HAL open science

Component twin-width as a parameter for BINARY-CSP and its semiring generalizations

Ambroise Baril, Miguel Couceiro, Victor Lagerkvist

▶ To cite this version:

Ambroise Baril, Miguel Couceiro, Victor Lagerkvist. Component twin-width as a parameter for BINARY-CSP and its semiring generalizations. 2022. hal-03739614

HAL Id: hal-03739614

https://inria.hal.science/hal-03739614v1

Preprint submitted on 27 Jul 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

arXiv:2207.12368v1 [cs.CC] 14 Jul 2022

COMPONENT TWIN-WIDTH AS A PARAMETER FOR BINARY-CSP AND ITS SEMIRING GENERALISATIONS

AMBROISE BARIL, MIGUEL COUCEIRO, AND VICTOR LAGERKVIST

Abstract. We investigate the fine-grained and the parameterized complexity of several generalizations of binary constraint satisfaction problems (BINARY-CSPs), that subsume variants of graph colouring problems. Our starting point is the observation that several algorithmic approaches that resulted in complexity upper bounds for these problems, share a common structure.

We thus explore an algebraic approach relying on semirings that unifies different generalizations of BINARY-CSPs (such as the counting, the list, and the weighted versions), and that facilitates a general algorithmic approach to efficiently solving them. The latter is inspired by the (component) twin-width parameter introduced by Bonnet et al., which we generalize via edge-labelled graphs in order to formulate it to arbitrary binary constraints. We consider input instances with bounded component twin-width, as well as constraint templates of bounded component twin-width, and obtain an FPT algorithm as well as an improved, exponential-time algorithm, for broad classes of binary constraints.

We illustrate the advantages of this framework by instantiating our general algorithmic approach on several classes of problems (e.g., the H-coloring problem and its variants), and showing that it improves the best complexity upper bounds in the literature for several well-known problems.

1. Introduction

Constraint satisfaction problems (CSPs) are rooted in artificial intelligence and operations research and provide a rich framework for encoding many different types of problems (for a wealth of applications, see e.g. the book by Rossi et al. [31]). A CSP example of paramount importance is the *Boolean satisfiability problem*, SAT, which can be viewed as a constraint satisfaction problem over the Boolean domain. Another noteworthy CSP example is the H-COLORING problem [28] that asks whether there exists an homomorphism from an (undirected) input graph G to the target graph H. Indeed, this problem can be formalized as a particular case of a CSP over binary constraints (BINARY-CSP): given a set of variables V and a set of constraints of the form $R(u_1, u_2)$ for $(u_1, u_2) \in V^2$ and binary relations $R \subseteq D^2$, the objective is to map variables in V to values in the domain D such that every such constraint is satisfied. Such a mapping is then referred as a *solution* of the instance. When the relations belong to a fixed language Γ, this problem is usually denoted by BINARY-CSP(Γ), and the decision problem is stated as the problem of deciding whether a solution exists. Clearly, the H-COLORING problems subsume the well known q-COLORING problem with H being the q-clique, which itself can be seen as the satisfaction of constraints expressed through the relation $\mathrm{NEQ}_q = \{(a, b) \in [q]^2 \mid a \neq b\}$. Hell & Nešetřil [28] showed that the complexity of H-COLORING problems for a given graph H is NP-complete whenever H is not bipartite; otherwise, it is in P. This result can be seen as a particular case of the *CSP dichotomy theorem* which states that every CSP problem (and, in particular, BINARY-CSP) is either in P or NP-complete [17, 34].

For certain applications however, the basic decidability setting provides insufficient modelling power and one may be instead interested in counting the number of solutions (we prefix the problem by # to denote the corresponding counting problem). Also, counting problems may be subjected to additional constraints such as lists, where a function $l : V \mapsto \mathcal{P}(D)$ is given and any solution f has to satisfy $\forall u \in V, f(u) \in l(u)$, and costs, where sending $u \in V$ to $v \in D$ has a cost $C(u, v)$, with the goal of minimizing $\sum_{u \in V} C(u, f(u))$. This framework makes it possible to encode phase transition systems modelled by partition functions, modeling problems such as counting q-particle Widom–Rowlinson configurations and counting Beach models, or the classical Ising model (for many more examples, see e.g. Dyer & Greenhill [24]). The complexity of these generalized problems has been the subject of intense research, starting with the complexity dichotomy for the #H-COLORING problems by Dyer & Greenhill [24], later extended to #CSP by Bulatov [17], and culminating into the dichotomy theorem by Cai & Chen [18] in the presence of weights.

In contrast to the classical CSP, which admits a rich tractability landscape [17, 34], the associated counting problems are generally hard except for trivial cases. For example, $\#H$-COLORING is #P-hard if H is not a disjoint union of looped cliques and loopless complete bipartite graphs, and is in P otherwise [24]. Thus, non-trivial templates H of actual interest tend to result in NP-hard $(\#)H$-COLORING problems, necessitating tools and techniques from *parameterized* and *fine-grained* complexity.

Here, there is no lack of results for *specific* templates H. One of the most well-known results is likely Björklund et al.'s [4] dynamic programming algorithm which solves q-COLORING, as well as the counting, summation, and optimization versions, in $O^*(2^n)$[1] time. Another result for $\#H$-COLORING from Díaz et al. [21] is the existence of an improved algorithm assuming that the input graph G has tree-width k and is given with a nice tree decomposition: for any graph H, $\#H$-COLORING can then be solved on G in $O(|V_H|^{k+1} \min(k, n) \cdot n)$ time using only $O(|V_H|^{k+1} \log(n))$ additional space. The latter examples indicate that bounding the class of input instances by parameters such as tree-width and the related clique-width, can be applied more generally and with powerful algorithmic results.

More generally, two noteworthy extensions of H-COLORING can be defined by restricting either the input or the target graphs with respect to a given parameter k. In the former, the goal is to obtain FPT algorithms, while one in the latter simply wishes to improve upon exhaustive search ($O^*(|D|^n)$), with a single-exponential running time $O^*(c^n)$ for a fixed c depending only on k representing the most desired outcome. Hence, restricting the class of input graphs yields a problem of interest in parameterized complexity while restricting the target graphs yields a problem with closer ties to fast exponential-time algorithms and fine-grained complexity. For instance, the algorithm by Fomin et al. [26] solves H-COLORING in $O^*((t+3)^n)$ time if H has tree-width at most t on every graph G with $|V_G| = n$, assuming that the tree-decomposition is given. Moreover, Wahlström [32] designed an algorithm based on k-expressions and clique-width in order to solve the $\#H$-COLORING problem in time $O^*((2k+1)^n)$, assuming that the clique-width of the graph H is $k \geq 1$. The running time descends to $O^*((k+2)^n)$ if H admits a linear k-expression, leading to a $O^*(6^n)$ complexity for $\#C_p$-COLORING (where C_p is the p-cycle for $p \geq 5$, C_p having a linear 4-expression), and an $O^*(5^n)$ complexity for $\#H$-COLORING, for any cograph H (the cographs with at least one edge being exactly the graphs of clique-width 2). Recently, Okrasa and Rzążewski gave in [30] an algorithm running in time $O^*(|V_H|^{tw(G)})$ solving H-COLORING (assuming an optimal tree-decomposition of G is given) and established its optimality, in the sense that H-COLORING can not be solved in time $O^*((|V_H| - \varepsilon)^{tw(G)})$ for all $\varepsilon > 0$ (under the hypothesis that H is a projective core), unless the *strong exponential-time hypothesis* (SETH) fails[2]. They also achieved similar results involving clique-width [27], where they showed the existence and the optimality of their algorithm running in $O^*(s(H)^{cw(G)})$ (with $s(H)$ being a new structural parameter of H), in the sense that the running time $O^*((s(H) - \varepsilon)^{cw(G)})$ can not be reached under the SETH.

In this paper we pursue this line of research and study the fine-grained and parameterized complexities of $(\#)H$-COLORING problems and, more generally, of #BINARY-CSP(Γ) with a particular focus on the parameter *twin-width*, recently introduced and now widely investigated [2, 6, 7, 9, 8, 10, 11, 12, 13, 14, 15, 16]. This parameter, as well as the underlying *contraction sequences*, give information on whether two vertices are "similar" (in the sense that they have almost the same neighborhoods) in order to treat them simultaneously and thus reduce the computation time. A major achievement of twin-width is that deciding whether a graph G is a model of a closed first-order formula φ is FPT when parameterized by the twin-width of G and the length of φ [14]. Also, the k-INDEPENDENT SET problem is FPT when parameterized by k and twin-width [9]. In [12] Bonnet et al. also gave a proof that the q-COLORING problem was FPT when parameterized by *component twin-width*. Specifically, the parameter *component twin-width* appears to be the most relevant and natural parameter in the setting of homomorphism problems. In fact, many of the algorithms in [9] that are FPT when parameterized by twin-width, such as the one solving k-IND-SET, can be seen as an optimization of a more natural FPT algorithm parameterized by component twin-width.

Despite their impressive success, parameters based on twin-width have not been used to tackle $\#H$-COLORING problems, except for the very restricted case of q-COLORING briefly discussed in [12]. To extend the applicability of such parameters to larger classes of graphs and problems, in Section 3 we propose

[1]The O^* notation means that we ignore polynomial factors.

[2]I.e., that SAT can not be solved in time $(2-\varepsilon)^n \times (n+m)^{O(1)}$ on instances with n variables and m clauses.

a representation of instances and templates of BINARY-CSPs as "edge-labelled graphs". However, classical representations via Gaifman graphs are not adapted to algorithms dealing with component twin-width because the latter does not necessarily increase with higher number of edges.

We thus propose the algebraic notion of *pre-morphism into a semiring* in Section 4, that enables a unified formalism for the list, the counting and the cost generalisations of BINARY-CSP. This formalism also subsumes the semiring based generalisations of CSP (SCSP) by Bistarelly [3], including in particular the counting version of CSP. Even though Wilson [33] proposed an equivalent generalisation through semirings, Wilson's formalism can be seen as a descendant approach, studying subsets of the set of solutions with less and less specifications over time. In contrast, our method builds the set of solutions from trivial cases and two basic operations $\uplus$ and $\bowtie$, that we will perform (through our pre-morphism) in the semiring instead. This approach is often a more common and prolific way to involve graph parameters; see, e.g., Wahlström [32] in the context of clique-width.

Finally, in Section 5 we use contraction sequences along with component twin-width to implement dynamic programming algorithms that efficiently solve these generalisations of BINARY-CSPs. Here, we consider both the case where we bound the input graphs and the case where we bound the template by component twin-width, and we solve both of these questions by two novel algorithms: an FPT algorithm applicable for inputs of bounded component twin-width, and a superpolynomial but significantly improved algorithm applicable to templates of bounded component twin-width. They strongly generalize and improve the results by, e.g., Wahlström [32], and are to the best of our knowledge the most general algorithms of their kind (for binary constraints). In fact, our two algorithms even solve combinations of generalisations of BINARY-CSP without impacting the running time. In Table 1 we summarize a few cases where our approach improves the upper bounds that we can derive from [32], which uses k-expressions and clique-width, and which complements the q-COLORING problems untreated by the inclusion-exclusion method of Björklund et al. [4]. Regarding the latter, we also observe that while it runs in $O^*(2^n)$ time and is able to solve many combinations of generalisations of q-COLORING, it does not cover all combinations of generalisations of BINARY-CSP solved by our algorithms, and is naturally restricted to the very specific case of complete graphs. For our FPT algorithms, component twin-width and clique-width are functionally equivalent [12] on graphs, which allows us to derive FPT conditions for generalized H-COLORING problems with respect to clique-width. This extends the corollary derived from Courcelle et al. [19] that for every graph H, H-COLORING is FPT when parameterized by clique-width even if we add costs, since we now allow combinations of counting generalisations. We summarize the FPT results in Table 4. These results raise several questions for future research, and we discuss some of them in Section 6.

2. Preliminaries

In this section we recall the basic notation and terminology that will be used throughout the paper.

2.1. **Basic Notation.** A *graph* H is a tuple (V_H, E_H) where V_H is a finite set referred as the *set of vertices of* H and E_H is a binary relation over V_H, called the *set of edges of* H[3]. The graph H is said to be *loopless* if E_H is irreflexive and *non-oriented* if E_H is symmetric.

