The Complexity of Finding $S$-Factors in Regular Graphs

Sanjana Kolisetty  
Departments of Mathematics and EECS, CSE Division,  
University of Michigan, Ann Arbor, MI, USA  
sanjank@umich.edu

Linh Le  
Departments of Mathematics and EECS, CSE Division,  
University of Michigan, Ann Arbor, MI, USA  
likle@umich.edu

Ilya Volkovich  
Department of EECS, CSE Division, University of Michigan, Ann Arbor, MI, USA  
ilyavol@umich.edu

Mihalis Yannakakis  
Department of Computer Science, Columbia University, New York, NY, USA  
mihalis@cs.columbia.edu

Abstract

A graph $G$ has an $S$-factor if there exists a spanning subgraph $F$ of $G$ such that for all $v \in V : \deg_F(v) \in S$. The simplest example of such factor is a 1-factor, which corresponds to a perfect matching in a graph. In this paper we study the computational complexity of finding $S$-factors in regular graphs. Our techniques combine some classical as well as recent tools from graph theory.

1 Introduction

The Constraint Satisfaction Problem (CSP for short) has been a classical topic in computer science of both theoretical and practical importance. While CSPs can be quite general, in this paper we focus on the “fixed-template” Boolean CSPs. That is, CSPs over the Boolean domain where the constraints come from a fixed set of Boolean relations $\Gamma$. Formally, given a fixed set of Boolean relations $\Gamma = \{R_1, R_2, \ldots, R_m\}$, a $\Gamma$-formula is a conjunction of constraints of the form $R_j(x_{i_1}, \ldots, x_{i_n})$ where $R_j \in \Gamma$ and the $x_{i_j}$-s are propositional variables; CSP($\Gamma$) forms a decision problem where one needs to determine if a given $\Gamma$-formula is satisfiable. In other words, one needs to determine whether it is possible to satisfy all the constraints as given by the relations from $\Gamma$ simultaneously.
The object of study is the computational complexity of CSP(\(\Gamma\)) as per the choice of \(\Gamma\). In a seminal work of [22], Schaefer identified six classes of sets of Boolean relations for which CSP(\(\Gamma\)) \(\in\) P and proved that all other sets of relations generate an NP-complete problem. This result is what is known as Schaefer's Dichotomy Theorem which provides a complete classification of the computational complexity of CSP(\(\Gamma\)). The two most popular examples of applications of this theorem are the NP-completeness of the 1-in-3SAT and not-all-equal 3SAT (NAE-3SAT) problems. Subsequently, in [3], a more refined classification was presented.

While a more general Dichotomy Theorem was recently proved for non-Boolean CSPs [6, 24], there has been a large body of work dedicated to the study of the computational complexity of a restricted version of CSP(\(\Gamma\)), denoted as CSP\(_2\)(\(\Gamma\)) or CSP\(_{Edge}\)(\(\Gamma\)) [15, 10, 7, 14, 11, 8, 17]. Formally introduced by Feder in [10], CSP\(_2\)(\(\Gamma\)) corresponds to a specialization of CSP(\(\Gamma\)) to the instances where each variable appears at most twice. Alternatively, one can think about embedding the input \(\Gamma\)-formula into a graph, such that edges correspond to variables and nodes to constraints, and the constraint satisfaction problem asks for a spanning subgraph such that the set of its edges at each node satisfies the constraint at the node.

The main subtlety is that if CSP(\(\Gamma\)) \(\in\) P then, clearly, CSP\(_2\)(\(\Gamma\)) \(\in\) P. However, in the general NP-hard instances there is usually no restriction of the number of appearances of a variable. Therefore, a proof that CSP(\(\Gamma\)) is NP-hard may not carry over to CSP\(_2\)(\(\Gamma\)). In particular, if we consider the aforementioned examples, CSP\(_2\)(1-in-3SAT) corresponds to determining existence of a perfect matching in a 3-regular graph, which is decidable in polynomial time. In addition, CSP\(_2\)(NAE-3SAT) is “trivial” since every read-twice\(^2\) NAE-3SAT-formula is always satisfiable!\(^3\)

Despite all the invested effort, we are still far away from the ultimate goal. Indeed, the known results do not even provide a complete classification for the cases when \(\Gamma\) consists of just a single relation! A natural focus taken in [15], was to consider the sets \(\Gamma\) that consist of symmetric relations as these instances often arise more naturally in the graph context, because incident edges to a node are typically treated symmetrically in graph theory. This class of problems can be regarded as generalized matchings. In this paper we give a complete classification of the computational complexity of CSP\(_2\)(\(\Gamma\)), where \(\Gamma\) consists of a single symmetric relation.

In turns out that the problem has a very natural interpretation in terms of finding “S-factors” of regular graphs. We say that a graph \(G\) has an S-factor, if there exists a spanning subgraph \(F\) of \(G\) such that for all \(v \in V\) : \(\deg_F(v) \in S\). The simplest example of such factor is a 1-factor, which corresponds to a perfect matching in a graph.

