradiction.

For the affine case, a small modification of the last step of the above argument shows that an affine relation also is OR-free; therefore, dually, it is also NAND-free. Namely, since a relation R is affine if and only if it is closed under ternary $\oplus$, it follows that $\mathbf{t^{01}} \oplus \mathbf{t^{11}} \oplus \mathbf{t^{10}} = \mathbf{t^{00}}$ must be in R.

Since the connected components of an affine relation are both OR-free and NAND-free, the subgraphs that they induce are hypercubes, which are also bijunctive relations. Therefore an affine relation is also componentwise bijunctive. □

These containments are proper. For instance, $R_{1/3} = \{100,010,001\}$ is componentwise bijunctive, but not bijunctive as $\mathrm{maj}(100,010,001) = 000 \notin R_{1/3}$.

4.2. Structural properties of tight sets of relations. In this section, we explore some structural properties of the solution graphs of tight sets of relations. These properties provide simple algorithms for CONN($\mathcal{S}$) and ST-CONN($\mathcal{S}$) for tight sets $\mathcal{S}$, and also guarantee that for such sets, the diameter of $G(\varphi)$ of CNF($\mathcal{S}$)-formula φ is linear.

LEMMA 4.3. *Let $\mathcal{S}$ be a set of componentwise bijunctive relations and φ a* CNF($\mathcal{S}$)*-formula. If $\mathbf{a}$ and $\mathbf{b}$ are two solutions of φ that lie in the same component of $G(\varphi)$, then $d_\varphi(\mathbf{a},\mathbf{b}) = |\mathbf{a}-\mathbf{b}|$, i.e., no distance expands.*

Proof. Consider first the special case in which every relation in $\mathcal{S}$ is bijunctive. In this case, φ is equivalent to a 2-CNF-formula and so the space of solutions of φ is closed under majority. We show that there is a path in $G(\varphi)$ from $\mathbf{a}$ to $\mathbf{b}$

such that along the path only the assignments on variables with indices from the set $D = \{i | a_i \neq b_i\}$ change. This implies that the shortest path is of length $|D|$ by induction on $|D|$. Consider any path $\mathbf{a} \to \mathbf{u^1} \to \cdots \to \mathbf{u^r} \to \mathbf{b}$ in $G(\varphi)$. We construct another path by replacing $\mathbf{u^i}$ by $\mathbf{v^i} = \text{maj}(\mathbf{a}, \mathbf{u^i}, \mathbf{b})$ for $i = 1, \ldots, r$ and removing repetitions. This is a path because for any i $\mathbf{v^i}$ and $\mathbf{v^{i+1}}$ differ in at most one variable. Furthermore, $\mathbf{v^i}$ agrees with $\mathbf{a}$ and $\mathbf{b}$ for every i for which $a_i = b_i$. Therefore, along this path only variables in D are flipped.

For the general case, we show that every component F of $G(\varphi)$ is the solution space of a 2-CNF-formula φ'. Let $R \in \mathcal{S}$ be a relation with two components, R_1, R_2, each of which are bijunctive. Consider a clause in φ of the form $R(x_1, \ldots, x_k)$. The projection of F onto $x_1, \ldots, x_k$ is itself connected and must satisfy R. Hence it lies within one of the two components R_1, R_2; assume it is R_1. We replace $R(x_1, \ldots, x_k)$ by $R_1(x_1, \ldots, x_k)$. Call this new formula φ_1. $G(\varphi_1)$ consists of all components of $G(\varphi)$ whose projection on $x_1, \ldots, x_k$ lies in R_1. We repeat this for every clause. Finally we are left with a formula φ' over a set of bijunctive relations. Hence φ' is bijunctive and $G(\varphi')$ is a component of $G(\varphi)$. So the claim follows from the bijunctive case. □

COROLLARY 4.4. *Let $\mathcal{S}$ be a set of componentwise bijunctive relations. Then*

1. *for every $\varphi \in \text{CNF}(\mathcal{S})$ with n variables, the diameter of each component of $G(\varphi)$ is bounded by n;*
2. *st-CONN($\mathcal{S}$) is in* P*;*
3. *CONN($\mathcal{S}$) is in* coNP*.*

Proof. The bound on diameter is an immediate consequence of Lemma 4.3.