Given an arbitrary set A, we denote by $\mathcal{P}(A)$ the set of all subsets of A. For $S \in \mathcal{P}(\mathcal{P}(A))$ (i.e. $S \subseteq \mathcal{P}(A)$), let $\cup S$ denote the set $\bigcup_{s \in S} s$. Note that $(\cup S \in \mathcal{P}(A))$ (i.e. $S \subseteq A$). We say that S is a *proper partition of* A, if every $a \in A$ belong to a unique $s \in S$, and that $\emptyset \notin S$. For two proper partitions S' and S of A, we say that S' *respects* S if for all $s' \in S'$, there exists $s \in S$ such that $s' \subseteq s$. We occasionally relax the notation and allow a proper partition of A to be family of pairwise disjointed non-empty subsets of A whose union equals A. This will be useful in contexts where the order of the partition matters. If B is an arbitrary set, we let B^A be the set of functions from A to B. For $f \in B^A$ and $A' \subseteq A$, the restriction of f to A' is the function $f|_{A'} : A' \mapsto B$. Also, for $B' \subseteq B$ and $f \in B^A$ with $f(A) \subseteq B'$, the corestriction of f to B' is the function $f|^{B'} : A \mapsto B'$

Throughout, p will only be used to denote an integer ≥ 1, and $[p]$ is the set $\{1, \ldots, p\}$. We let $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}_+$ denote the natural, real, and positive real numbers, respectively. We let $\overline{\mathbb{N}} = \mathbb{N} \cup \{+\infty\}$, $\overline{\mathbb{R}} := \mathbb{R} \cup \{+\infty\}$ and $\overline{\mathbb{R}}_+ := \mathbb{R}_+ \cup \{+\infty\}$. If $A \subseteq \overline{\mathbb{R}}$, we denote its minimum by $\min A$, with the convention that $\min \emptyset = \infty$. Similarly, if $A \subseteq \overline{\mathbb{R}}_+$, we denote its maximum by $\max A$, with the convention that $\max \emptyset = 0$. Also, for

[3]Any graph H will always be denoted $H = (V_H, E_H)$.

Generalisation\Graph	Clique	Cograph	Even cycle	Odd cycle
H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
list-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#list-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
cost-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#cost-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#list-cost-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
weighted-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#weighted-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#list-weighted-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
restricted-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#restricted-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#list-restricted-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#list-restricted-cost-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$
#list-restricted-weighted-H-COLORING	$O^*(3^n)$	$O^*(3^n)$	$O^*(5^n)$	$O^*(5^n)$

TABLE 1. Upper bounds derived from Theorem 16 and Algorithm 1. The $O^*(3^n)$ bounds in blue can be improved to $O^*(2^n)$ by the inclusion-exclusion method [4]. Even-cycle coloring is in P, (our algorithm solves it in $O^*(5^n)$, in green on this table). Also, C_{2k+1}-COLORING (with C_{2k+1} the $(2k+1)$ odd-cycle) can be done in $O^*((\alpha_k)^n)$, with $(\alpha_k)_{k\geq 1}$ decreasing and tending to 1, and with $\alpha_1 \leq \sqrt{2}$ [26] (our algorithm solves it in $O^*(5^n)$, in violet in this table). We improve the results of Wahlström [32] of #cograph-COLORING from $O^*(5^n)$ to $O^*(3^n)$, and of #cycle-COLORING from $O^*(6^n)$ to $O^*(5^n)$ (in red), while also generalising them. To our knowledge, no fine-grained algorithm have been given in the literature for every other problem in this table.

$n, m \geq 1$, $a = (a_1, \ldots, a_n) \in A^n$, and $a' = (a_{n+1}, \ldots, a_{n+m}) \in A^m$, we denote the concatenation of a and a' by a, a' and (a, a') that is defined as the $(n+m)$-tuple $(a_1, \ldots, a_n, a_{n+1}, \ldots, a_{n+m}) \in A^{n+m}$. For $I = \{i_1, \ldots i_{|I|}\} \subseteq [n]$ with $i_1 < \cdots < i_{|I|}$, we denote by a_I and $(a)_I$ the tuple $(a_{i_1}, \ldots, a_{i_{|I|}}) \in A^{|I|}$.

2.2. **Constraint Satisfaction Problems.** The *constraint satisfaction problem* asks whether it is possible to assign values to variables while satisfying all given constraint of the instance. Additionally, it is common to parameterize the problem by a set of relations Γ and a domain D, and constraints are then only allowed to use relations from Γ.

CSP(Γ):
Input: A set V of variables and a set C of constraints of the form $(R_i, (v_i^1, \ldots, v_i^{\mathrm{ar}(R_i)}))$ where $R_i \in \Gamma$ and $(v_i^1, \ldots, v_i^{\mathrm{ar}(R_i)}) \in V^{\mathrm{ar}(R_i)}$ (where $\mathrm{ar}(R_i)$ is the arity of the relation R_i).
Output: 1 if there exists a function $f : V \mapsto D$ such that for all constraints $(R_i, (v_i^1, \ldots, v_i^{\mathrm{ar}(R_i)})) \in C$, we have $(f(v_i^1), \ldots, f(v_i^{\mathrm{ar}(R_i)})) \in R_i$, 0 otherwise.

If Γ only contains binary relations, then we write BINARY-CSP(Γ). For example, when $\Gamma = \{E_H\}$, BINARY-CSP(Γ) is equivalent to the well-known *(digraph) H-COLORING problem*, often referred as H-COLORING when H is symmetric. Regarding the complexity of the latter, Hell and Nešetřil [28] proved that H-COLORING is in P when H is bipartite, and it is NP-complete, otherwise. This result was later generalized to the CSP dichotomy theorem, proven independently by Zhuk [34] and Bulatov [17], which states that every problem of the form CSP(Γ) is either in P or it is NP-complete. In the sequel, we prefer to state our algorithmic results for the most general problem possible (typically BINARY-CSP(Γ) since they imply the results of the specific problems (e.g., H-COLORING). Additionally, we consider the following generalized problems:

- The *counting* version (#): how many functions $f : V \mapsto D$ are solutions of the instances?
- The *list* version: given a matrix $L = \{0,1\}^{V \times D}$ and an instance $\mathcal{I}$, is there a solution $f : V \mapsto D$ satisfying $\forall (u,d) \in V \times D, f(u) = d \implies L(u,d) = 1$?

- The *cost* version [29]: given a matrix $C \in \overline{\mathbb{R}}^{V \times D}$ and an instance $\mathcal{I}$, what is the minimum value of $\sum_{u \in V} C(u, f(u))$ for a solution $f : V \mapsto D$?
- The *weighted* version [25]: given a matrix $W \in \overline{\mathbb{R}}_+^{V \times D}$ and an instance $\mathcal{I}$, what is the minimum value of $\sum_{d \in D} \max_{u \in f^{-1}(\{d\})} W(u, f(u))$ for a solution $f : V \mapsto D$?
- The *restrictive* version [22]: given a tuple $R = \overline{\mathbb{N}}^D$ and and instance $\mathcal{I}$, does there exist a solution $f : V \mapsto D$ where $\forall d \in D, R(d) \neq +\infty \implies |f^{-1}(\{d\})| = R(d)$?

These generalisations can naturally also be combined together. For example, the counting-cost version is the task of counting solutions of minimal cost. We will see in Section 4 that the semiring formulation of CSP makes it possible to subsume all of these generalizations into a single formalism.

2.3. **Parameterized Complexity.** For a computational problem $\mathcal{Q}$ we let $dom(\mathcal{Q})$ be the set of all possible instances of $\mathcal{Q}$. A function $\mu : dom(\mathcal{Q}) \mapsto \mathbb{N}$ is called a *parameter* of $\mathcal{Q}$. We say that $\mathcal{Q}$ is *fixed-parameter tractable w.r.t.* μ if there exists an algorithm solving $\mathcal{Q}$ on every instance $x \in dom(\mathcal{Q})$ of size $\|x\|$ in time $O(f(\mu(x)) \times \|x\|^{O(1)})$, where f can be any computable function. For additional details, we refer to the textbook by Downey and Fellows [23]. For example, *treewidth* of graphs is a well-known parameter frequently resulting in fixed parameter tractability, including the $(\#)H$-COLORING problem [21]. Unfortunately, computing treewidth is in general NP-hard, and the classes of graphs of bounded tree-width are rare. Hence, finding additional examples of (graph width) parameters can in practice be very useful. Another frequently occurring parameter is *clique-width*. This parameter can be seen as a generalisation of tree-width, in the sense that any treewidth-bounded class of graphs is also clique-width bounded, even though the reverse is not true. The use of clique-width has led to many positive results in graph algorithms, e.g., Bodlaender et al. [5] designed an algorithm solving Min-DOMINATING-SET in time $O^*(3^{\frac{\omega}{2}k})$ on graphs of clique width $\leq k$ ($\omega < 2.376$). Kobler et al. [29] also solved a cost version of list-q-COLORING in time $O(2^{2qk} qk^3 n)$ on graphs G with n vertices and clique-width k, and more generally Wahlström proved [32] that $(\#)H$-COLORING is FPT when parameterized by the clique-width of the input graph. Recently, Bonnet et. al [14] introduced *contraction sequences* in order to use dynamic programming on graphs. Contraction sequences allows one to derive a lot of additional parameters such as *twin-width*, oriented twin-width, total twin-width and component twin-width [12]. We do not formally define these parameters here since we in Section 3 define these in a slightly more general context, but remark that component twin-width and clique-width (and thus rank-width) have been proven to be functionally equivalent, allowing one to translate FPT results between these parameters.

2.4. **Fine-Grained Complexity.** A related approach to tackle (NP-)hard problem is the construction of exponential time algorithms running in $O^*(c^n)$ time for as small c as possible, where n is a complexity parameter. In this paper, n will always refer to the number of variables (or vertices) of a CSP (or H-COLORING) instance. This approach, especially in the context of proving matching *lower bounds* under the *(strong) exponential-time hypothesis* ((S)ETH), is often called *fine-grained complexity*.

Notable, general results for H-COLORING problems include the $O^*((2k+1)n)$ algorithm by Fomin et al. [26] for H-COLORING for graphs H of treewidth k, and Wahlström's [32] algorithm based on k-expressions and clique-width which solves $\#H$-COLORING problem in time $O^*((2k+1)^n)$ when H has clique-width $k \geq 1$. The running time descends to $O^*((k+2)^n)$ if H admits a linear k-expression, leading to a $O^*(6^n)$ complexity for $\#C_q$-COLORING (where C_q is the q-cycle for $q \geq 5$, C_q having a linear 4-expression), and a $O^*(5^n)$ complexity for $\#H$-COLORING, for any cograph H. As we will see later, these bounds can in many cases be significantly improved by considering component twin-width instead of clique-width.

3. Edge-Labelled Graphs and $\mathcal{R}$-morphisms

In this section we consider a generalization of graphs, *edge-labelled graphs*, where the corresponding morphism notion greatly increases the expressive power and leads to a rich computational problem. The main idea is that the morphism notion encodes the "rules" of the problems: for example, the "rule" respected by homomorphisms are that edges are sent to edges, whereas in the subgraph-isomorphism problem, the "rules" are that edges are sent to edges, non-edges to non-edges, and different vertices are sent to different vertices. We begin in Section 3.1 by extending the concept of graphs by allowing labels on every pair of vertices, and explicit the notion of morphism relatively to a binary relation over the sets of labels, in which

every pair of vertices must be send to a pair of vertices whose label is prescribed by the relation. Then, in Section 3.2, we prove that the associated morphism problems naturally encode exactly the problems of the form BINARY-CSP(Γ) via a reduction which does not introduce any fresh variables, and where the sets of solutions does not change. Last, in Section 3.3 we show how to formulate the component twin-width parameter in the context of edge-labelled graphs.

3.1. **Definitions.** We first introduce the concept of "edge-labelled graph".

Definition 1. An *edge-labelled graph* G is a structure $G = (V_G, l_G, X_G)$ where V_G and X_G are finite sets and $l_G : (V_G)^2 \mapsto X_G$. The sets V_G and X_G will be referred to respectively the sets of *vertices* and *labels of edges* of G, and l_G will be referred as the *label function* of edges of G.

For any edge-labelled graph G, the set of vertices of G, the set labels of edges of G, and the label function of edges of G, will always be denoted V_G, X_G and l_G (as in Definition 1). For $S \subseteq V_G$, $G[S]$ denotes the edge-labelled graph induced by S on G, i.e., $G[S] := (S, l_G|_{S^2}, X_G)$.

We now introduce a symbol **e** that will play a special role as a label for edge-labelled graph (see Section 3.3 for more details). We implicitly assume that this symbol does not occur in any other context. An edge-labelled graph G is said to be ***e**-free* if $\mathbf{e} \notin X_G$. We now have the necessary technical machinery to properly generalize the concept of a graph homomorphism to edge-labelled graphs.