### 1.1 Results

In light of the natural interpretation of the problem in terms of finding S-factors of graphs, we present our main results using that terminology. The CSP versions of the main results (and their proofs) can be found in Section 4. Our first result is a Dichotomy Theorem for regular graphs of even degree. The Dichotomy is obtained by classifying all the tractable cases.

\(^1\) Affirming what was known as the CSP Dichotomy Conjecture formulated in [12].

\(^2\) A formula in which every variable appears at most twice.

\(^3\) This follows immediately from Tutte’s Theorem (Lemma 8), however there is a more direct way to see that.
Theorem 1. Let $\ell \in \mathbb{N}$. There is a polynomial-time algorithm that given a $2\ell$-regular graph $G$ as an input, finds an $S$-factor in $G$, if there is one, in the following four cases:
1. $S$ contains an even number.
2. $\ell \in S$.
3. $\{\ell - 1, \ell + 1\} \subseteq S$.
4. $S = \{p, p + 2, \cdots, p + 2r\}$ for some $p, r \geq 0$.
Otherwise, finding an $S$-factor is NP-Hard.

As could be observed, all the tractable cases reduce to the case of finding a perfect matching in a graph (see Section 2.1 for details). Consequently, an algorithm for perfect matching in graphs (e.g. [9]) could be used to find these $S$-factors, for the “yes”-instances of the problem. For regular graph of odd degree, we obtain a somewhat weaker result: we show that for each set $S$, the decision problem is either polynomial-time solvable or NP-hard, yet we are unable to classify explicitly all the tractable cases. Closing this gap will require resolving several conjectures in graph theory (see [2, 19, 1, 5] for more details).

Theorem 2. Let $\ell \in \mathbb{N}$. There is a polynomial-time algorithm that given a $(2\ell + 1)$-regular graph $G$ as an input, decides if $G$ has an $S$-factor, in the following two cases:
1. Every $(2\ell + 1)$-regular graph has an $S$-factor.
2. $S = \{p, p + 2, \cdots, p + 2r\}$ for some $p, r \geq 0$.
Otherwise, deciding if $G$ has an $S$-factor is NP-Hard.

There are specific sets $S$, for which it is an open problem in graph theory whether every $(2\ell + 1)$-regular graph has an $S$-factor. A simple concrete example is the case of $S = \{1, 4\}$ for degree-5 graphs (the conjecture in this case is that there is always an $S$-factor). The theorem tells us that, even though we may not know the answer to the open problem for a particular $S$, if it does not hold trivially for all graphs and there is a counterexample, then the corresponding $S$-factor problem is NP-hard; that is, there is a way to use any counterexample (as a black box) to generate an NP-hardness reduction.

1.2 Comparison to Previous Results

In [15], Istrate studied the special case when $\Gamma$ consists of symmetric relations. In that work, several “patterns” for which CSP$_2(\Gamma) \in \mathbb{P}$ were identified. In particular, one such pattern corresponds to Case 4 of Theorem 1. This result was obtained via connections to covering problems. In addition, Istrate formulated a sufficient condition under which the computational complexity of CSP$_2(\Gamma)$ and CSP(\Gamma) is the same, with the additional “constants for free” assumption. That is, one can fix some variables to either 0 and 1 (for more details, see Lemma 26 and the preceding discussion). Later on, Feder [10], extended the condition to non-symmetric relations, introducing Delta Matroids, and showed that if $\Gamma$ contains some relation that is not a Delta matroid then CSP$_2(\Gamma)$ and CSP(\Gamma) have the same complexity (in the presence of constants). Several subsequent works [7, 8, 17] introduced further refinements to Delta Matroids. Yet, “constants for free” remained a prevalent assumption in these and other CSP-related works. Nonetheless, even with the assumption, no classification for the mere case of a single symmetric relation was known prior to our work.

We also would like to point out that the “constants for free” assumption is implicitly equivalent to adding two more relations $P(x) = x$ and $Q(x) = \neg x$ to $\Gamma$. It is important to stress that adding these relations can completely tilt the scale. For example, consider a single 8-ary symmetric relation “two or six out of eight”. Formally, $R(x) = 1$ iff $w_H(x) = 2$ or 6. In the graphical perspective, this corresponds to the problem of finding a $\{2, 6\}$-factor of an 8-regular graph. Now by Tutte’s Theorem (Lemma 8), every 8-regular graph has a 2-factor. Hence CSP$_2(R) \in \mathbb{P}$ in a “trivial” way. On the other hand, CSP$_2(\{R, x, \neg x\})$ is NP-hard.
The Complexity of Finding $S$-Factors in Regular Graphs

(follows e.g. from [15]). Our results do not rely on the “constants for free” assumption. In fact, they complement it: roughly speaking, we show that either CSP$_2(\Gamma) \in P$ or there exist $\Gamma$-formulas that “implement” the relations $x$ and $\neg x$. See Lemmas 29 and 30 for more details.