The following algorithm solves st-CONN($\mathcal{S}$) given vertices $\mathbf{s}, \mathbf{t} \in G(\varphi)$. Start with $\mathbf{u} = \mathbf{s}$. At each step, find a variable x_i so that $u_i \neq t_i$ and such that if we flip x_i, the assignment would still be satisfying. Repeat until $\mathbf{t}$ is reached. If at any stage no such variable exists, then declare that $\mathbf{s}$ and $\mathbf{t}$ are not connected. If the $\mathbf{s}$ and $\mathbf{t}$ are disconnected, the algorithm is bound to fail. So assume that they are connected. Correctness is proved by induction on $d = |\mathbf{s} - \mathbf{t}|$. It is clear that the algorithm works when $d = 1$. Assume that the algorithm works for $d - 1$. If s and t are connected and are distance d apart, Lemma 4.3 implies there is a path of length d between them in $G(\varphi)$. In particular, the algorithm will find a variable x_i to flip. The resulting assignment is at distance $d - 1$ from $\mathbf{t}$, so now we proceed by induction.

Next we prove that CONN($\mathcal{S}$) $\in$ coNP. A short certificate that the graph is not connected is a pair of assignments $\mathbf{s}$ and $\mathbf{t}$ which are solutions from different components. To verify that they are disconnected it suffices to run the algorithm for st-CONN. □

We consider sets of OR-free relations. Define the *coordinatewise partial order* $\leq$ on Boolean vectors as follows: $\mathbf{a} \leq \mathbf{b}$ if $a_i \leq b_i$ for each i. A monotone path between $\mathbf{a}$ and $\mathbf{b}$ is a path in $G(\varphi)$, $\mathbf{a} \to \mathbf{u^1} \to \cdots \to \mathbf{u^r} \to \mathbf{b}$ such that $\mathbf{a} \leq \mathbf{u^1} \leq \cdots \leq \mathbf{u^r} \leq \mathbf{b}$.

LEMMA 4.5. *Let $\mathcal{S}$ be a set of* OR*-free relations and φ a* CNF($\mathcal{S}$)*-formula. Every component of $G(\varphi)$ contains a minimum solution with respect to the coordinatewise order; moreover, every solution is connected to the minimum solution in the same component via a monotone path.*

Proof. We call a satisfying assignment locally minimal if it has no neighboring satisfying assignments that are smaller than it. We will show that there is exactly one such assignment in each component of $G(\varphi)$.

Suppose there are two distinct locally minimal assignments $\mathbf{u}$ and $\mathbf{u'}$ in some component of $G(\varphi)$. Consider the path between them where the maximum Hamming weight of assignments on the path is minimized. If there are many such paths, pick

one where the smallest number of assignments have the maximum Hamming weight. Denote this path by $\mathbf{u} = \mathbf{u^1} \to \mathbf{u^2} \to \cdots \to \mathbf{u^r} = \mathbf{u'}$. Let $\mathbf{u^i}$ be an assignment of largest Hamming weight in the path. Then $\mathbf{u^i} \neq \mathbf{u}$ and $\mathbf{u^i} \neq \mathbf{u'}$, since $\mathbf{u}$ and $\mathbf{u'}$ are locally minimal. The assignments $\mathbf{u^{i-1}}$ and $\mathbf{u^{i+1}}$ differ in exactly 2 variables, say, in x_1 and x_2. So $\{u_1^{i-1}u_2^{i-1},\ u_1^i u_2^i,\ u_1^{i+1}u_2^{i+1}\} = \{01, 11, 10\}$. Let $\hat{\mathbf{u}}$ be such that $\hat{u}_1 = \hat{u}_2 = 0$, and $\hat{u}_i = u_i$ for $i > 2$. If $\hat{\mathbf{u}}$ is a solution, then the path $\mathbf{u^1} \to \mathbf{u^2} \to \cdots \to \mathbf{u^{i-1}} \to \hat{\mathbf{u}} \to \mathbf{u^{i+1}} \to \cdots \to \mathbf{u^r}$ contradicts the way we chose the original path. Therefore, $\hat{\mathbf{u}}$ is not a solution. This means that there is a clause that is violated by it but is satisfied by $\mathbf{u^{i-1}}$, $\mathbf{u^i}$, and $\mathbf{u^{i+1}}$. So the relation corresponding to that clause is not OR-free, which is a contradiction.