Definition 2. Let G and H be two **e**-free edge-labelled graphs, and let $\mathcal{R} \subseteq X_G \times X_H$. A function $f\colon V_G \mapsto V_H$ is said to be an $\mathcal{R}$*-morphism* ($f\colon G \underset{\mathcal{R}}{\to} H$) if $\forall (u,v) \in (V_G)^2, (l_G(u,v), l_H(f(u), f(v))) \in \mathcal{R}$.

In this definition, the relation $\mathcal{R}$ encodes whether a pair of vertices (u,v) of G of label $x \in X_G$ is allowed to be sent to a pair of vertices (a,b) of H of a label $y \in X_H$: which it is if and only if $(x,y) \in \mathcal{R}$. By a slight abuse of notation, viewing a graph as an edge-labelled graph whose edges are labelled by 1, and every other pair of vertices is labelled by 0, we notice that a homomorphism is exactly a HOM-morphism if we let HOM be the binary relation over $\{0,1\}$ defined as HOM$= \{(0,0),(0,1),(1,1)\}$.

Similarly to H-COLORING, we will mostly be interested in the version of this problem where the target edge-labelled graph H is fixed. This allows us to model different types of problems simply by changing the template H. Thus, let H be a **e**-free edge-labelled graph, X a finite set, and $\mathcal{R} \subseteq X \times X_H$. We define the following computational problem.

H**-(**$\mathcal{R}$**-MORPHISM):**
Instance: An **e**-free edge-labelled graphs G with $X_G = X$.
Question: Does there exist a function $f\colon G \underset{\mathcal{R}}{\to} H$?

Note that H-COLORING is the same problem as H-(HOM-MORPHISM), i.e., H-($\mathcal{R}$-MORPHISM) is at least as expressive as H-COLORING. We will see Section 3.2 that these problems encode exactly the various BINARY-CSP(Γ) problems, for any set of binary relations Γ.

3.2. **Equivalence Between** H**-(**$\mathcal{R}$**-MORPHISM) and BINARY-CSP**(Γ)**.** We now show that any BINARY-CSP(Γ) problem can be reformulated as a H-($\mathcal{R}$-MORPHISM) problem. For any set Γ of binary relation over a finite domain, we thus need to build an **e**-free edge-labelled graph $H(\Gamma)$, a set X and a relation $\mathcal{R}_\Gamma \subseteq X \times X_{H(\Gamma)}$, in such a way that BINARY-CSP(Γ) and $H(\Gamma)$-($\mathcal{R}_\Gamma$-MORPHISM) are the same problems.

Definition 3. For a finite domain D and a set of binary relations Γ over D we let:

- $H(\Gamma) := (D, l_\Gamma, \mathcal{P}(\Gamma))$ with l_Γ being defined by: $\forall (a,b) \in D, l_\Gamma(a,b) = \{R \in \Gamma \mid (a,b) \in R\}$, and
- $\mathcal{R}_\Gamma$ be defined by, for all $(Y,Z) \in \mathcal{P}(\Gamma)^2, (Y,Z) \in \mathcal{R}_\Gamma \iff Y \subseteq Z$.

Next, we show how to build an equivalent instance of $H(\Gamma)$-($\mathcal{R}_\Gamma$-MORPHISM), given an instance of BINARY-CSP(Γ). Intuitively, for all $(u,v) \in V^2$, the label $l_{G(\mathcal{I})}(u,v)$ represent the constraints that must be respected for the tuple (u,v), while, for $f\colon V \mapsto D$, the label $l_\Gamma((f(u), f(v)))$ represent the constraints of Γ that are actually satisfied by $(f(u), f(v))$. In the following theorem, for an instance $\mathcal{I} = (V, C)$ of BINARY-CSP(Γ), we let $G(\mathcal{I}) := (V, l_\mathcal{I}, \mathcal{P}(\Gamma))$ with $l_\mathcal{I}$ being defined by: $\forall (u,v) \in V, l_\mathcal{I}(u,v) := \{R \in \Gamma \mid R(u,v) \in C\}$.

BINARY-CSP(Γ)	**H-($\mathcal{R}$-MORPHISM)**
V (set of variables)	V_G (vertices of G)
D (domain)	V_H (vertices of H)
Γ	$\mathcal{R}$, X and H
Constraint(s) required for $(u,v) \in V^2$	$l_G(u,v) \in X$
Constraint(s) respected by $(a,b) \in D^2$	$l_H(a,b) \in X_H$
$f : V \mapsto D$ respects every contraint	$f : V_G \mapsto V_H$ is an $\mathcal{R}$-morphism
Graph homomorphism	HOM-morphism with HOM $:= \{(0,0),(0,1),(1,1)\}$
H-COLORING	H-(HOM-MORPHISM)

TABLE 2. Translations between BINARY-CSP and the edge-labelled graph formalism. We view graphs as edge-labelled graphs where the edges are labelled by 1, and every other pair of vertices is labelled by 0.

Theorem 4. *Let Γ be a set of binary relations over a finite domain D, and let $\mathcal{I}$ be an instance of BINARY-CSP(Γ) over a set of variables V. Then, for every $f\colon V \mapsto D$, f is a solution to $\mathcal{I}$ if and only if f is a solution to the instance $G(\mathcal{I})$ of $H(\Gamma)$-($\mathcal{R}_\Gamma$-MORPHISM).*

Conversely, we show that any H-($\mathcal{R}$-MORPHISM) problem can be reformulated as a BINARY-CSP via the following translation.

Definition 5. Let $H = (V_H, l_H, X_H)$ be an **e**-free edge-labelled graph, X a finite set and $\mathcal{R} \subseteq X \times X_H$. For all $x \in X$, define the binary relation over V_H: $R_x := \{(a,b) \in (V_H)^2 \mid (x, l_H(a,b)) \in \mathcal{R}\}$, and let $\Gamma_{H,\mathcal{R}} := \{R_x, x \in X\}$.

In the following theorem, for two edge-labelled graphs G and H and $\mathcal{R} \subseteq X_G \times X_H$, we let $\mathcal{I}(G)$ be the instance of BINARY-CSP($\Gamma_{H,\mathcal{R}}$) with variables V_G and constraints $\{R_{l_G(u,v)}(u,v) \mid (u,v) \in (V_G)^2)\}$.

Theorem 6. *Let $H = (V_H, l_H, X_H)$ be an **e**-free edge-labelled graph, X a finite set and $\mathcal{R} \subseteq X \times X_H$. Let G be an instance of H-($\mathcal{R}$-morphism), i.e. an **e**-free edge-labelled graph with $X_G = X$. Then, for any $f\colon V_G \mapsto V_H$, f is a solution to the instance $\mathcal{I}$ of BINARY-CSP(Γ) if and only if f is a solution to the instance G of H-($\mathcal{R}$-MORPHISM).*

We summarize the translation between the vocabulary of BINARY-CSP and the equivalent concepts of H-($\mathcal{R}$-MORPHISM) in Table 3.2. We again remind the reader that this formulation of BINARY-CSP in terms of edge-labelled graphs makes it easier to employ graph parameters such as the aforementioned component twin-width parameter [12], which we will explicitly show in the forthcoming section.

3.3. **Contractions of Edge-Labelled Graphs and Component Twin-Width.** We now generalize the notion of graph contraction defined by Bonnet et al. [14] to edge-labelled graphs, and define the key notions of contraction sequences and component twin-width. We first define a suitable notion of vertex merging in an edge-labelled graph.

Definition 7. Let $H = (V_H, l_H, X_H)$ be an edge-labelled graph, and let $\mathbb{S}$ be a proper partition of V_H. The *contraction of H relative to* $\mathbb{S}$ is the edge-labelled graph $H_{\mathbb{S}} = (V_{H_{\mathbb{S}}}, l_{H_{\mathbb{S}}}, X_{H_{\mathbb{S}}})$ with $V_{H_{\mathbb{S}}} = \mathbb{S}$, $X_{H_{\mathbb{S}}} = X_H \cup \{\mathbf{e}\}$ and for all $(S_1, S_2) \in \mathbb{S}^2$, if there exists $x \in X_H$ such that $\forall (u,v) \in S_1 \times S_2$, $l_G(u,v) = x$, then $l_{H_{\mathbb{S}}}(S_1, S_2) = x$, and $l_H(S_1, S_2) = \mathbf{e}$ otherwise.

If we view a graph as an edge-labelled graph whose edges are labelled by 1, and every other pair of vertices is labelled by 0, this notion coincides with the notion of (iterated) contraction(s) defined by Bonnet et al. [14] for graphs, up to identifying the set of red edges (defined in [14]) with the set of pairs of $\mathbb{S}$ labelled with **e**. With a slight abuse of notation, we see a proper partition $\mathbb{S}_1$ of a proper partition $\mathbb{S}_2$ of a set V as a proper partition of V, by seeing any set $S \in \mathcal{P}(\mathcal{P}(V))$ of $\mathbb{S}_1$ as $\cup S \in \mathcal{P}(V)$. Under this viewpoint, the proper partition $\mathbb{S}_1$ of V respects the proper partition $\mathbb{S}_2$ of V, and the contraction relation is then transitive. This naturally leads to the following definition of a *contraction sequence*.

Definition 8. Let H be an **e**-free edge-labelled graph on $n \geq 1$ vertices. Then, a *contraction sequence* of H is a sequence of edge-labelled graph $(H_n, \ldots, H_1)$ such that $H_n = H$, and for all $k \in [n-1]$, H_k is a contraction of H_{k+1} with $|V_{H_k}| = k$.

In particular, H_1 is an edge-labelled graph with 1 vertex, and $V_{H_1} = \{V_H\}$. Again, one may notice that the notion of contraction sequence of edge-labelled graph coincides with the particular case of binary loopless non-oriented graphs defined in [14].

Definition 9. Let H be an edge-labelled graph. Define the ***e**-connected components* of H as the connected components of the unoriented graph $H(\mathbf{e}) = (V_H, \{(u,v) \in (V_H)^2 \mid \mathbf{e} \in \{l_H(u,v), l_H(v,u)\}\})$.

The edges of $H(\mathbf{e})$ encode a loss of information in the contraction, and are intended to play the role of the red edges of contraction sequences defined in [14]. We can now introduce the notion of component (twin-)width of a contraction sequence

Definition 10. Let H be an $\mathbf{e}$-free edge-labelled graph on $n \geq 1$ vertices. Let $(H_n, \dots, H_1)$ be a contraction sequence of H. The *component-width* of this contraction sequence is denoted by $ctw((H_n, \dots H_1))$ and is the maximal size of the $\mathbf{e}$-connected component of the graphs $H_{n-1}, \dots, H_1$. The *component twin-width* of H is denoted by $ctww(H)$, and it is the minimal component-width of all its contraction sequences. Any contraction sequence of H whose component-width equals the component twin-width of H is called an *optimal contraction sequence* of H.

Note that the graphs of component twin-width 1 are exactly the cographs, and that cycles of length ≥ 5 have component twin-width 3. In order to extend the notion of component twin-width to BINARY-CSP(Γ) compatible with the reduction involved in Theorem 4, we define the *component twin-width* of the template Γ as the component twin-width of $H(\Gamma)$, and the instance $\mathcal{I}$ as the component twin-width of $G(\mathcal{I})$. Note that while the primary focus of [14] was simply the *twin-width* (the minimum over all contraction sequences of the maximal $\mathbf{e}$-degree of the graphs that appear in the contraction sequence), the latter seems a less useful parameter for BINARY-CSP(Γ). Intuitively, one could argue that component twin-width is a more "natural" parameter to consider, but that improved algorithms using twin-width are sometimes feasible. For instance, Bonnet et al. [9] solves k-IND-SET problem in FPT when parameterized by twin-width, by bounding the number of red-connected subgraphs by a function depending only on k and d, where d is the twin-width of the input graph. This trick transforms an algorithm parameterized by component twin-width into an algorithm parameterized by twin-width. Unfortunately, it does not seem applicable here. Note also that to generalize algorithm of Bonnet et al. [9] to IND-SET, we have to give up the parameterization by twin-width and instead consider component twin-width.

4. Semirings and Generalisations of CSP

In this section we define and extend the *semiring* framework of CSP by Bisterelli [3] and Wilson [33] in the context of H-($\mathcal{R}$-MORPHISM) problems. In particular we will see that all the extensions of the basic CSP problem defined in Section 2 (e.g., counting, finding a solution of minimal cost) can be expressed within this framework, allowing all extended problems to be expressed within a single algebraic framework.

4.1. Semirings and Pre-Morphisms.

In this section, we will define a new algebraic notion in order to encompass the many generalisations of CSP evoked so far.