There is, of course, extensive work in graph theory on factors in graphs, (see e.g. the surveys [2, 21] and references therein), with the development of a rich theory of matchings, as well as more general factors. This includes structural results on the existence of factors, starting from Petersen’s theorem from 1891 [20]; algorithmic results, including e.g. Edmonds’ matching algorithm [9] and its extensions and refinements; and hardness results, starting e.g. with Lovász’s theorem [18] that for any $a, b \in \mathbb{N}$ such that $1 \leq a \leq b - 3$, the problem of deciding whether a graph has an $\{a, b\}$-factor is NP-hard even for simple graphs (not necessarily of a given, regular degree). We will leverage several of these graph theoretic results on (generalized) matchings and the existence of suitable factors in graphs. We review some of these theorems that we use in the next section.

2 Preliminaries

Definition 1 (Zebras and Holes). Let $S \subseteq \mathbb{N}$ be a subset of $\mathbb{N}$. Following [15], we say that $S$ contains a hole of size $t$ if there exist $i$ such that: $i, i+t+1 \in S$ and $[i+1, i+t] \cap S = \emptyset$. Let $a \leq b \in \mathbb{N}$ such that $a \equiv b \pmod{2}$. We say that $S$ is an $(a, b)$-zebra if $S = \{a, a+2, a+4, \ldots, b\}$. We call a set $S$ a zebra, if it is an $(a, b)$-zebra for some $a, b \in \mathbb{N}$.

Remark. A set $S = \{a\}$ also constitutes a zebra since it is an $(a, a)$-zebra. The following is a simple observation about the structure of finite subsets of $\mathbb{N}$, that will be useful for us later.

Observation 2. Let $S \subseteq \mathbb{N}$ be a finite, non-empty subset of $\mathbb{N}$ then (at least) one of the following holds:
- $S$ contains two consecutive numbers.
- $S$ is zebra.
- $S$ contains a hole of size at least 2.

2.1 Graphs

In this paper we consider graphs $G = (V, E)$. Unless specified otherwise, all the graphs considered in the paper are general graphs (i.e. with self-loops and parallel edges). The focus of this paper is the complexity of finding a particular kinds of subgraphs in graphs, known as factors. We define this formally now.

Definition 3 (Factors). Let $G = (V, E)$ be a graph with $V$ vertices and $E$ edges. [2]
1. $H$-factor: Let $H$ be a set function associated with $G$ that maps $V \to 2^\mathbb{N}$. We say that $G$ has an $H$-factor if there exists a spanning subgraph $F$ of $G$ such that for all $v \in V : \deg_F(v) \in H(v)$.
2. $f$-factor is a specialization to the case when $\forall v \in V : H(v) = \{f(v)\}$, for some function $f : V(G) \to \mathbb{Z}_+$.
3. $S$-factor is a specialization to the case when $H(v) = S$ for all $v \in V$, for some fixed set $S \subseteq \mathbb{N}$.
4. $[a, b]$-factor is a further specialization to the case when $S$ is the interval $[a, b]$.
5. $k$-factor is a further specialization to the case when $S = \{k\}$.

The simplest case of a graph factor is a 1-factor which corresponds to a perfect matching of a graph. This problem has a well-known efficient algorithm known as “Blossom Algorithm.”
There exists a polynomial-time algorithm that given a graph \(G = (V, E)\) as an input, outputs a 1-factor \(F\) of \(G\), if one exists.

The algorithm can be easily extended to handle \(f\)-factors due to the following observation:

There exists a polynomial-time algorithm that given a graph \(G = (V, E)\) and a function \(f : V \rightarrow \mathbb{Z}\) (as a vector) as an input, outputs a graph \(G'\) such that \(G'\) has a 1-factor \(F'\) iff \(G\) has an \(f\)-factor \(F\). Moreover, \(F\) can be computed in polynomial time given \(F'\).

In addition, using the simple idea of [16] of introducing self-loops, the algorithm can be further extended to \(H\)-factors, where each \(H(v)\) is a zebra (See Definition 1).

There exists a polynomial-time algorithm that given a graph \(G = (V, E)\) and a function \(H : V \rightarrow 2^\mathbb{N}\), where each \(H(v)\) is zebra, as an input, outputs a graph \(G'\) and a function \(f : V \rightarrow \mathbb{Z}\) such that \(G'\) has a \(f\)-factor \(F'\) iff \(G\) has an \(H\)-factor \(F\). Moreover, \(F\) can be computed in polynomial time given \(F'\).

The following is immediate given the above reductions to the perfect matching case.

There exists a polynomial-time algorithm that given a graph \(G = (V, E)\) and a function \(H : V \rightarrow 2^\mathbb{N}\), where each \(H(v)\) is zebra, as an input, outputs an \(H\)-factor \(F\) of \(G\), if one exists.

We note that an efficient algorithm for this kind of \(H\)-factors has been obtained in [15] using a different argument. Recently in [17], the algorithm was extended to also handle the “asymmetric” version. Next, we require the following results regarding the existence of regular factors in regular graphs.

\[23\] Let \(r, k \in \mathbb{N}\) such that \(1 \leq k \leq r - 1\). Then any \(r\)-regular graph has a \(\{k, k + 1\}\)-factor.

\[20\] Let \(r\) and \(k\) be even integers such that \(2 \leq k \leq r\). Then any \(r\)-regular graph has a \(k\)-factor.

\[13\] Suppose \(r\) is even and \(\frac{r}{2}\) is odd. Then any connected \(r\)-regular graph of even order has a \(\frac{r}{2}\)-factor.