The unique locally minimal solution in a component is its minimum solution, because starting from any other assignment in the component, it is possible to keep moving to neighbors that are smaller, and the only time it becomes impossible to find such a neighbor is when the locally minimal solution is reached. Therefore, there is a monotone path from any satisfying assignment to the minimum in that component. □

COROLLARY 4.6. *Let $\mathcal{S}$ be a set of* OR*-free relations. Then*

1. *for every $\varphi \in \mathrm{CNF}(\mathcal{S})$ with n variables, the diameter of each component of $G(\varphi)$ is bounded by $2n$;*
2. ST-CONN($\mathcal{S}$) *is in* P*;*
3. CONN($\mathcal{S}$) *is in* coNP*.*

Proof. Given solutions $\mathbf{s}$ and $\mathbf{t}$ in the same component of $G(\varphi)$, there is a monotone path from each to the minimal solution $\mathbf{u}$ in the component. This gives a path from $\mathbf{s}$ to $\mathbf{t}$ of length at most $2n$. To check if $\mathbf{s}$ and $\mathbf{t}$ are connected, we just check that the minimal assignments reached from $\mathbf{s}$ and $\mathbf{t}$ are the same. □

Sets of NAND-free relations are handled dually to OR-free relations. In this case there is a maximum solution in every connected component of $G(\phi)$ and every solution is connected to it via a monotone path. Finally, putting everything together, we complete the proofs of all our dichotomy theorems.

COROLLARY 4.7. *Let $\mathcal{S}$ be a tight set of relations. Then*

1. *for every $\varphi \in \mathrm{CNF}(\mathcal{S})$ with n variables, the diameter of each component of $G(\varphi)$ is bounded by $2n$;*
2. ST-CONN($\mathcal{S}$) *is in* P*;*
3. CONN($\mathcal{S}$) *is in* coNP*.*

4.3. The complexity of CONN for tight sets of relations. We pinpoint the complexity of CONN($\mathcal{S}$) for the tight cases which are not Schaefer, using a result of Juban [19].

LEMMA 4.8. *For $\mathcal{S}$ tight, but not Schaefer,* CONN($\mathcal{S}$) *is* coNP*-complete.*

Proof. The problem ANOTHER-SAT($\mathcal{S}$) is as follows: Given a formula φ in CNF($\mathcal{S}$) and a solution $\mathbf{s}$, does there exist a solution $\mathbf{t} \neq \mathbf{s}$? Juban [19, Theorem 2] shows that if $\mathcal{S}$ is not Schaefer, then ANOTHER-SAT is NP-complete. He also shows [19, Corollary 1] that if $\mathcal{S}$ is not Schaefer, then the relation $x \neq y$ is expressible from $\mathcal{S}$ through substitutions.

Since $\mathcal{S}$ is not Schaefer, ANOTHER-SAT($\mathcal{S}$) is NP-complete. Let $\varphi, \mathbf{s}$ be an instance of ANOTHER-SAT on variables $x_1, \ldots, x_n$. We define a CNF($\mathcal{S}$)-formula ψ on the variables $x_1, \ldots, x_n, y_1, \ldots, y_n$ as

$$\psi(x_1, \ldots, x_n, y_1, \ldots, y_n) = \varphi(x_1, \ldots, x_n) \wedge_i (x_i \neq y_i).$$

It is easy to see that $G(\psi)$ is connected if and only if $\mathbf{s}$ is the unique solution to φ. □

We are left with the task of determining the complexity of CONN($\mathcal{S}$) for the case when $\mathcal{S}$ is a Schaefer set of relations. In Lemmas 4.9 and 4.10 we show that CONN($\mathcal{S}$) is in P if $\mathcal{S}$ is affine or bijunctive. This leaves the case of Horn and dual Horn, which we discuss at the end of this section.