Definition 11. A *semiring* is a structure $(A, +, \times, 0_A, 1_A)$ such that $(A, +, 0_A)$ is a commutative monoid, $(A, \times, 1_A)$ is a monoid, $\times$ is distributive over $+$, and 0_A is absorbing for $\times$. Moreover, if A is ordered by the binary relation $\leq_A$ over A defined by $\forall (a,b) \in A^2, a \leq_A b \iff \exists c \in A, a + c = b$, then $(A, +, \times, 0_A, 1_A)$ is said to be a *dioid*.

Note that rings and dioids are both particular cases of semirings.

Definition 12. Let S_1 and S_2 be two disjointed sets, and T_1 and T_2 be two sets. Let $f_1 \in (T_1)^{S_1}$ and $f_2 \in (T_2)^{S_2}$. We define the *join* of f_1 and f_2 as $(f_1 \bowtie f_2) \in (T_1 \cup T_2)^{S_1 \uplus S_2}$ defined by $(f_1 \bowtie f_2)|_{S_1} = f_1$ and $(f_1 \bowtie f_2)|_{S_2} = f_2$. Also, for $\mathcal{F}_1 \in \mathcal{P}((T_1)^{S_1})$ and $\mathcal{F}_2 \in \mathcal{P}((T_2)^{S_2})$, we define the *join* of $\mathcal{F}_1$ and $\mathcal{F}_2$ by $\mathcal{F}_1 \bowtie \mathcal{F}_2 = \{(f_1 \bowtie f_2), (f_1, f_2) \in \mathcal{F}_1 \times \mathcal{F}_2\} \in \mathcal{P}((T_1 \cup T_2)^{S_1 \uplus S_2})$.

The join operation will be used by our algorithm, guided by the contraction sequences, to iteratively extend the domain and codomain of the sets of functions considered, with the set of solutions being the final achievement of the algorithm. Similarly, we need disjointed union in order to extend the sets of functions considered. The basic idea behind the semiring framework is then to consider a set A representing the sets

of possible output of a function Ω applied to the set of solutions (of a given CSP instance). In the following definition, we will also take into account a weight matrix W, the weights being elements of a set B.

Definition 13. Let $(A, +, \times, 0_A, 1_A)$ be a semiring, and B a set. Let Ω be a function that maps, for G and H any two edge-labelled graphs, any couple $(\mathcal{F}, W)$ with $\mathcal{F}$ an element of $\mathcal{P}(T^S)$ (with $S \subseteq V_G$ and $T \subseteq V_H$), and $W \in B^{V_G \times V_H}$ a weight matrix, to an element of A, denoted by $\Omega_W(\mathcal{F})$.[4] We say that Ω is a *A-pre-morphism with weights in B*[5] if:

- For all edge-labelled graphs G and H, $W \in B^{V_G \times V_H}$, $S \subseteq V_G$ and $T \subseteq V_H$:
  $\forall (\mathcal{F}_1, \mathcal{F}_2) \in (\mathcal{P}(T^S))^2, \mathcal{F}_1 \cap \mathcal{F}_2 = \emptyset \implies \Omega_W(\mathcal{F}_1 \uplus \mathcal{F}_2) = \Omega_W(\mathcal{F}_1) + \Omega_W(\mathcal{F}_2)$.
- For all edge-labelled graphs G and H, and $W \in B^{V_G \times V_H}$, for all S_1, S_2 being two disjointed subsets of V_G and T_1, T_2 being two disjointed subsets of V_H:
  $\forall (\mathcal{F}_1, \mathcal{F}_2) \in \mathcal{P}((T_1)^{S_1}) \times \mathcal{P}((T_2)^{S_2})$: $\Omega_W(\mathcal{F}_1 \bowtie \mathcal{F}_2) = \Omega_W(\mathcal{F}_1) \times \Omega_W(\mathcal{F}_2)$.
- For all edge-labelled graphs G and H and $W \in B^{V_G \times V_H}$, $\Omega_W(\emptyset) = 0_A$.

Additionally, if we can remove the assumption that T_1 and T_2 are disjointed in the second axiom then we say that the A-pre-morphism Ω is *strong*. If the functions $+$ and $\times$ can be computed in constant time, and if $(G, H, W, S, a) \mapsto \Omega_W(\{f^S_{\{a\}}\})$ (with $f^S_{\{a\}}$ being the constant function of domain S and codomain $\{a\}$) are polynomial time computable, we will say that Ω is a *poly-time computable* A-pre-morphism with weights in B. Let us also remark that $\emptyset$ is neutral for $\uplus$, meaning that the third axiom stating that $\forall W, \Omega_W(\emptyset) = 0_A$ can often be seen as a consequence of the first axiom in many practical cases. The presence of this axiom is only necessary to avoid pathological cases built especially to contradict this axiom that never occur in practice. Similarly, denoting $f^\emptyset_\emptyset$ the unique function with an empty domain and an empty codomain, it is interesting to see that, in the cases that we consider, we will have that, for all weight matrix W, $\Omega_W(\{f^\emptyset_\emptyset\}) = 1_A$ as a consequence that $\{f^\emptyset_\emptyset\}$ is neutral for $\bowtie$ and of the second axiom.

In order to generalize the list, the counting, and the various weighted versions of H-($\mathcal{R}$-MORPHISM), consider the following problem, where H is an edge-labelled graph $\mathcal{R} \subseteq X \times X_H$ with X finite.

$\Omega(H$-($\mathcal{R}$-**MORPHISM**)):
Input: An instance G of H-($\mathcal{R}$-**MORPHISM**), and $W \in B^{V_G \times V_H}$.
Output: The value of $\Omega_W(\{f : G \underset{\mathcal{R}}{\to} H\})$.

The CSP formulation (Ω(CSP(Γ))) is defined analogously, i.e., we ask for the output of Ω_W applied to the set of solutions to the input instance. As usual, we write (Ω(BINARY-CSP(Γ))) when Γ is a set of binary relations. The main advantage of the $\Omega(H$-($\mathcal{R}$-MORPHISM)) formulation is that it easily permits our algorithms to compute values of Ω of larger and larger sets of partial solutions of the given H-($\mathcal{R}$-MORPHISM) instance, via the two operations $\uplus$ and $\bowtie$.

Remark 14. The semiring generalisations of BINARY-CSP(Γ), for Γ over a finite D, subsume BINARY-CSP(Γ), #BINARY-CSP(Γ), #list-BINARY-CSP(Γ) #cost-list-BINARY-CSP(Γ), #weighted-list-BINARY-CSP(Γ), and #restricted-list-BINARY-CSP(Γ), among others. See Table 3 for a summary of these problems and Appendix E for the precise definitions of the associated semirings and pre-morphisms. Also, for a graph H, #restrictive-list(BINARY-CSP($\{E_H\}$)) is the same problem as the counting version of the restrictive-list-H-COLORING problem defined by Díaz et. al in [22]. Similarly, we could have implemented a weighted version of restrictive-list H-COLORING as a generalisation of H-COLORING, by taking a weight matrix W with coefficient in $\overline{\mathbb{N}} \times \overline{\mathbb{R}}$, instead of simply $\overline{\mathbb{N}} \times \{0, 1\}$.

We remark that the approach of generalizing the CSPs using semirings has already been studied by Bistarelli et al. [3], introducing the structure of *c-semirings*, and the computational problem *SCSP*. However, Bistarelli et al. focused on generalisations of CSP involving an optimisation process, requiring a relation $a \leq b$ stating that a is "preferable" to b, achieved by defining $a \leq b \iff a + b = b$ and requiring the $+$ operation to be idempotent. Wilson [33] also defined an equivalent framework that generalizes CSP with semirings. Our contribution with respect to these alternative frameworks is the introduction of the notion of a pre-morphism and its clear link to the operations $\uplus$ and $\bowtie$, which are heavily used in the algorithms in Section 5.

[4] In fact, the value of $\Omega_W(\mathcal{F})$ also depends on the graphs G and H considered, but to ease the notation they are omitted.

[5] If the value of $\Omega_W(\mathcal{F})$ does not depend on W, the precision the set B is irrelevant, and we say that Ω is a *A-pre-morphism ignoring weights*.

Function Ω	A	B	$\Omega(H$-COLORING)	Target for Theorem 15
$\mathbf{1}_{\neq\emptyset}$	$\mathbf{2}$	Unused	H-COLORING	YES
list	$\mathbf{2}$	$\mathbf{2}$	list-H-COLORING	YES
#	$\mathbb{N}$	Unused	#H-COLORING	YES
$\#_{\text{list}}$	$\mathbb{N}$	$\mathbf{2}$	#list-H-COLORING	YES
MinCost, #ArgMinCost	$\overline{\mathbb{R}} \times \mathbb{N}$	$\overline{\mathbb{R}}$	#cost-list-H-COLORING [29]	YES
MinWeight, #ArgMinWeight	$\overline{\mathbb{R}} \times \mathbb{N}$	$\overline{\mathbb{R}}$	#weighted-list-H-COLORING [25]	NO
#restricted-list	$\mathbb{N}$	$\overline{\mathbb{N}} \times \mathbf{2}$	#restricted-list-H-COLORING [22]	NO

TABLE 3. Examples of H-COLORING problems through semirings (with the notation $\mathbf{2} = \{0, 1\}$).

5. Complexity of $\Omega(H$-($\mathcal{R}$-MORPHISM)) via Component Twin-Width

In this section we analyze the complexity of $\Omega(H$-($\mathcal{R}$-MORPHISM)), and thus of Ω(BINARY-CSP(Γ)), with respect to component twin-width. In Subsection 5.1 we consider input graphs with bounded component twin-width, and in Subsection 5.2 we consider target templates H with bounded component-twin width. In Subsection 5.3 illustrate our tractability results on examples from the literature.

5.1. **Parameterized Complexity and Fixed-Parameter Tractability.** We begin by proposing a dynamic programming algorithm applicable to BINARY-CSP and its generalizations in the semiring framework. To simplify the statement of Theorem 15, say that a pre-morphism Ω is *corestriction independent* if for all V_1 and V_2, $W \in B^{V_1 \times V_2}$, and for all subsets $S \subseteq V_1$ and $T \subseteq V_2$, $f \in T^S$, and $T' \subseteq T$ with $f(S) \subseteq T'$, the corestriction $f|^{T'} \in (T')^S$ satisfies $\Omega_W(\{f\}) = \Omega_W(\{f|^{T'}\})$. This rather weak assumption is satisfied by every strong pre-morphism we considered: essentially, it only requires that the singleton values of a function f, do not depend on the vertices of the target graph that are not in the image of f.

Theorem 15. *Let $(A, +, \times, 0_A, 1_A)$ be a semiring, Ω a poly-time computable strong A-pre-morphism with weights in a set B, H an* e*-free edge-labelled graph, X a finite set, and $\mathcal{R} \subseteq X \times X_H$. Assume that Ω is corestriction independent. Then, for every instance G on $n \geq 1$ vertices, Algorithm 2 with fixed, H, $\mathcal{R}$ and Ω solves Ω(H-($\mathcal{R}$-MORPHISM)) in $O(((2^{|V_H|} - 1)^{ctww(G)}) \times |V_G|^2)$ time, provided that an optimal contraction sequence $\mathbb{G}$ of G is given.*

Algorithm 2 can be found in Appendix D, and uses dynamic programming along with optimal contraction sequences to achieve the desired solution(s). We are able to deal with semiring generalisations since, for an e-connected component C in the contraction sequence, instead of just keeping in memory whether a function $\gamma : C \mapsto \mathcal{P}(V_H) \setminus \emptyset$ is a profile (as it is done in [12]), we store the value by Ω of the set of partial solutions that induce this profile (i.e., the set of $\mathcal{R}$-morphisms $f : C \mapsto V_H$ such that for all $S \in C, f(S) = \gamma(S)$). We see that the algorithm by Bonnet et al. [12] solving the q-COLORING problem in FPT time parameterized by component twin-width is the particular case where the pre-morphism considered is over the semirings of Booleans, and maps $\emptyset$ to 0 and any other set to 1 (see Lemma 31). Also, we are able to deal with arbitrary edge-labelled graphs H instead of only the q-clique K_q by replacing the update of sets of profiles by Bonnet et al. [12] that checks the absence of a black edge, by the more general test of *feasibility* (see Appendix B). Thus, Algorithm 2 is much more general than the aforementioned algorithm by Bonnet et al. since it applies to arbitrary binary constraints rather than the specific template K_q, and is applicable to generalized problems described by the pre-morphism Ω (counting, with weights, and so on).