As a corollary we obtain the following, which was observed for simple graphs in [1]. We also note that the proof of [1] is merely existential whereas our proof is algorithmic.

Let \(r \in \mathbb{N}\) such that both \(r\) and \(\frac{r}{2}\) are even. Then any \(r\)-regular graph of even order has a \(\\{\frac{r}{2} - 1, \frac{r}{2} + 1\}\)-factor.

Proof. Let \(G = (V, E)\) be a graph satisfying the preconditions. For every \(v \in V\), we add a self-loop. Call this new resulting graph \(G' = (V, E')\). Observe that \(G'\) is a \((r + 2)\)-regular graph of even order and \(\frac{r + 2}{2} = \frac{r}{2} + 1\) is odd. Therefore, by Lemma 8, \(G'\) has a \((\frac{r}{2} + 1)\)-factor.

Now, consider two cases: if \(v \in V\) uses the self-loop to fulfill its factor, then the induced degree of \(v\) in \(G\) is \(\frac{r}{2} - 1\). Otherwise, the induced degree of \(v\) in \(G\) is \(\frac{r}{2} + 1\).

2.2 Boolean Relations

\[10\] The Hamming Weight of a vector \(\bar{v} \in \{0, 1\}^n\) is defined as: \(w_H(\bar{v}) \triangleq |\{i \mid v_i \neq 0\}|.\) That is, the number of its non-zero coordinates.
Definition 11 (Symmetric Relation). We say that an m-ary relation \( R(x_1, x_2, \ldots, x_m) \) is symmetric if there exists a set \( \text{Spec}(R) \subseteq \{0 \ldots m\} \) such that \( R(\bar{x}) = 1 \) if and only if \( w_H(\bar{x}) \in \text{Spec}(R) \). The set \( \text{Spec}(R) \) is called the spectrum of \( R \).

The following are examples of particular symmetric relations we will be utilizing.

Example 12.

1. \( \text{NE}(x_1, x_2) \) is a binary relation with \( \text{Spec}(\text{NE}) = \{1\} \).
2. Let \( k \in \mathbb{N} \). \( \text{EQ}_k \) is a k-ary relation with \( \text{Spec}(\text{EQ}_k) = \{0, k\} \).

Definition 13 (Dual Relation). Let \( R(x_1, \ldots, x_m) \) be a relation. We define the dual relation of \( R \) as:

\[
R^*(x_1, \ldots, x_m) \overset{\triangleq}{=} R(\lnot x_1, \lnot x_2, \ldots, \lnot x_m).
\]

The following observation is immediate with respect to symmetric relations.

Observation 14. For a symmetric m-ary relation \( R \) we have: \( \text{Spec}(R^*) = \{m - i \mid i \in \text{Spec}(R)\} \).

2.2.1 \( \Gamma \)-Instances, \( \text{CSP}(\Gamma) \), Triviality

In what follows, let \( \Gamma = \{R_1, R_2, \ldots, R_\ell\} \) be a fixed set of Boolean relations. We will use \( \Gamma^* \) to denote the set of dual relations. Formally, \( \Gamma^* \overset{\triangleq}{=} \{R_1^*, R_2^*, \ldots, R_\ell^*\} \), where \( R_i^* \) is the dual relation of \( R_i \).

Definition 15. A \( \Gamma \)-instance or \( \Gamma \)-formula \( \Phi \) is a conjunction of constraints of the form \( R_j(x_{i_1}, \ldots, x_{i_m}) \) where \( R_j \in \Gamma \) and the \( x_{i_j} \)'s are propositional variables. The read of a variable \( x_i \) in \( \Phi \) is the number of occurrences of \( x_i \) in \( \Phi \). The read of a formula \( \Phi \) is the maximal read of a variable in it.

In this paper we will focus on read-twice formulas, that is formulas in which all the variables appear at most two times. We now formally introduce the main problem we will study.

Problem 16. \( \text{CSP}(\Gamma) \) forms a decision problem where one needs to determine if a given \( \Gamma \)-formula is satisfiable. In other words, one needs to determine whether it is possible to satisfy all the constraints as given by the relations from \( \Gamma \), simultaneously. For \( k \geq 1 \), \( \text{CSP}_k(\Gamma) \) is a specialization of \( \text{CSP}(\Gamma) \) to read-\( k \) instances. If \( \Gamma = \{R\} \) has a single relation \( R \), we will write \( \text{CSP}(R) \) and \( \text{CSP}_k(R) \).

As was pointed out in the introduction, in this paper we are interested in the computational complexity of \( \text{CSP}_2(\Gamma) \), as per the choice of \( \Gamma \). We now recall Schaefer’s Dichotomy Theorem [22].