LEMMA 4.9. *If $\mathcal{S}$ is a bijunctive set of relations, then there is a polynomial-time algorithm for* CONN($\mathcal{S}$).

Proof. Consider a formula $\phi(x_1, \ldots, x_n)$ in CNF($\mathcal{S}$). Since $\mathcal{S}$ is a bijunctive set of relations ϕ can be written as a 2-CNF-formula. Since satisfiability of 2-CNF-formulas is decidable in polynomial time, it is easy to decide for a given variable x_i whether there exist solutions in which it takes a particular value in $\{0, 1\}$. The variables which can take only one value are assigned that value. Without loss of generality we can assume that the resulting 2-CNF formula is $\psi(x_1, \ldots, x_m)$.

Consider the graph of implications of ψ defined in the following way: The vertices are the literals $x_1, \ldots, x_m, \bar{x}_1, \ldots, \bar{x}_m$. There is a directed edge from literal l_1 to literal l_2 if and only if ψ contains a clause containing l_2 and the negation of l_1, which we denote by $\bar{l}_1$ (if l_1 is a negated variable $\bar{x}$, then $\bar{l}_1$ denotes x). The directed edge represents the fact that in a satisfying assignment if the literal l_1 is assigned true, then the literal l_2 is also assigned true. We will show that $G(\psi)$ is disconnected if and only if the graph of implications contains a directed cycle. This property can be checked in polynomial time.

Suppose the graph of implications contains a directed cycle of literals $l_1 \to l_2 \to l_3 \to \cdots \to l_k \to l_1$. By the construction, the graph also contains a directed cycle on the negations of these literals, but in the opposite direction: $\bar{l}_k \to \bar{l}_{k-1} \to \cdots \to \bar{l}_2 \to \bar{l}_1 \to \bar{l}_k$. There is a satisfying assignment $\mathbf{s}$ in which l_1 is assigned 1, and also a satisfying assignment $\mathbf{t}$ in which $\bar{l}_1$ is assigned 1. By the implications, in $\mathbf{s}$ the literals $l_1, l_2, \ldots, l_k$ are assigned 1, and in $\mathbf{t}$ $\bar{l}_1, \bar{l}_2, \ldots, \bar{l}_k$ are assigned 1. Suppose there is a path from $\mathbf{s}$ to $\mathbf{t}$. Then let l_i be the first literal in the cycle whose value changes along the path from $\mathbf{s}$ to $\mathbf{t}$. Then there is a satisfying assignment in which l_i is assigned 0, whereas all other literals on the cycle are assigned 1. On the other hand, this cannot be a satisfying assignment because the edge (l_{i-1}, l_i) implies that there is a clause containing only l_i and the negation of l_{i-1}, and this clause is violated by the assignment. This is a contradiction, and therefore there can be no path from $\mathbf{s}$ to $\mathbf{t}$.

Next, suppose the graph of implications contains no directed cycle, and $G(\psi)$ is disconnected. Let $\mathbf{s}$ and $\mathbf{t}$ be satisfying assignments from different connected components of $G(\psi)$ that are at minimum Hamming distance. Let U be the set of variables on which $\mathbf{s}$ and $\mathbf{t}$ differ. There are two literals corresponding to each variable, and let $U^{\mathbf{s}}$ and $U^{\mathbf{t}}$ denote, respectively, the literals that are true in $\mathbf{s}$ and in $\mathbf{t}$. Since the directed graph induced by $U^{\mathbf{s}}$ in the implications graph contains no directed cycle, there exists a literal $l \in U^{\mathbf{s}}$ without an incoming edge from another literal in $U^{\mathbf{s}}$. Otherwise, by following the incoming edges we would find a cycle in the graph induced by $U^{\mathbf{s}}$. In addition, for every literal $l' \notin U^{\mathbf{s}}$ that is assigned true by $\mathbf{s}$, there is no edge from l' to l because that would contradict the fact that $\mathbf{t}$ is also a satisfying assignment (by the definition of U, l' is assigned true by $\mathbf{t}$). Therefore, with respect to $\mathbf{s}$ the literal l does not appear in any clause in which it is implied, i.e., in which it is the only satisfying literal. Thus, the value of the corresponding variable can be flipped and the resulting assignment is still satisfying. This assignment is in the same component as $\mathbf{s}$ but it is closer to $\mathbf{t}$, which contradicts our choice of $\mathbf{s}$ and $\mathbf{t}$. □