It may also be interesting to remark note that the complexity of Algorithm 2 depends neither on the semiring A, nor on the pre-morphism Ω, i.e., the generalized problems do not impact the running time.

5.2. **Upper Bounds on Fine-Grained Complexity.** In Section 5.1 we exploited the contraction sequence of an instance $\mathcal{I}$ to obtain an FPT algorithm with respect to the component twin-width of $G(\mathcal{I})$. We now turn to the dual question of constructing an improved (exponential time) algorithm with respect to the component twin-width of $H(\Gamma)$. Note that the computation of an optimal contraction sequence of $H(\Gamma)$ can be seen as a form of pre-computation since it is independent of the instance. Also, when working with H instead of G, our algorithm will have to guess preimages of subsets T of V_H instead of images of subsets

Generalisation	Target for Courcelle [20]	Target for Theorem 15
H-COLORING	YES	YES
#H-COLORING		YES
list-H-COLORING	YES	YES
#list-H-COLORING		YES
cost-H-COLORING	YES	YES
#cost-H-COLORING		YES
#list-cost-H-COLORING		YES
weighted-H-COLORING	YES	see discussion in Section 6
#weighted-H-COLORING		see discussion in Section 6
#list-weighted-H-COLORING		see discussion in Section 6
restricted-H-COLORING	YES	see discussion in Section 6
#restricted-H-COLORING		see discussion in Section 6
#list-restricted-H-COLORING		see discussion in Section 6
#list-restricted-cost-H-COLORING		see discussion in Section 6
#list-restricted-weighted-H-COLORING		see discussion in Section 6

TABLE 4. Tractability results parameterized by the (functionally equivalent) parameters clique-width/component twin-width.

S of V_G. Since preimages of pairwise disjoint subsets are pairwise disjointed, our algorithm also applies to semiring pre-morphisms that are not strong.

Theorem 16. *Let $(A, +, \times, 0_A, 1_A)$ be a semiring, and Ω a A-pre-morphism with weights in a set B, H an e-free edge-labelled graph, $\mathbb{H}$ an optimal contraction sequence of H, and X a finite set, $\mathcal{R} \subseteq X \times X_H$. Then, Algorithm 1 with fixed $\mathbb{H}$, $\mathcal{R}$ and Ω solves Ω(H-($\mathcal{R}$-MORPHISM)) in time $O((ctww(H) + 2)^{|V_G|} \times |V_G|^2)$ every intance G on $n \geq 1$ vertices.*

Algorithm 1 has similarities with the algorithm given by Wahlström [32] which solves #H-COLORING in time $O^*((2cw(H) + 1)^n)$ on input graphs G on n vertices (with $cw(H)$ the clique-width of H). However, our algorithm has two significant advantages. First, it is applicable to arbitrary BINARY-CSP problems, and its generalized problems in the semiring framework (without impacting the running time). Second, even though clique-width and component twin-width are functionally equivalent, even a minor increase of the width parameter can significantly change the run time of the algorithm. For example, the problem of counting the number of homomorphisms into a cycle of length k, #C_k-COLORING, can be solved in $O^*(5^n)$ time by Algorithm 1 but requires $O^*(6^n)$ time by the clique-width algorithm. Additional examples are provided in Section 5.3.

5.3. **Consequences.** The tractability of many semiring generalisations of H-COLORING and BINARY-CSP(Γ) easily follow from Theorem 15.

Corollary 17. Let Γ be a set of binary relations over a finite domain. Then, BINARY-CSP(Γ), #BINARY-CSP(Γ), #list-BINARY-CSP(Γ), #cost-list-BINARY-CSP(Γ) with weights are FPT parameterised by the component twin-width of the instance.

This strongly generalizes many results in the literature. For example, Kobler and Rotics [29] proved that costs-list-q-COLORING is FPT when parameterized by clique-width (recall that clique-width and component twin-width are functionally equivalent on graphs [12]). Similarly, we strongly generalize Wahlström's FPT algorithm for #H-COLORING (with respect to clique-width) since we can handle arbitrary binary constraints as well as the extended problems. This also supplements to the well known result that, for every graph H, H-COLORING is FPT when parameterized by clique-width, by solving also counting versions. This is a corollary derived from Courcelle et al. [19] algorithm that solves an optimization version of the problem of checking whether a valuation over the vertices of structure is a model of a fixed monadic second-order logic formula in FPT time (parameterized by clique-width), see Table 4.

We can also use Theorem 16 to derive upper bounds on the complexities of several generalisations of H-COLORING problems (for some specific values of H) through semirings that improve previously know

results. These results are summarized in Table 1 and are straightforward consequences of Theorem 15 and Theorem 16 (but explicitly demonstrated in in Appendix E).

6. Conclusions and Perspectives

We investigated the complexity of binary constraint satisfaction problems under the lens of the component twin-width parameter. In order to obtain as general results as possible, we considered several frequently occurring problem extensions, e.g., counting, allowing weights, cost and list constraints, which we formulated in a unifying semiring framework which greatly simplified the algorithmic results. Importantly, we obtained two novel algorithms by bounding either the class of input instances, or the constraint template, and presented several instances where our approach beats both the best known upper bound (e.g., counting homomorphisms to cycles) as well as improving upon earlier algorithms making use of tree-width and clique-width. These results raise several questions for future work:

Generalized problems. Even though Theorem 15 and Theorem 16 are very general algorithms applicable to broad classes of binary constraints it is still tempting to generalize them to even wider classes of problems. For instance, even though Theorem 15 does not apply to every generalisation of BINARY-CSP presented in Table 4 (the one involving *weighted* and *restricted*), it is still possible to prove that these problems are FPT parametrized by component twin-width. It is sufficient to modify Algorithm 2 by adding to the tabular OMEGA m entries corresponding to the m vertices of H, corresponding to the weights/cardinal (when considering the weighted/restricted generalisations) of preimages reached by every vertex of H, which allows dynamic programming. We expect that implementing an algebraic structure over the set of weights B would make possible to reformulate Theorem 15 and Algorithm 2 in order to include these kind of algorithms and tractability results as well. To go even further, it would be interesting to generalize Theorem 15 and Theorem 16 to constraints of arbitrary arity. One possibility is to express such CSPs by generalising the concept of edge-labelled graphs to "edge-labelled hypergraphs", labelling arbitrary large tuples over V_G. Are the usual parameters on hypergraphs, and in particular generalizations of (component) twin-width, applicable to edge-labelled hypergraphs? Variants of CSP could also be considered. For instance, can we solve the PROMISE-BINARY-CSP problems (see Barto et al. [1] for details) similarly? A promise constraint satisfaction problem (PCSPs) requires two finite similar structures $\mathcal{A}$ and $\mathcal{B}$ with an homomorphism $h : \mathcal{A} \to \mathcal{B}$, and asks, given a structure $\mathcal{I}$ whether $\mathcal{I} \to \mathcal{A}$ or $\mathcal{I} \not\to \mathcal{B}$, with the promise that these two statements are not both false (we know that they are not both true from the existence of h). The algebraic approach proposed by Barto et al. [1] already led to interesting hardness results, such as the NP-hardness of the distinguishment of the q-colourable graphs from those that are not $(2q-1)$-colourable. As a complete dichotomy theorem still eludes us, it would be interesting to check whether techniques from parameterized and fine-grained complexity can be applied in the promise setting.

Comparison with clique-width and other parameters. Other graph parameters can probably be efficiently extended to edge-labelled graphs and thus to BINARY-CSP, while preserving the soundness of the algorithms working on this parameters. We believe that clique-width should be a good candidate since it is functionally equivalent on graphs [12], with the main structural subtlety being the creation of labelled edges in k-expressions. Assuming that we have been able to extend clique-width to edge-labelled graphs, does it stay functionally equivalent to component twin-width on edge-labelled graphs? We also believe that investigating the parameterized complexity of semirings generalisations of (BINARY-)CSP with other parameters than component twin-width could lead to similar interesting results. However, we have been unable to produce any meaningful results with twin-width rather than component twin-width. Can this difficulty be formalized into a concrete lower bound?

Algebraic developments. The algebraic generalisations of BINARY-CSP through semiring pre-morphism are inconvenient in certain aspects. For instance, as discussed earlier, it would be interesting to describe convenient structures over the sets of weights. Most importantly, we still lack algebraic operations that combine semiring pre-morphisms together, in order to automatically handle combinations of semiring generalisations without redefining a new semiring pre-morphism each time. For example, it would be desirable to be able to build the $\mathbb{N}$-pre-morphism "$\#_{\text{list}}$" (which leads to the counting-list generalisation, see Lemma 34) using the $\mathbb{N}$-pre-morphism "#" (which leads to the counting generalisation, see Lemma 33) and the $\mathbf{2}$-pre-morphism "list" (which leads to the list generalisation, see Lemma 32). Moreover, we noticed that every semiring used

in this paper is even a dioid. Can we take advantage of the additional properties of dioids? More generally, can we extend BINARY-CSP via other algebraic structures?

Appendix A. Proofs of Section 3.2

We give here a proof of Theorem 4. It follows naturally from the constructions $H(\Gamma)$, $\mathcal{R}_\Gamma$ and $G(\mathcal{I})$.

Theorem 4. *Let Γ be a set of binary relations over a finite domain D, and let $\mathcal{I}$ be an instance of BINARY-CSP(Γ) over a set of variables V. Then, for every $f\colon V \mapsto D$, f is a solution to $\mathcal{I}$ if and only if f is a solution to the instance $G(\mathcal{I})$ of $H(\Gamma)$-($\mathcal{R}_\Gamma$-MORPHISM).*

Proof. f is a solution of the instance $\mathcal{I}$ of BINARY-CSP(Γ) $\underset{\text{By definition of BINARY-CSP}(\Gamma)}{\iff}$

For all $(u,v) \in V^2$, for all constraint $R(u,v)$ of $\mathcal{I}$, $(f(u), f(v)) \in R$ $\underset{\text{By definition of } l_{\mathcal{I}(u,v)}}{\iff}$

For all $(u,v) \in V^2$, for all $R \in l_{\mathcal{I}}(u,v)$, $(f(u), f(v)) \in R$ $\underset{\text{By definition of } l_\Gamma(f(u),f(v))}{\iff}$

For all $(u,v) \in V^2$, for all $R \in l_{\mathcal{I}}(u,v)$, $R \in l_\Gamma(f(u), f(v))$ $\underset{\text{By definition of inclusion}}{\iff}$

For all $(u,v) \in V^2$, $l_{\mathcal{I}}(u,v) \subseteq l_\Gamma(f(u), f(v))$ $\underset{\text{By definition of } \mathcal{R}_\Gamma}{\iff}$

For all $(u,v) \in V^2$, $(l_{\mathcal{I}}(u,v), l_\Gamma(f(u), f(v))) \in \mathcal{R}_\Gamma$ $\underset{\text{By definition of } H(\Gamma)\text{-}(\mathcal{R}_\Gamma\text{-MORPHISM})}{\iff}$

f is a solution of the instance $G(\mathcal{I})$ of $H(\Gamma)$-($\mathcal{R}_\Gamma$-MORPHISM)

□

Similarly, we give a proof to Theorem 6 which essentially follows from the definitions of $\Gamma_{H,\mathcal{R}}$ and $\mathcal{I}(G)$.

Theorem 6. *Let $H = (V_H, l_H, X_H)$ be an **e**-free edge-labelled graph, X a finite set and $\mathcal{R} \subseteq X \times X_H$. Let G be an instance of H-($\mathcal{R}$-morphism), i.e. an **e**-free edge-labelled graph with $X_G = X$. Then, for any $f\colon V_G \mapsto V_H$, f is a solution to the instance $\mathcal{I}$ of BINARY-CSP(Γ) if and only if f is a solution to the instance G of H-($\mathcal{R}$-MORPHISM).*

Proof. f is a solution of the instance $\mathcal{I}(G)$ of BINARY-CSP($\Gamma_{H,\mathcal{R}}$) $\underset{\text{By definitions of BINARY-CSP}(\Gamma_{H,\mathcal{R}}) \text{ and } \mathcal{I}(G)}{\iff}$

For all $(u,v) \in (V_G)^2$, $(f(u), f(v)) \in R_{l_G(u,v)}$ $\underset{\text{By definition of } R_{l_G(u,v)}}{\iff}$

For all $(u,v) \in (V_G)^2$, $(l_G(u,v), l_H((f(u), f(v)))) \in \mathcal{R}$ $\underset{\text{By definition of } H\text{-}(\mathcal{R}\text{-MORPHISM})}{\iff}$

f is a solution of the instance G of H-($\mathcal{R}$-MORPHISM).