Lemma 17 ([22]). \( \text{CSP}(\Gamma) \in \mathbb{P} \) in the following six cases:

1. \( \forall R_j \in \Gamma : R_j(0) = 1 \)
2. \( \forall R_j \in \Gamma : R_j(\bar{1}) = 1 \)
3. \( \forall R_j \in \Gamma : R_j \) is equivalent to a conjunction of binary relations
4. \( \forall R_j \in \Gamma : R_j \) is equivalent to a conjunction of Horn clauses
5. \( \forall R_j \in \Gamma : R_j \) is equivalent to a conjunction of dual-Horn clauses
6. \( \forall R_j \in \Gamma : R_j \) is equivalent to a conjunction of affine forms

Otherwise, \( \text{CSP}(\Gamma) \) is \( \text{NP-Hard} \).

The following is an instantiation of the Theorem to the case of a single symmetric relation.
Corollary 18. Let $R(x_1, \ldots, x_m)$ be a symmetric relation. Then $\text{CSP}(R) \in \mathsf{P}$ in the following cases:

1. $R(\emptyset) = 1$
2. $R(\{\}\Gamma) = 1$
3. $m \leq 2$
4. $\text{Spec}(R)$ contains all odd numbers in $\{1, \ldots, m\}$

Definition 19 (Triviality). We say that $\text{CSP}_k(\Gamma)$ is trivial if every instance of $\text{CSP}_k(\Gamma)$ (i.e. every read-twice $\Gamma$-instance) is satisfiable.

To put the above definitions into a context, observe the first two of the six tractable classes correspond to cases when $\text{CSP}(\Gamma)$ and $\text{CSP}(\Gamma^*)$ are trivial. Similarly, observe that Cases 4 and 5 correspond to the same conditions applied to both $\text{CSP}(\Gamma)$ and $\text{CSP}(\Gamma^*)$. With some extra work, you can see that the same holds true for Cases 3 and 6. In the same vein, the following lemma is immediate from the definition.

Lemma 20. 1. $\text{CSP}_1(\Gamma)$ is trivial, as long as $\Gamma$ does not contain a contradiction.
2. For any $k \in \mathbb{N}$ and any $\Gamma$: $\text{CSP}_k(\Gamma)$ is trivial iff $\text{CSP}_k(\Gamma^*)$ is trivial.

2.2.2 Induced Relations and Implementation

Definition 21 (Induced relation). For a relation $R(\bar{x}, \bar{y})$ we define the induced relation $\exists y R(\bar{x}, \bar{y})$ on $\bar{x}$ as

\[ \exists y R(\bar{x}, \bar{y}) = 1 \iff \exists y \text{ such that } R(\bar{x}, \bar{y}) = 1. \]

Definition 22 (Implementation). Let $R(\bar{x})$ be an arbitrary relation. We say that $\Gamma$ implements $R$, denoted by $\Gamma \text{ imp } R$, if there exists a $\Gamma$-instance $\Phi(\bar{x}, \bar{y})$ such that $R(\bar{x}) = \exists y \Phi(\bar{x}, \bar{y})$. Furthermore, we say that $\Gamma$ read-twice-implements $R$, denoted by $\Gamma \text{ imp}_2 R$, if in addition:

1. Each $x_i$ is read-once in $\Phi$.
2. Each $y_j$ is (at most) read-twice in $\Phi$.

The intuition behind the definition is that if $\Gamma$ read-twice-implements $R$ then we can, effectively, consider the set $\Gamma \cup \{R\}$ instead of $\Gamma$. The following lemma summarizes this intuition and will be used implicitly in our proofs.

Lemma 23. Let $R$ be a relation such that $\Gamma$ read-twice-implements $R$ and let $\Gamma' \supseteq \Gamma \cup \{R\}$. Then for every read-twice $\Gamma'$-instance $\Phi'(\bar{x})$ there exists a read-twice $\Gamma$-instance $\Phi(\bar{x}, \bar{y})$ such that $\Phi'(\bar{x}) = \exists y \Phi(\bar{x}, \bar{y})$.

The following lemma showcases this intuition further, by showing that a three-way Equality $\text{EQ}_3$ can be used to implement $k$-way equality $\text{EQ}_k$ for any $k \geq 3$. Conversely, one can use $k$-way equality $\text{EQ}_k$ to implement $k'$-way equality $\text{EQ}_{k'}$ for any $k' \leq k$.

Lemma 24. If $\Gamma$ read-twice-implements $\text{EQ}_3$ then $\Gamma$ read-twice-implements $\text{EQ}_k$, for any $k \geq 3$.

Proof. By induction on $k$. The base case $k = 3$ is trivial. Let $\Phi_k$ denote a $\Gamma$-instance that read-twice-implements $\text{EQ}_k(x_1, \ldots, x_k)$. Given $\Phi_k$, we can read-twice implement $\text{EQ}_{k+1}(x_1, \ldots, x_{k+1})$ in the following way:

\[ \Phi_k(x_1, \ldots, x_{k-1}, y) \land \Phi_3(y, x_k, x_{k+1}). \]

Given our inductive hypothesis and the fact that we can read-twice implement $\text{EQ}_3$, we can conclude that $\Gamma$ read-twice-implements $\text{EQ}_{k+1}$.
The Complexity of Finding $S$-Factors in Regular Graphs

We note this was already observed in [15, 10]. Similar ideas can be used to show that if $\Gamma$ read-twice-implements particular relations, then read-twice $\Gamma$-formulas exhibit some interesting closure properties.