LEMMA 4.10. *If $\mathcal{S}$ is an affine set of relations, then there is a polynomial-time algorithm for* CONN($\mathcal{S}$).

Proof. An affine formula can be described as the set of solutions of a linear system of equations. For any solution, if only a variable that appears in at least one of the equations is flipped, the resulting assignment is not a solution. Therefore it suffices to check whether the system has more than one solution (after variables that don't appear in any equation are removed), which is easily done by checking the rank of the matrix obtained from the Gaussian elimination algorithm. □

We are left with characterizing the complexity of CONN for sets of Horn relations and for sets of dual Horn relations. In the conference version [16] of the present paper, we had conjectured that if $\mathcal{S}$ is Horn or dual Horn, then CONN($\mathcal{S}$) is in P, but this was disproved by Makino, Tamaki, and Yamamoto [24]. They showed that CONN($\{R_2\}$) is coNP-complete, where $R_2 = \{0,1\}^3 \backslash \{110\}$; hence there exist Horn (and by symmetry also dual Horn) sets of relations for which CONN is coNP-complete. Their proof is via a reduction from POSITIVE NOT-ALL-EQUAL 3-SAT, which as seen earlier is SAT($\{R_{\text{NAE}}\}$), where $R_{\text{NAE}} = \{0,1\}^3 \setminus \{000, 111\}$. This problem is also known as 3-hypergraph 2-colorability,

The relation R_2 is a 3-clause with one positive literal. We will describe a natural set of Horn relations first introduced in [14], which cannot be used to express R_2. We show that for this set there is a polynomial-time algorithm for CONN.

DEFINITION 4.11. *A logical relation R is* implicative hitting set-bounded− *(*IHSB−*) if it is the set of solutions of a Horn formula in which all clauses of size greater than* 2 *have only negative literals. Similarly, R is* implicative hitting set-bounded+ *(*IHSB+*) if it is the set of solutions of a dual Horn formula in which all clauses of size greater than* 2 *have only positive literals.*

These types of logical relations can be characterized by closure properties. A relation R is IHSB− if and only if it is closed under $\mathbf{a} \wedge (\mathbf{b} \vee \mathbf{c})$; in other words if $\mathbf{a}, \mathbf{b}, \mathbf{c} \in R$, where R is of arity r, then $\mathbf{a} \wedge (\mathbf{b} \vee \mathbf{c}) = (a_1 \wedge (b_1 \vee c_1), a_2 \wedge (b_2 \vee c_2), \ldots, a_r \wedge (b_r \vee c_r)) \in R$. A relation R is IHSB+ if and only if it is closed under $\mathbf{a} \vee (\mathbf{b} \wedge \mathbf{c})$. While the definition may at first look unnatural, it comes from Post's classification of Boolean functions (see [5]). One of the consequences of this classification is that IHSB− relations cannot express all Horn relations, and in particular R_2, even in the sense of Schaefer's expressibility. For the purposes of structural expressibility we can define an even larger class of relations which cannot structurally express R_2 (unless P = coNP).

DEFINITION 4.12. *A logical relation R is* componentwise IHSB− (IHSB+) *if every connected component of $G(R)$ is IHSB− (IHSB+).*

By Lemma 4.1, every relation that is IHSB− (IHSB+) is also componentwise IHSB− (IHSB+). Of course, the class of componentwise IHSB− relations is much broader and in fact includes relations that are not even Horn, such as $R_{1/3}$. However, in the following lemma we consider only componentwise IHSB− (IHSB+) relations which are Horn (dual Horn). We will say that a set of relations $\mathcal{S}$ is componentwise IHSB− (IHSB+) if every relation in $\mathcal{S}$ is componentwise IHSB− (IHSB+).