□

Appendix B. Feasibility

Looking at Definition 7, we can interpret the **e**-edges of H' as a loss of information. Since we want to study the $\mathcal{R}$-morphisms of every subgraph of G to some wisely chosen subgraphs of H, it seems natural to choose the subgraphs of H of the form $H[T_1 \uplus \cdots \uplus T_p]$, when $\{T_1, \ldots, T_p\}$ is a **e**-connected component of a graph H_k ($k \in [m]$) of the contraction. This is the main reason why component twin-width affects the complexity of the algorithm.

Definition 18. Let G and H be two **e**-free edge-labelled graphs, $\mathcal{R} \subseteq X_G \times X_H$, H' a contraction of H, $\mathbb{T} = (T_1, \ldots, T_p)$ a tuple of p different vertices of H', and $\mathbb{S} = (S_1, \ldots, S_p)$ a tuple of p pairwise disjointed subsets of V_G (possibly empty). We say that $\mathbb{S}$ is *$\mathcal{R}$-feasible* with respect to $\mathbb{T}$ and we denote $\mathbb{S} \preceq_{\mathcal{R}} \mathbb{T}$ if for all $(i,i') \in [p]^2$, we have:

$$l_{H'}(T_i, T_{i'}) \neq \mathbf{e} \implies \forall (u,v) \in S_i \times S_{i'}, (l_G(u,v), l_{H'}(T_i, T_{i'})) \in \mathcal{R}.$$

We also define a symmetrical notion when a contraction of G is considered instead.

Definition 19. Let G and H be two **e**-free edge-labelled graphs, $\mathcal{R} \subseteq X_G \times X_H$, G' a contraction of G, $\mathbb{S} = (S_1, \ldots, S_p)$ a tuple of p different vertices of G', and $\mathbb{T} = (T_1, \ldots, T_p)$ a tuple of p non-empty subsets of V_H (not necessarily pairwise disjointed). We say that $\mathbb{T}$ *makes* $\mathbb{S}$ *$\mathcal{R}$-feasible*, and we denote $\mathbb{T} \succeq_{\mathcal{R}} \mathbb{S}$ if for all $(i,i') \in [p]^2$, we have:

$$l_{G'}(S_i, S_{i'}) \neq \mathbf{e} \implies \forall (a,b) \in T_i \times T_{i'}, (l_G(S_i, S_{i'}), l_H(a,b)) \in \mathcal{R}.$$

In the Definition 18, we ask for the elements of $\mathbb{S}$ to be pairwise disjointed, whereas in Definition 19, we require the elements of $\mathbb{T}$ to be non-empty. The reason is that when employing Definition 18, the elements of $\mathbb{S}$ will play the role of the preimages of the elements of $\mathbb{T}$, and will therefore be pairwise disjointed (since the elements of $\mathbb{T}$ are non-empty and pairwise disjointed), whereas in Definition 19, the elements of $\mathbb{T}$ will play the role of the images of the elements of $\mathbb{S}$, and will therefore be non-empty (since the elements of $\mathbb{S}$ are non-empty and pairwise disjointed).

Remark 20. Using Definition 7, we notice the truthness of the proposition "$\mathbb{S} \preceq_{\mathcal{R}} \mathbb{T}$" and "$\mathbb{T} \succeq_{\mathcal{R}} \mathbb{S}$" does not depend on the contraction H' or G'.

Appendix C. Soundness of Algorithm 1

Create a tabular OMEGA filled with 0_A
for $S \subseteq V_G$, $a \in V_H$ **do**
  if $\forall (u,v) \in S^2, (l_G(u,v), l_H(a,a)) \in \mathcal{R}$ **then**
    $\text{OMEGA}\begin{bmatrix} S & \{a\} \end{bmatrix} \leftarrow \Omega_W(\{f^S_{\{a\}}\})$ (with $f^S_{\{a\}} : \begin{matrix} S & \mapsto & \{a\} \\ u & \mapsto & a \end{matrix}$)
  end
end
for $k = m-1$ *downto* 1 **do**
  $(T_p, T_{p+1}) \leftarrow$ contracted pair in the contraction $H_{k+1} \to H_k$ of $\mathbb{H}$
  $T_0 \leftarrow$ contraction of T_p and T_{p+1} in H_k
  $C = \{T_0, T_1, \dots T_{p-1}\}$ the e-connected components of H_k containing T_0
  $C_1 \uplus \dots \uplus C_q = \{T_1, \dots, T_{p-1}, T_p, T_{p+1}\}$ be the partitionning of $(C \setminus \{T_0\}) \cup \{T_p, T_{p+1}\}$ into e-connected components in H_{k+1}
  for $j = 1$ *to* q **do**
    Define $I_j \subseteq [p]$ such that $C_j = \{T_i, i \in I_j\}$
    $I_j =: \{I_j[1], \dots, I_j[p_j]\}$ with $p_j = |I_j|$
  end
  for $S_0, S_1, \dots, S_{p-1} \subseteq V_G$ *pairwise disjointed* **do**
    for $S_p \uplus S_{p+1}$ *partitionning* S_0 **do**
      if $(S_1, \dots, S_{p-1}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (T_1, \dots, T_{p-1}, T_p, T_{p+1})$ **then**
        $$\text{OMEGA}\begin{bmatrix} S_0 & T_0 \\ S_1 & T_1 \\ \vdots & \vdots \\ S_{p-1} & T_{p-1} \end{bmatrix} += \text{OMEGA}\begin{bmatrix} S_{I_1[1]} & T_{I_1[1]} \\ \vdots & \vdots \\ S_{I_1[p_1]} & T_{I_1[p_1]} \end{bmatrix} \times \dots \times \text{OMEGA}\begin{bmatrix} S_{I_q[1]} & T_{I_q[1]} \\ \vdots & \vdots \\ S_{I_q[p_q]} & T_{I_q[p_q]} \end{bmatrix}$$
      end
    end
  end
end
return $\text{OMEGA}\begin{bmatrix} V_G & V_H \end{bmatrix}$

Algorithm 1: Solving $\Omega(H\text{-}(\mathcal{R}\text{-MORPHISM}))$ ie. $\Omega(\text{BINARY-CSP}(\Gamma))$ (fine-grained version)

Definition 21. Let G and H two e-free edge-labelled graphs on n and m vertices respectively. Let $\mathcal{R} \subseteq X_G \times X_H$. Let $\mathbb{T} = (T_1, \dots, T_p)$ be a tuple of p pairwise disjointed subsets of V_H (in particular, $T_1, \dots, T_p$ can be different vertices of a contraction H' of H) and $T = T_1 \uplus \dots \uplus T_p$. Let $\mathbb{S} = (S_1, \dots, S_p)$ be a tuple of p pairwise disjointed subsets of V_G, and $S = \cup \mathbb{S} = S_1 \uplus \dots \uplus S_p$. We denote by $\mathcal{R}^{\mathbb{S}}_{\mathbb{T}}$ the set

$$\mathcal{R}^{\mathbb{S}}_{\mathbb{T}} := \{f : G[S] \underset{\mathcal{R}}{\to} H[T] \mid f(S_1) \subseteq T_1, \dots, f(S_p) \subseteq T_p\}$$

Lemma 22. *Let G and H two* **e***-free edge-labelled graphs, on n and m vertices respectively. Let H' be a contraction of H, $\mathcal{R} \subseteq X_G \times X_H$, $C_1, \dots, C_q$ be q different* **e***-connected components of H' ($q \geq 1$), $\{T_1, \dots, T_p\} = C_1 \uplus \dots \uplus C_q$ and let $\mathbb{T} = (T_1, \dots, T_p)$, $\mathbb{S} = (S_1, \dots, S_p)$ be a tuple of p pairwise disjointed subsets of V_G. Let for all $j \in q$, $I_j \subseteq [p]$ be such that $C_j = \{T_i, i \in I_j\}$. Then*

$$\mathcal{R}_{\mathbb{T}}^{\mathbb{S}} = \begin{cases} \emptyset & \text{if } \mathbb{S} \not\preceq_{\mathcal{R}} \mathbb{T} \\ \mathcal{R}_{\mathbb{T}_{I_1}}^{\mathbb{S}_{I_1}} \bowtie \dots \bowtie \mathcal{R}_{\mathbb{T}_{I_q}}^{\mathbb{S}_{I_q}} & \text{if } \mathbb{S} \preceq_{\mathcal{R}} \mathbb{T} \end{cases}$$

Proof. First, assume that $\mathbb{S} \not\preceq_{\mathcal{R}} \mathbb{T}$: Assume by contradiction that there exists $f \in \mathcal{R}_{\mathbb{T}}^{\mathbb{S}}$, i.e. f is an $\mathcal{R}$-morphism from the edge-labeled graph $G[S_1 \uplus \dots \uplus S_p]$ to the edge-labeled graph H and $\forall i \in [p], f(S_i) \subseteq T_i$. Since $\mathbb{S}$ is not $\mathcal{R}$-feasible with respect to $\mathbb{T}$, there exists a pair $(i, i') \in [p]^2$ such that:

- $l_{H'}(T_i, T_{i'}) \neq$ **e**,
- there exists $u \in S_i$ and $v \in S_{i'}$ with $(l_G(u, v), l_{H'}(T_i, T_{i'})) \notin \mathcal{R}$.

By Definition 7, $\forall (a, b) \in T_i \times T_{i'}, l_H(a, b) = l_{H'}(T_i, T_{i'})$. Notice that, by definition of $f \in \mathcal{R}_{\mathbb{T}}^{\mathbb{S}}$, $(f(\underbrace{u}_{\in S_i}), f(\underbrace{v}_{\in S_{i'}})) \in T_i \times T_{i'}$. Thus, $l_H(f(u), f(v)) = l_{H'}(T_i, T_{i'})$. Since f is an $\mathcal{R}$-momorphism, we have $(l_G(u, v), \underbrace{l_H(f(u), f(v))}_{= l_{H'}(T_i, T_{i'})}) \in \mathcal{R}$, which contradicts the definition of (u, v). We have a contradiction, which proves that $\mathcal{R}_{\mathbb{T}}^{\mathbb{S}} = \emptyset$.

Second, assume that $\mathbb{S} \preceq_{\mathcal{R}} \mathbb{T}$:

Take $f \in \mathcal{R}_{\mathbb{T}}^{\mathbb{S}}$ and $j \in [q]$. Since restrictions of $\mathcal{R}$-morphisms are $\mathcal{R}$-morphisms, $f|_{\cup \mathbb{S}_{I_j}}$ is an $\mathcal{R}$-morphism. Moreover, since $f \in \mathcal{R}_{\mathbb{T}}^{\mathbb{S}}$, for all $j \in I_j$, $f(S_j) \subseteq T_j$. We deduce that $f|_{\cup \mathbb{S}_{I_j}} \in \mathcal{R}_{\mathbb{T}_{I_j}}^{\mathbb{S}_{I_j}}$. This proves that $\mathcal{R}_{\mathbb{T}}^{\mathbb{S}} \subseteq \mathcal{R}_{\mathbb{T}_{I_1}}^{\mathbb{S}_{I_1}} \bowtie \dots \bowtie \mathcal{R}_{\mathbb{T}_{I_q}}^{\mathbb{S}_{I_q}}$.

We now prove the reverse. Let $(f_1, \dots, f_q) \in \mathcal{R}_{\mathbb{T}_{I_1}}^{\mathbb{S}_{I_1}} \times \dots \times \mathcal{R}_{\mathbb{T}_{I_q}}^{\mathbb{S}_{I_q}}$, and let $f = f_1 \bowtie \dots \bowtie f_q$. We will prove that $f \in \mathcal{R}_{\mathbb{T}}^{\mathbb{S}}$. Clearly, by definition of the $\mathcal{R}_{\mathbb{T}_{I_j}}^{\mathbb{S}_{I_j}}$ for $j \in [q]$, we have, for all $i \in [p], f(S_i) \subseteq T_i$ (knowing that $(I_1, \dots, I_q)$ is a partition of $[p]$). There only remains to prove that f is an $\mathcal{R}$-morphism. Let $S = S_1 \uplus \dots \uplus S_p$ and take $(u, v) \in S^2$. We will prove that $(l_G(u, v), l_H(f(u), f(v))) \in \mathcal{R}$. Let $(i, i') \in [p]^2$ be such that $u \in S_i$ and $v \in S_{i'}$.