Lemma 25 (Read-Twice Implementing Particular Relations).
1. Closure Under Variable Negation: Suppose $\Gamma$ read-twice-implements NE. Then read-twice $\Gamma$-formulas are closed under variable negation. Formally, if $\Gamma \text{imp}_1 R(x, y)$ then $\Gamma \text{imp}_1 R(\neg x, y)$.
2. Closure Under Setting Variables to Constants: Suppose $\Gamma$ read-twice-implements $x$ or $\neg x$. Then read-twice $\Gamma$-formulas are closed under setting variables to either 1 or 0, respectively. Formally, if $\Gamma \text{imp}_2 R(x, y)$ then $\Gamma \text{imp}_2 R(1, y)$ or $R(0, y)$, respectively.
3. Closure Under Variable Repetition: Suppose $\Gamma$ read-twice-implements EQ. Then read-twice $\Gamma$-formulas are closed under repetition of any variable arbitrary number of times.

We will use the above implicitly. We finish this section with the following simple observation from [15].

Lemma 26 ([15]). Let $R$ be a symmetric relation such that $\text{Spec}(R)$ contains a hole of size at least 2. Then $\{R, x, \neg x\}$ read-twice-implements EQ.

We note that Feder [10] extended this claim to a non-symmetric case defining Delta Matroids. In the same paper it was observed that WLOG every variable in a read-twice formula occurs exactly twice. Furthermore, such formulas have very natural interpretation as graphs where edges play the role of variables and nodes the role of constraints.

Lemma 27 (Graphical Perspective of CSP$_2$ [10]). Every read-twice formula can be efficiently transformed into an exact read-twice formula, and furthermore viewed as a graph.

3 Main Technical Tools

In this section we present our main technical tools, which we will use to prove Theorems 1 and 2. We begin by showing that the sets $\Gamma$ for which CSP$_2(\Gamma)$ is non-trivial (in the sense of Definition 19), read-twice implement NE$(x, y)$ or $\{x, \neg x\}$. Consequently, by Lemma 25, such read-twice $\Gamma$-formula are closed under variable negation or setting variables to constants $\{0, 1\}$. Note that the result holds for general relations (not necessarily symmetric).

Lemma 28. Suppose that CSP$_2(\Gamma)$ is non-trivial. Then $\Gamma$ read-twice-implements NE$(x, y)$ or $\{x, \neg x\}$.

Proof. Since CSP$_2(\Gamma)$ is non-trivial, there exists an unsatisfiable read-twice $\Gamma$-instance $\Phi$. On the other hand, recall (e.g. Lemma 20) that any read-once $\Gamma$-instance is satisfiable. Consider the “unpaired” version $\Phi'$ of $\Phi$. Formally, for each variable $x_i$ we replace one of the occurrences with a fresh new variable $y_i$. Observe that the resulting $\Phi'$ is read-once and hence satisfiable. Now, consider the process of gradually pairing the variables of $\Phi'$, that will eventually recover $\Phi$. Formally, $\Phi'_0 \triangleq \Phi'$. $\Phi'_i$ results from $\Phi'_0$ by setting $x_1 = y_1$. More generally, $\Phi'_i$ results from $\Phi'_{i-1}$ by setting $x_i = y_i$. As $\Phi'_0 = \Phi'$ is satisfiable and $\Phi'_n = \Phi$ is not, by a hybrid argument, there exist $i$ such that $\Phi'_{i-1}$ is satisfiable and $\Phi'_i$ is not. Let $\varphi(x_i, y_i)$ be the relation given by $\Phi'_{i-1}$ induced to the variables $x_i$ and $y_i$ (Recall Definition 21). By the above, $\varphi(0, 0) = \varphi(1, 1) = 0$ and either $\varphi(0, 1) = 1$ or $\varphi(1, 0) = 1$ (or both). Consider three cases:
Lemma 29. Let $R$ be a symmetric $2\ell$-ary relation such that: $\ell \notin \text{Spec}(R)$ and $\{\ell - 1, \ell + 1\} \notin \text{Spec}(R)$. Then $\{R, NE\}$ read-twice-implements $\text{EQ}_3$ or $\{x, \neg x\}$.

Proof. We define the following two sets: $S_- = \{a \mid R(\ell - a) = 1\}$ and $S_+ = \{a \mid R(\ell + a) = 1\}$. Furthermore, let $a_- = \min S_-$ and $a_+ = \min S_+$. We define $a_-$ or $a_+$ to be infinity if $S_-$ or $S_+$ is empty, respectively. We consider three cases:

Case 1: $a_+ = a_-$. Observe that $a_- \geq 2$. Using NE and Lemma 25, we plug $\ell - a_-$ pairs $z_i, \neg z_i$ into the relation $R$. Formally, consider,

$$R(z, \bar{y}) \triangleq R(z_1, \neg z_1, \ldots, z_{\ell-a_-}, \neg z_{\ell-a_-}, y_1, \ldots, y_{2a_-}).$$

By definition, $w_H(z) = \ell - a_-$ and $0 \leq w_H(\bar{y}) \leq 2a_-$. Now, since $a_+ = a_-:

$$R(z, \bar{y}) = 1 \iff w_H(\bar{y}) \in \{0, 2a_-\}.\]$$

Consequently, $\exists z R(z, \bar{y}) = \text{EQ}_3(\bar{y})$, where $k = 2a_- \geq 4$.