LEMMA 4.13. *If $\mathcal{S}$ is a set of relations that are Horn (dual Horn) and componentwise IHSB− (IHSB+), then there is a polynomial-time algorithm for* CONN($\mathcal{S}$).

Proof. First we consider the case in which every relation in $\mathcal{S}$ is IHSB−. The formula can be written as a conjunction of Horn clauses, such that clauses of length greater than 2 have only negative literals. Let all unit clauses be assigned and propagated—their variables take the same value in all satisfying assignments. The

resulting formula is also IHSB− and has two kinds of clauses: 2-clauses with one positive and one negative literal, and clauses of size 2 or more with only negative literals. The assignment of zero to all variables is satisfying. There is more than one connected component if and only if there is another assignment that is locally minimal by Lemma 4.5. A locally minimal satisfying assignment is such that if any of the variables assigned 1 is changed to 0 the resulting assignment is not satisfying. Thus all variables assigned 1 appear in at least one 2-clause with one positive and one negative literal for which both variables are assigned 1. We say that such an assignment certifies the disconnectivity.

To describe the algorithm, we first define the following implication graph G. The vertices are the set of variables. There is a directed edge (x_i, x_j) if and only if $(x_j \vee \bar{x}_i)$ is a clause in the IHSB− representation. Let $S_1, \ldots, S_m$ be the sets of variables in clauses with only negative literals. For every variable x_i let T_i denote the set of variables reachable from x_i in the directed graph. If $x_i \in T_i$, then x_i lies in a directed cycle. Note that if x_i is set to 1, then every variable in T_i must also be set to 1. The algorithm rejects if and only if there exists a variable x_i such that $x_i \in T_i$ and T_i does not contain S_j for any $j \in \{1, \ldots, m\}$. We show that this happens if and only if the solution graph is disconnected. Note that the algorithm runs in polynomial time.

Assume that the graph of solutions is disconnected and consider the satisfying assignment $\mathbf{s}$ that certifies disconnectivity. Let U be the set of variables x_i such that $s_i = 1$. Since every variable in U appears in at least one 2-clause for which both variables are from U, the directed graph induced by U is such that every vertex has an incoming edge. By starting at any vertex in U and following the incoming edge backwards until we repeat some vertex, we find a cycle in the subgraph induced by U. For any variable x_i in such a cycle it holds that $x_i \in T_i$. Further $T_i \subseteq U$, since setting x_i to 1 forces all variables in T_i to be 1. Also T_i cannot contain S_j for any j, else the corresponding clause would not be satisfied by $\mathbf{s}$. Thus the algorithm rejects whenever the solution graph is disconnected.

Conversely, if the algorithm rejects, there exists a variable x_i such that $x_i \in T_i$ and T_i does not contain S_j for any $j \in \{1, \ldots, m\}$. Consider the assignment in which all variables from T_i are assigned 1, and the rest are assigned 0. We will show that this assignment is satisfying and it is a certificate for disconnectivity. Clauses which contain only negated variables are satisfied since $S_j \not\subset T_i$ for all j. Now consider a clause of the form $(x_j \vee \bar{x}_k)$ and note that there is a directed edge (x_k, x_j). If $x_k = 0$, this is satisfied. If $x_k = 1$, then $x_k \in T_i$, and hence $x_j \in T_i$ because of the edge (x_k, x_j). But then x_j is set to 1, so the clause is satisfied. To show that this solution is minimal, consider trying to set $x_k \in T_i$ to 0. There is an incoming edge (x_j, x_k) for some $x_j \in T_i$, and hence a clause $(x_k \vee \bar{x}_j)$, which will become unsatisfied if we set $x_k = 0$. Thus we have a certificate for the space being disconnected.