(1) If there exists $j \in [q]$ such that $(i, i') \in (I_j)^2$ (ie. if T_i and $T_{i'}$ belong to the same **e**-connected component C_j), then, $(u, v) \in (S_{I_j})^2$. It follows by definition of f that $(f(u), f(v)) = (f_j(u), f_j(v))$, and then $(l_G(u, v), l_H(f(u), f(v))) \in \mathcal{R}$ because f_j is an $\mathcal{R}$-morphism.

(2) Else, by definitions of I_j and C_j for $j \in [q]$, T_i and $T_{i'}$ are not **e**-connected in H'. We deduce that, in particular, $l_{H'}(T_i, T_{i'}) \neq$ **e**. Using Definition 7: $\forall (a, b) \in T_i \times T_{i'}, l_H(a, b) = l_{H'}(T_i, T_{i'})$. Then, $(f(u), f(v)) = (f_i(u), f_{i'}(v)) \in T_i \times T_{i'}$, thus $l_H(f(u), f(v)) = l_{H'}(T_i, T_{i'})$. Using $\mathbb{S} \preceq_{\mathcal{R}} \mathbb{T}$, we have $(l_G(u, v), \underbrace{l_{H'}(T_i, T_{i'})}_{= l_H(f(u), f(v))}) \in \mathcal{R}$.

We have proven that $(l_G(u, v), l_H(f(u), f(v))) \in \mathcal{R}$. Hence, f is indeed an $\mathcal{R}$-morphism, which concludes the proof.

□

Lemma 23. *Let G and H be two* **e***-free edge-labelled graphs, and let $\mathcal{R} \subseteq X_G \times X_H$, $T_0, T_1, \dots, T_{p-1}$ be p pairwise disjointed subsets of V_H, and let $\mathbb{T} = (T_1, \dots, T_{p-1})$, $S_0, S_1, \dots, S_{p-1}$ be p pairwise disjointed subsets of V_G, and let $\mathbb{S} = (S_1, \dots, S_{p-1})$, $T_p \uplus T_{p+1} = T_0$ be a partition of T_0. Then*

$$\mathcal{R}_{\mathbb{T}, T_0}^{\mathbb{S}, S_0} = \biguplus_{\substack{S_p \uplus S_{p+1} = S_0 \\ (\mathbb{S}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (\mathbb{T}, T_p, T_{p+1})}} \mathcal{R}_{\mathbb{T}, T_p, T_{p+1}}^{\mathbb{S}, S_p, S_{p+1}}$$

Proof. Partitioning the set $\mathcal{R}_{\mathbb{T}, T_0}^{\mathbb{S}, S_0}$ in equivalence classes with respect to the equivalence relation $\sim$ defined by:

$$\forall (f,f') \in (\mathcal{R}^{\mathbb{S},S_0}_{\mathbb{T},T_0})^2, f \sim f' \iff \underbrace{(f^{-1}(T_p), f^{-1}(T_{p+1}))}_{\text{partition of } S_0} = \underbrace{(f'^{-1}(T_p), f'^{-1}(T_{p+1}))}_{\text{partition of } S_0}$$

we obtain:

$$\mathcal{R}^{\mathbb{S},S_0}_{\mathbb{T},T_0} = \biguplus_{S_p \uplus S_{p+1} = S_0} \mathcal{R}^{\mathbb{S},S_p,S_{p+1}}_{\mathbb{T},T_p,T_{p+1}}$$

and then, the conclusion follows from Lemma 22.

□

Theorem 24. *Let Ω a semiring pre-morphism and W a weight matrix for Ω, G and H be two* **e***-free edge-labelled graphs, $(H_m, \dots, H_1)$ a contraction sequence of H (with $m := |V_H|$), and $k \in [m-1]$, $C_1, \dots, C_q$ be q ($q \geq 1$) different* **e***-connected components of H_{k+1} ($q \geq 1$). Let $\{T_1, \dots, T_{p-1}, T_p, T_{p+1}\} = C_1 \uplus \dots \uplus C_q$ and $\mathbb{T} = (T_1, \dots, T_{p-1})$, $S_0, S_1, \dots, S_{p-1}$ be p pairwise disjointed subsets of V_G. Let $\mathbb{S} = (S_1, \dots, S_{p-1})$. Denote, for all $j \in q$, $I_j \subseteq [p+1]$ such that $C_j = \{T_i, i \in I_j\}$. Assume that H_k is obtained from H_{k+1} by contracting the two different vertices T_p and T_{p+1} to the vertex $T_0 = T_p \uplus T_{p+1}$. Then, we have:*

$$\Omega_W(\mathcal{R}^{\mathbb{S},S_0}_{\mathbb{T},T_0}) = \sum_{\substack{S_p \uplus S_{p+1} = S_0 \\ (\mathbb{S}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (\mathbb{T}, T_p, T_{p+1})}} \left(\prod_{j=1}^{q} \Omega_W(\mathcal{R}^{(\mathbb{S},S_p,S_{p+1})_{I_j}}_{(\mathbb{T},T_p,T_{p+1})_{I_j}})\right)$$

(the notations $\sum$ and $\prod$ reffer to the sum and product of the semiring).

Proof. Using Lemma 23:

$$\Omega_W(\mathcal{R}^{\mathbb{S},S_0}_{\mathbb{T},T_0}) = \Omega_W\Big(\biguplus_{\substack{S_p \uplus S_{p+1} = S_0 \\ (\mathbb{S}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (\mathbb{T}, T_p, T_{p+1})}} \mathcal{R}^{\mathbb{S},S_p,S_{p+1}}_{\mathbb{T},T_p,T_{p+1}} \Big)$$

using the first axiom of pre-morphisms:

$$\Omega_W(\mathcal{R}^{\mathbb{S},S_0}_{\mathbb{T},T_0}) = \sum_{\substack{S_p \uplus S_{p+1} = S_0 \\ (\mathbb{S}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (\mathbb{T}, T_p, T_{p+1})}} \Omega_W(\mathcal{R}^{\mathbb{S},S_p,S_{p+1}}_{\mathbb{T},T_p,T_{p+1}})$$

and we can apply iteratively the second axiom of pre-morphisms with $\mathcal{F} := \mathcal{R}^{\mathbb{S},S_p,S_{p+1}}_{\mathbb{T},T_p,T_{p+1}}$ and $\forall j \in [q], \mathcal{F}_j := \mathcal{R}^{(\mathbb{S},S_p,S_{p+1})_{I_j}}_{(\mathbb{T},T_p,T_{p+1})_{I_j}}$, thanks to Lemma 22 (since $(\mathbb{S}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (\mathbb{T}, T_p, T_{p+1})$) in order to write:

$$\Omega_W(\mathcal{R}^{\mathbb{S},S_0}_{\mathbb{T},T_0}) = \sum_{\substack{S_p \uplus S_{p+1} = S_0 \\ (\mathbb{S}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (\mathbb{T}, T_p, T_{p+1})}} \left(\prod_{j=1}^{q} \Omega_W(\mathcal{R}^{(\mathbb{S},S_p,S_{p+1})_{I_j}}_{(\mathbb{T},T_p,T_{p+1})_{I_j}})\right)$$

Which was what we wanted to prove.

□

Lemma 25. *Let $(A, +, \times, 0_A, 1_A)$ be a semiring, B a set, Ω a A-pre-morphism with weights in B, $W \in B^{V_G \times V_H}$, G and H be two* **e***-free edge labelled graphs, and $(H_m, \dots, H_1)$ be a contraction sequence of H.*

For all $k \in [m]$, for all **e***-connected component $\{T_1, \dots, T_p\}$ of H_k, and for all $S_1, \dots, S_p$ pairwise disjointed subsets of V_G, after the iteration of index k (and before the iteration of index $k-1$) of the loop "for $k = m-1$ downto 1" of Algorithm 1, OMEGA $\begin{bmatrix} S_1 & T_1 \\ \vdots & \vdots \\ S_p & T_p \end{bmatrix}$ contains $\Omega_W(\mathcal{R}^{S_1,\dots,S_p}_{T_1,\dots,T_p})$.*

Proof. We proceed by induction over $k = m$ down to 1.

Initialisation: For $k = m$, it comes the loop: "for $S \subseteq V_G$", noticing that the **e**-connected components of H_m are exactly its singletons of vertices, ie, the $\{a\}$ for $a \in V_H$, and noticing that, for all $S \subseteq V_G$,

$\mathcal{R}^S_{\{a\}} = \{f^S_{\{a\}}\}$ if $\forall (u,v) \in S^2, (l_G(u,v), l_H(a,a)) \in \mathcal{R}$, and $\mathcal{R}^S_{\{a\}} = \emptyset$ otherwise, recalling that $\Omega_W(\emptyset) = 0_A$ (third axiom of pre-morphisms).

Hereditary: Assume the lemma is true for $k+1$. Assume that the vertices merged in the contraction $H_{k+1} \to H_k$ are called T_p and T_{p+1} and that the merged vertex is called T_0. Notice that the only **e**-connected component of H_k that is not also a **e** connected component of H_{k+1} is the one that contains T_0. Call it $C = \{T_0, T_1, \dots, T_{p-1}\}$. Let $C_1 \uplus \dots \uplus C_q := \{T_1, \dots, T_{p-1}, T_p, T_{p+1}\}$ be the partitioning of $(C \setminus \{T_0\}) \cup \{T_p, T_{p+1}\}$ into **e**-connected component in H_{k+1}. For every $j \in [q]$, recalling that $C_j = \{T_{I_j[1]}, \dots, T_{I_j[p_j]}\}$ (by definition of I_j, and denoting $p_j := |I_j|$), we have by the induction hypothesis: OMEGA $\begin{bmatrix} S_{I_j[1]} & T_{I_j[1]} \\ \vdots & \vdots \\ S_{I_j[p_j]} & T_{I_j[p_j]} \end{bmatrix}$ contains $\Omega_W(\mathcal{R}^{\mathbb{S}_{I_j}}_{\mathbb{T}_{I_j}})$, where, we recall, $\mathcal{R}^{\mathbb{S}_{I_j}}_{\mathbb{T}_{I_j}}$ is a notation for $\mathcal{R}^{S_{I_j[1]},\dots,S_{I_j[p_j]}}_{T_{I_j[1]},\dots,T_{I_j[p_j]}}$. Then, we see that, at the end of the loop of index k (and thus at the beginning of the loop of index $k-1$), "for all $S_0, \dots, S_{p-1}$ subsets of V_G pairwise disjointed": OMEGA $\begin{bmatrix} S_0 & T_0 \\ S_1 & T_1 \\ \vdots & \vdots \\ S_{p-1} & T_{p-1} \end{bmatrix}$ contains $\sum_{\substack{S_p \uplus S_{p+1} = S_0 \\ (\mathbb{S}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (\mathbb{T}, T_p, T_{p+1})}} (\prod_{j=1}^{q} \Omega_W(\mathcal{R}^{(\mathbb{S}, S_p, S_{p+1})_{I_j}}_{(\mathbb{T}, T_p, T_{p+1})_{I_j}}))$ which, by Theorem 24, equals $\Omega_W(\mathcal{R}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0})$. OMEGA $\begin{bmatrix} S_0 & T_0 \\ S_1 & T_1 \\ \vdots & \vdots \\ S_{p-1} & T_{p-1} \end{bmatrix}$ contains $\Omega_W(\mathcal{R}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0})$ at the end of the loop of index k. Recall that $\mathcal{R}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0}$ is a notation for $\mathcal{R}^{S_1, \dots, S_{p-1}, S_0}_{T_1, \dots, T_{p-1}, T_0} = \mathcal{R}^{S_0, S_1, \dots, S_{p-1}}_{T_0, T_1, \dots, T_{p-1}}$. This concludes the proof. □

Theorem 16. *Let $(A, +, \times, 0_A, 1_A)$ be a semiring, and Ω a A-pre-morphism with weights in a set B, H an **e**-free edge-labelled graph, $\mathbb{H}$ an optimal contraction sequence of H, and X a finite set, $\mathcal{R} \subseteq X \times X_H$. Then, Algorithm 1 with fixed $\mathbb{H}$, $\mathcal{R}$ and Ω solves Ω(H-($\mathcal{R}$-MORPHISM)) in time $O((ctww(H)+2)^{|V_G|} \times |V_G|^2)$ every intance G on $n \geq 1$ vertices.*

Proof. The soundness of Algorithm 1 is a consequence of Lemma 25, noticing that the set of solutions of the instance G of H-($\mathcal{R}$-MORPHISM) is exactly $\mathcal{R}^{V_G}_{V_H}$, and that $\{V_H\}$ is a connected component of H_1.