Case 2: $a_+ > a_-$. Observe that $a_- \geq 1$ and consider $R(z, \bar{y})$ as above. Now, however, since $a_+ > a_-:

$$R(z, \bar{y}) = 1 \iff w_H(\bar{y}) = 0.\]$$

Hence, we obtain $\neg y_i$. Using NE, we can obtain $y_i$.

Case 3: $a_- > a_+$. Observe that $a_+ \geq 1$. We repeat the argument of Case 2 for the dual relation $R^*$ of $R$. As $R^*$ read-twice-implements $\{x, \neg x\}$, so does $R$.

We use a similar argument for relations of odd arity.

Lemma 30. Let $R$ be a symmetric $2\ell + 1$-ary relation such that: $\{\ell, \ell + 1\} \notin \text{Spec}(R)$. Then $\{R, NE\}$ read-twice-implements $\text{EQ}_3$ or $\{x, \neg x\}$.

Proof. We define the following two sets: $S_- = \{a \mid R(\ell - a) = 1\}$ and $S_+ = \{a \mid R(\ell + 1 + a) = 1\}$. Furthermore, let $a_- = \min S_-$ and $a_+ = \min S_+$. We define $a_-$ or $a_+$ to be infinity if $S_-$ or $S_+$ is empty, respectively. We consider three cases:

Case 1: $a_+ = a_-$. Observe that $a_- \geq 1$. Consider,

$$R(z, \bar{y}) \triangleq R(z_1, \neg z_1, \ldots, z_{\ell-a_-}, \neg z_{\ell-a_-}, y_1, \ldots, y_{2a_-+1}).$$

By definition, $w_H(z) = \ell - a_-$ and $0 \leq w_H(\bar{y}) \leq 2a_- + 1$. Now, since $a_+ = a_-:

$$R(z, \bar{y}) = 1 \iff w_H(\bar{y}) \in \{0, 2a_- + 1\}.\]$$

Consequently, $\exists z R(z, \bar{y}) = \text{EQ}_3(\bar{y})$, where $k = 2a_- + 1 \geq 3$.

Case 2: $a_+ > a_-$. Observe that $a_- \geq 0$ and consider $R(z, \bar{y})$ as above. Now, however, since $a_+ > a_-:

$$R(z, \bar{y}) = 1 \iff w_H(\bar{y}) = 0.\]$$

Hence, we obtain $\neg y_i$. Using NE, we can obtain $y_i$.

Case 3: $a_- > a_+$. Observe that $a_+ \geq 0$. We repeat the argument of Case 2 for the dual relation $R^*$ of $R$. As $R^*$ read-twice-implements $\{x, \neg x\}$, so does $R$.\]
4 Characterization Proof

In this section we give our main results, thus proving Theorems 1 and 2.

Theorem 31 (Characterization of Even-Arity Relations). Let \( R \) be a symmetric \( 2\ell \)-ary relation which is not constantly false. Then \( \text{CSP}_2(R) \in \mathbb{P} \) in the following four cases:

1. There is an even \( k \in \text{Spec}(R) \).
2. \( \ell \in \text{Spec}(R) \).
3. \( \{\ell - 1, \ell + 1\} \subseteq \text{Spec}(R) \).
4. \( \text{Spec}(R) \) is a zebra.

Otherwise, \( \text{CSP}_2(R) \) is \( \text{NP} \)-Hard.

Proof. For Cases 1–4, we will take the graphical perspective (Lemma 27). Indeed, the problem corresponds to finding an \( S \)-factor of a given \( 2\ell \)-regular graph, where \( S = \text{Spec}(R) \).

1. Follows from Item 2 of Lemma 8.
2. We can assume WLOG that \( S \) contains only odd numbers. In particular, \( \ell \) is odd. Consider the following algorithm, given a graph \( G \) as an input.
   - Find all the connected components \( C_1, C_2, \ldots, C_t \) of \( G \).
   - If each \( C_i \) is of even order, return “true”; otherwise, return “false”.

   Analysis: If each \( C_i \) is of even order, then by Lemma 8, each \( C_i \) has an \( \ell \)-factor and so does \( G \). Conversely, suppose some \( C_i \) is of odd order. Then by Handshaking Lemma, \( C_i \) cannot have an \( S \)-factor, as otherwise the overall sum of the degrees will be odd.
3. As before, we can assume WLOG that \( S \) contains only odd numbers. Hence, \( \ell \) is even. Apply the procedure outlined in the proof of Lemma 9. This will reduce the problem to the previous case.
4. Apply Corollary 7 with \( H(v) = \text{Spec}(R) \) for every vertex \( v \) in the graph.