Next, consider a formula $\phi(x_1, \ldots, x_n)$ in CNF($\mathcal{S}$). We reduce the connectivity question to one for a formula with IHSB− relations. Since satisfiability of Horn formulas is decidable in polynomial time and every connected component of a Horn relation is a Horn relation by Lemma 4.1, it is easy to decide for a given clause and a given connected component of its corresponding relation (the relation obtained after identifying repeated variables) whether there exists a solution for which the variables in this clause are assigned a value in the specified connected component. If there exists a clause for which there is more than one connected component for which solutions exist, then the space of solutions is disconnected. This follows from the fact that the projection of $G(\phi)$ onto the hypercube corresponding to the variables appearing in

this clause is disconnected. Therefore we can assume that the relation corresponding to every clause has a single connected component. Since that component is IHSB− the relation itself is IHSB−. □

It is still open whether CONN is coNP-complete for every remaining Horn set of relations, i.e., every set of Horn relations that contains at least one relation that is not componentwise IHSB−. Following the same line of reasoning as in the proof of our structural expressibility theorem, we are able to show that one of the paths of length 4 defined in section 3.2, namely, $M(\bar{x}_1, \bar{x}_2, x_3)$, can be expressed structurally from every such set of relations. Thus the trichotomy would be established if one shows that CONN($\{M(\bar{x}_1, \bar{x}_2, x_3)\}$) is coNP-hard.

5. Discussion and open problems. In section 2, we conjectured a trichotomy for CONN($\mathcal{S}$). In view of the results established here, what remains is to pinpoint the complexity of CONN($\mathcal{S}$) when $\mathcal{S}$ is Horn but not componentwise IHSB−, and when $\mathcal{S}$ is dual Horn but not componentwise IHSB+. We conjecture that for those cases CONN($\mathcal{S}$) is coNP-complete.

We mentioned that one could also consider CNF($\mathcal{S}$)-formulas without constants, and the extension of our results to this setting is still an open problem. A more interesting and challenging direction is the extension of our results to larger domains.

Finally, our techniques may shed light on other connectivity-related problems, such as approximating the diameter and counting the number of components, or proving hardness of other configuration connectivity problems. An example of the latter appears in recent work of Ito et al. [18].

REFERENCES

[1] D. ACHLIOPTAS AND F. RICCI-TERSENGHI, *On the solution-space geometry of random constraint satisfaction problems*, in Proceedings of the 38th Annual ACM Symposium on Theory of Computing, 2006, pp. 130–139.

[2] D. ACHLIOPTAS, P. BEAME, AND M. MOLLOY, *Exponential bounds for DPLL below the satisfiability threshold*, in Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, 2004, pp. 132–133.

[3] D. ACHLIOPTAS, A. NAOR, AND Y. PERES, *Rigorous location of phase transitions in hard optimization problems*, Nature, 435 (2005), pp. 759–764.

[4] G. BIROLI, R. MONASSON, AND M. WEIGT, *A variational description of the ground state structure in random satisfiability problems*, Eur. Phys. J. B, 14 (2000), pp. 551–568.

[5] E. BÖHLER, N. CREIGNOU, S. REITH, AND H. VOLLMER, *Playing with Boolean blocks, Part* II: *Constraint satisfaction problems*, ACM SIGACT-Newsletter, 35 (2004), pp. 22–35.

[6] P. S. BONSMA AND L. CERECEDA, *Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances*, in Mathematical Foundations of Computer Science, Springer, Berlin, 2007, pp. 738–749.

[7] G. BRIGHTWELL AND P. WINKLER, *Gibbs measures and dismantlable graphs*, J. Combin. Theory Ser. B, 78 (2000), pp. 141–166.

[8] A. BULATOV, *A dichotomy theorem for constraint satisfaction problems on a* 3*-element set*, J. ACM, 53 (2006), pp. 66–120.

[9] L. CERECEDA, J. VAN DEN HEUVEL, AND M. JOHNSON, *Finding paths between* 3*-colourings*, submitted (2007).

[10] N. CREIGNOU, *A dichotomy theorem for maximum generalized satisfiability problems*, J. Comput. System Sci., 51 (1995), pp. 511–522.

[11] N. CREIGNOU AND H. DAUDÉ, *Combinatorial sharpness criterion and phase transition classification for random CSPs*, Inform. and Comput., 190 (2004), pp. 220–238.

[12] N. CREIGNOU AND M. HERMANN, *Complexity of generalized satisfiability counting problems*, Inform. and Comput., 125 (1996), pp. 1–12.

[13] N. CREIGNOU AND B. ZANUTTINI, *A complete classification of the complexity of propositional abduction*, SIAM J. Comput., 36 (2006), pp. 207–229.

[14] N. CREIGNOU, S. KHANNA, AND M. SUDAN, *Complexity Classification of Boolean Constraint*

Satisfaction Problems, SIAM Monogr. Discrete Math. Appl. 7, SIAM, Philadelphia, 2001.

[15] T. Feder and M. Y. Vardi, *The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory*, SIAM J. Comput., 28 (1998), pp. 57–104.

[16] P. Gopalan, P. G. Kolaitis, E. Maneva, and C. H. Papadimitriou, *The connectivity of boolean satisfiability: Computational and structural dichotomies*, in Proceedings of the 33rd International Colloquium on Automata, Languages and Programming, 2006, pp. 346–357.

[17] R. Hearne and E. Demaine, *The nondeterministic constraint logic model of computation: Reductions and applications*, in Proceedings of the 29th International Colloquium on Automata, Languages and Programming, 2002, pp. 401–413.

[18] T. Ito, E. D. Demaine, N. J. A. Harvey, C. H. Papadimitriou, M. Sideri, R. Uehara, and Y. Uno, *On the complexity of reconfiguration problems*, in Proceedings of the 19th Annual International Symposium on Algorithms and Computation (ISAAC 2008), 2008, pp. 28–39.

[19] L. Juban, *Dichotomy theorem for the generalized unique satisfiability problem*, in Proceedings of the 12th International Symposium Fundamentals of Computation Theory, 1999, pp. 327–337.

[20] D. Kavvadias and M. Sideri, *The inverse satisfiability problem*, SIAM J. Comput., 28 (1998), pp. 152–163.

[21] S. Khanna, M. Sudan, L. Trevisan, and D. P. Williamson, *The approximability of constraint satisfaction problems*, SIAM J. Comput., 30 (2001), pp. 1863–1920.

[22] L. Kirousis and P. Kolaitis, *The complexity of minimal satisfiability problems*, Inform. and Comput., 187 (2003), pp. 20–39.

[23] R. Ladner, *On the structure of polynomial time reducibility*, J. ACM, 22 (1975), pp. 155–171.

[24] K. Makino, S. Tamaki, and M. Yamamoto, *On the boolean connectivity problem for horn relations*, in Proceedings of the 10th International Conference on Theory and Applications of Satisfiability Testing (SAT), 2007, pp. 187–200.

[25] E. Maneva, E. Mossel, and M. J. Wainwright, *A new look at survey propagation and its generalizations*, J. ACM, 54 (2007), pp. 2–41.

[26] M. Mézard and R. Zecchina, *Random k-satisfiability: From an analytic solution to an efficient algorithm*, Phys. Rev. E, 66 (2002), article 056126.

[27] M. Mézard, T. Mora, and R. Zecchina, *Clustering of solutions in the random satisfiability problem*, Phys. Rev. Lett., 94 (2005), article 197205.

[28] M. Mézard, G. Parisi, and R. Zecchina, *Analytic and algorithmic solution of random satisfiability problems*, Science, 297 (2002), pp. 812–815.

[29] M. Molloy, *Models for random constraint satisfaction problems*, SIAM J. Comput., 32 (2003), pp. 935–949.

[30] S. Reith and H. Vollmer, *Optimal satisfiability for propositional calculi and constraint satisfaction problems*, Inform. and Comput., 186 (2003), pp. 1–19.

[31] T. Schaefer, *The complexity of satisfiability problems*, in Proceedings of the 10th Annual ACM Symposium on Theory of Computing, 1978, pp. 216–226.

[32] B. Selman, H. Kautz, and B. Cohen, *Local search strategies for satisfiability testing*, in Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, D. S. Johnson and M. A. Trick, eds., DIMACS Ser. Discrete Math. Theoret. Comput. Sci. 26, AMS, Providence, RI, 1996, pp. 521–532.