For the \( \text{NP} \)-Hardness proof, we take the CSP view of the problem. We show that if none of the Cases 1–4 hold, then \( \text{CSP}_2(R) \) is as hard as \( \text{CSP}(R) \). That is, we can lift the restriction on the read. The hardness then follows from Schaefer’s Dichotomy Theorem instantiated to a single symmetric relation - Corollary 18.

Claim 32. If \( \text{Spec}(R) \) does not fall into any of the four cases, then \( \text{Spec}(R) \) contains a hole of size at least 2 and \( \{ R \} \) read-twice-implements \( \text{EQ}_3 \).

Proof. First, observe that \( \text{Spec}(R) \) cannot have two consecutive numbers (as one of them will be even) and is not a zebra (Case 4). Therefore, by Observation 2, \( \text{Spec}(R) \) must contain a hole of size at least 2.

Next, consider the relation:

\[
N(x, y) \overset{\Delta}{=} \exists z R(z_1, z_1, z_2, z_2, \ldots, z_{\ell-1}, z_{\ell-1}, x, y).
\]

Since \( \text{Spec}(R) \) does not contain even numbers (Case 1), \( N(x, y) = \text{NE}(x, y) \). Thus, by Lemma 29 given Cases 2 and 3, \( \{ R \} \) read-twice-implements \( \text{EQ}_3 \) or \( \{ x, \neg x \} \). In the former case, the claim follows. In the latter case, Lemma 26 completes the proof of the claim.

In conclusion, \( \text{CSP}_2(R) \) is as hard as \( \text{CSP}(R) \) and is thus \( \text{NP} \)-Hard by Corollary 18.

For symmetric relations of odd arity, we obtain a somewhat weaker result.
Theorem 33 (Characterization of Odd-Arity Relations). Let $R$ be a symmetric $(2\ell + 1)$-ary relation which is not constantly false. Then $\text{CSP}_2(R) \in \mathbb{P}$ in the following cases:

1. $\text{CSP}_2(R)$ is trivial.
2. $\text{Spec}(R)$ is a zebra.

Otherwise, $\text{CSP}_2(R)$ is $\text{NP}$-Hard.

Proof. Case 1 is trivial and Case 2 follows from Corollary 7. For the $\text{NP}$-Hardness proof, we use a similar argument as in Theorem 31 to conclude that $\text{CSP}_2(R)$ is as hard as $\text{CSP}(R)$. Here is the high-level idea:

- $\{R\}$ read-twice-implements $\text{NE}(x, y)$ or $\{x, \neg x\}$ - Lemma 28.
- $\text{Spec}(R)$ cannot contain two consecutive numbers - Lemma 8.
- $\text{Spec}(R)$ contains a hole of size at least 2 - Observation 2.
- $\{R\}$ read-twice-implements $\text{EQ}_3$ or $\{x, \neg x\}$ - Lemma 30.
- $\{R\}$ read-twice-implements $\text{EQ}_3$ - Lemma 26.

First observe that $\text{Spec}(R)$ cannot contain two consecutive numbers since by Lemma 8, this case is trivial, in the graphical perspective. Consequently, by Observation 2, $\text{Spec}(R)$ must contain a hole of size at least 2. In addition, by Lemma 30, $\{R\}$ read-twice-implements $\text{EQ}_3$ or $\{x, \neg x\}$. In the former case, we are done. In the latter case, Lemma 26 completes the proof.

5 Discussion & Open Questions

In this paper we obtain the first classification of the computational complexity of $\text{CSP}_2(R)$, where $R$ is a single symmetric relation. Alternatively, we obtain a classification of the complexity of the $S$-factor problem for regular graphs. The characterization is explicit for even degree graphs (even arity), while for odd degrees it states that all nontrivial cases, except for zebras, are $\text{NP}$-hard. An obvious open question is to identify for which sets $S$, an $S$-factor is always guaranteed to exist; this amounts to resolving certain open problems in graph theory, even for some small specific $S$, and looks rather challenging.

More generally, the goal of this line of research is to obtain a complete classification of the computational complexity of $\text{CSP}_2(\Gamma)$, analogous to Schaefer’s Dichotomy Theorem. While an explicit classification may encounter difficult graph-theoretic questions, even for some specific $\Gamma$, it may well be possible to prove a general complexity dichotomy theorem, as we have done here, without having to resolve explicitly all the hard graph-theoretic questions.

One can observe that all the $\text{NP}$-hardness results of $\text{CSP}_2(\Gamma)$, for the special case when $\Gamma$ consists of symmetric relation(s), are established via the route of showing that $\Gamma$ implements the Equality relation. This, in turn, allows to apply Schaefer’s Dichotomy Theorem. One interesting open question is whether there exists a set $\Gamma$ (consisting of not necessarily symmetric relations) that does not implement Equality, yet for which $\text{CSP}_2(\Gamma)$ is $\text{NP}$-Hard. This would imply that Schaefer’s Dichotomy does not cover all the cases of bounded read.

References