For the complexity, note that there are $(p+2)^{|V_G|}$ ways to choose $S_0, S_1, \dots, S_{p-1}$ and S_p, S_{p+1} such that $S_0, S_1, \dots, S_{p-1}$ are pairwise disjointed subsets of V_G and (S_p, S_{p+1}) partitions S_0, and that the maximum p that will occur in the execution of Algorithm 1 is exactly $ctw(\mathbb{H}) = ctww(H)$. Also, checking if $(S_1, \dots, S_{p-1}, S_p, S_{p+1}) \preceq_{\mathcal{R}} (T_1, \dots, T_{p-1}, T_p, T_{p+1})$ can be performed in $O(|V_G|^2)$ time. □

Appendix D. Soundness of Algorithm 2

Definition 26. Let G and H two **e**-free edge-labelled graphs on n and m vertices respectively. Let $\mathcal{R} \subseteq X_G \times X_H$. Let $\mathbb{S} = (S_1, \dots, S_p)$ be a tuple of p pairwise disjointed subsets of V_G (in particular, $S_1, \dots, S_p$ can be different vertices of a contraction G' of G) and $S = S_1 \uplus \dots \uplus S_p$. Let $\mathbb{T} = (T_1, \dots, T_p)$ be a tuple of p non-empty subsets of V_G and $T = T_1 \uplus \dots \uplus T_p$.

We then define the set $\overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}}$ by

$$\overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}} = \{f : G[S] \underset{\mathcal{R}}{\to} H[T] \mid f(S_1) = T_1, \dots, f(S_p) = T_p\}.$$

Lemma 27. *Let G and H two **e**-free edge-labelled graphs, on n and m vertices respectively, G' be a contraction of G, $\mathcal{R} \subseteq X_G \times X_H$, $C_1, \dots, C_q$ be q different **e**-connected components of G' ($q \geq 1$), $\{S_1, \dots, S_p\} = C_1 \uplus \dots \uplus C_q$ let $\mathbb{S} = (S_1, \dots, S_p)$, $\mathbb{T} = (T_1, \dots, T_p)$ be a tuple of p non-empty subsets of V_H. Let for all $j \in q$, $I_j \subseteq [p]$ be such that $C_j = \{S_i, i \in I_j\}$. Then*

$$\overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}} = \begin{cases} \emptyset & \text{if } \mathbb{T} \not\succeq_{\mathcal{R}} \mathbb{S} \\ \overline{\mathcal{R}}^{\mathbb{S}_{I_1}}_{\mathbb{T}_{I_1}} \bowtie \dots \bowtie \overline{\mathcal{R}}^{\mathbb{S}_{I_q}}_{\mathbb{T}_{I_q}} & \text{if } \mathbb{T} \succeq_{\mathcal{R}} \mathbb{S} \end{cases}$$

Create a tabular $\overline{\text{OMEGA}}$ filled with 0_A
for $s \in V_G$, $t \in V_H$ **do**
  if $(l_G(s,s), l_H(t,t)) \in \mathcal{R}$ **then**
    $\overline{\text{OMEGA}}\begin{bmatrix}\{s\} & \{t\}\end{bmatrix} \leftarrow \Omega_W(\{f^{\{s\}}_{\{t\}}\})$ (with $f^{\{s\}}_{\{a\}} : \begin{matrix}\{s\} & \mapsto & \{a\}\\ s & \mapsto & a\end{matrix}$)
  end
end
for $k = n-1$ *downto* 1 **do**
  $(S_p, S_{p+1}) \leftarrow$ contracted pair in the contraction $G_{k+1} \to G_k$ of $\mathbb{G}$ $S_0 \leftarrow$ contraction of S_p and S_{p+1} in G_k $C = \{S_0, S_1, \dots S_{p-1}\}$ the e-connected component of G_k containing S_0
  $C_1 \uplus \dots \uplus C_q = \{S_1, \dots, S_{p-1}, S_p, S_{p+1}\}$ be the partitionning of $(C \setminus \{S_0\}) \cup \{S_p, S_{p+1}\}$ into e-connected component in G_{k+1}
  for $j = 1$ *to* q **do**
    Define $I_j \subseteq [p]$ such that $C_j = \{S_i, i \in I_j\}$
    $I_j =: \{I_j[1], \dots, I_j[p_j]\}$ with $p_j = |I_j|$
  end
  for $T_0, T_1, \dots, T_{p-1} \subseteq V_H$ *with* $\emptyset \notin \{T_0, T_1, \dots, T_{p-1}\}$ **do**
    for $T_p \cup T_{p+1} = T_0$ *with* $\emptyset \notin \{T_p, T_{p+1}\}$ **do**
      if $(T_1, \dots, T_{p-1}, T_p, T_{p+1}) \succeq_{\mathcal{R}} (S_1, \dots, S_{p-1}, S_p, S_{p+1})$ **then**

$$\overline{\text{OMEGA}}\begin{bmatrix} S_0 & T_0 \\ S_1 & T_1 \\ \vdots & \vdots \\ S_{p-1} & T_{p-1}\end{bmatrix} += \overline{\text{OMEGA}}\begin{bmatrix} S_{I_1[1]} & T_{I_1[1]} \\ \vdots & \vdots \\ S_{I_1[p_1]} & T_{I_1[p_1]}\end{bmatrix} \times \dots \times \overline{\text{OMEGA}}\begin{bmatrix} S_{I_q[1]} & T_{I_q[1]} \\ \vdots & \vdots \\ S_{I_q[p_q]} & T_{I_q[p_q]}\end{bmatrix}$$

      end
    end
  end
  return $\sum_{T \subseteq V_H} \overline{\text{OMEGA}}\begin{bmatrix}V_G & T\end{bmatrix}$
end

Algorithm 2: Solving $\Omega(H\text{-}(\mathcal{R}\text{-MORPHISM}))$ ie. $\Omega(\text{BINARY-CSP}(\Gamma))$ (parameterized version)

Proof. First, assume that $\mathbb{T} \not\succeq_{\mathcal{R}} \mathbb{S}$:

Assume by contradiction that there exists $f \in \overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}}$, ie. f is an $\mathcal{R}$-morphism from the edge-labeled graph $G[S_1 \uplus \dots \uplus S_p]$ to the edge-labeled graph H and $\forall i \in [p], f(S_i) = T_i$. Since $\mathbb{T}$ does not make $\mathbb{S}$ $\mathcal{R}$-feasible, there exists a pair $(i, i') \in [p]^2$ such that:

- $l_{G'}(S_i, S_{i'}) \neq$ e,
- there exists $a \in T_i$ and $b \in T_{i'}$ with $(l_{G'}(S_i, S_{i'}), l_H(a,b)) \notin \mathcal{R}$.

By Definition 7, $\forall (u,v) \in S_i \times S_{i'}, l_G(u,v) = l_{G'}(S_i, S_{i'})$. Notice that, by definition of $f \in \overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}}$, since $f(S_i) = T_i$ and $f(S_{i'}) = T_{i'}$, there exists $(u,v) \in S_i \times S_{i'}$ such that $(f(u), f(v)) = (a,b)$. Since f is an $\mathcal{R}$-morphism, we have $(l_G(u,v), l_H(f(u), f(v))) = (l_{G'}(S_i, S_{i'}), l_H(a,b)) \in \mathcal{R}$, which contradicts the definition of (a,b). We have a contradiction, which proves that $\overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}} = \emptyset$.

Second, assume that $\mathbb{T} \succeq_{\mathcal{R}} \mathbb{S}$:

Take $f \in \overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}}$ and $j \in [q]$. Since restrictions of $\mathcal{R}$-morphisms are $\mathcal{R}$-morphisms, $f|_{\cup \mathbb{S}_{I_j}}$ is an $\mathcal{R}$-morphism. Moreover, since $f \in \overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}}$, for all $j \in I_j$, $f(S_j) = T_j$. We deduce that $f|_{\cup \mathbb{S}_{I_j}} \in \overline{\mathcal{R}}^{\mathbb{S}_{I_j}}_{\mathbb{T}_{I_j}}$, which proves that $\overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}} \subseteq \overline{\mathcal{R}}^{\mathbb{S}_{I_1}}_{\mathbb{T}_{I_1}} \bowtie \dots \bowtie \overline{\mathcal{R}}^{\mathbb{S}_{I_q}}_{\mathbb{T}_{I_q}}$.

We now prove the reverse. Let $(f_1, \dots, f_q) \in \overline{\mathcal{R}}^{\mathbb{S}_{I_1}}_{\mathbb{T}_{I_1}} \times \dots \times \overline{\mathcal{R}}^{\mathbb{S}_{I_q}}_{\mathbb{T}_{I_q}}$, and let $f = f_1 \bowtie \dots \bowtie f_q$. We will prove that $f \in \overline{\mathcal{R}}^{\mathbb{S}}_{\mathbb{T}}$. Clearly, by definition of the $\overline{\mathcal{R}}^{\mathbb{S}_{I_j}}_{\mathbb{T}_{I_j}}$ for $j \in [q]$, we have, for all $i \in [p], f(S_i) = T_i$ (knowing that $(I_1, \dots, I_q)$ is a partition of $[p]$). There only remains to prove that f is an $\mathcal{R}$-morphism.

Let $S = S_1 \uplus \dots \uplus S_p$, and $(u, v) \in S^2$. We will prove that $(l_G(u, v), l_H(f(u), f(v))) \in \mathcal{R}$. Let $(i, i') \in [p]^2$ be such that $u \in S_i$ and $v \in S_{i'}$.

(1) If there exists $j \in [q]$ such that $(i, i') \in (I_j)^2$ (i.e. if S_i and $S_{i'}$ belong to the same **e**-connected component C_j), then, $(u, v) \in (S_{I_j})^2$. It follows by definition of f that $(f(u), f(v)) = (f_j(u), f_j(v))$, and then $(l_G(u, v), l_H(f(u), f(v))) \in \mathcal{R}$ since f_j is an $\mathcal{R}$-morphism.

(2) Else, by definitions of I_j and C_j for $j \in [q]$, S_i and $S_{i'}$ are not **e**-connected in H'. We deduce that, in particular, $l_{G'}(S_i, S_{i'}) \neq$ **e**. Using Definition 7: $\forall (u, v) \in S_i \times S_{i'}, l_G(u, v) = l_{G'}(S_i, S_{i'})$. Then, $(f(u), f(v)) = (f_i(u), f_{i'}(v)) \in T_i \times T_{i'}$, thus using the hypothesis of $\mathbb{T} \succeq_{\mathcal{R}} \mathbb{S}$, we have $(l_G(u, v), l_H(f(u), f(v))) = (l_{G'}(S_i, S_{i'}), l_H(f(u), f(v))) \in \mathcal{R}$.

We have proven that $(l_G(u, v), l_H(f(u), f(v))) \in \mathcal{R}$. Hence, f is indeed an $\mathcal{R}$-morphism, which concludes the proof.

□

Lemma 28. *Let G and H be two **e**-free edge-labelled graphs, $\mathcal{R} \subseteq X_G \times X_H$, $T_0, T_1, \dots, T_{p-1}$ be p non-empty subsets of V_H, $\mathbb{T} = (T_1, \dots, T_{p-1})$, $S_0, S_1, \dots, S_{p-1}$ be p pairwise disjointed subsets of V_G, $\mathbb{S} = (S_1, \dots, S_{p-1})$, and $S_p \uplus S_{p+1} = S_0$ be a partition of S_0. Then*

$$\overline{\mathcal{R}}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0} = \biguplus_{\substack{T_p \cup T_{p+1} = T_0 \\ (\mathbb{T}, T_p, T_{p+1}) \succeq_{\mathcal{R}} (\mathbb{S}, S_p, S_{p+1})}} \overline{\mathcal{R}}^{\mathbb{S}, S_p, S_{p+1}}_{\mathbb{T}, T_p, T_{p+1}}$$

Proof. Partitioning the set $\overline{\mathcal{R}}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0}$ in equivalence classes with respect to the equivalence relation $\sim$ defined by:

$$\forall (f, f') \in (\mathcal{R}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0})^2, f \sim f' \iff (f(S_p), f(S_{p+1})) = (f'(S_p), f'(S_{p+1}))$$

we obtain:

$$\overline{\mathcal{R}}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0} = \biguplus_{T_p \cup T_{p+1} = T_0} \overline{\mathcal{R}}^{\mathbb{S}, S_p, S_{p+1}}_{\mathbb{T}, T_p, T_{p+1}}$$

and then, the conclusion follows from Lemma 27.

□

Theorem 29. *Let Ω be a semiring pre-morphism and W a weight matrix. Let G and H be two **e**-free edge-labelled graphs, $(G_n, \dots, G_1)$ a contraction sequence of G (with $n := |V_G|$), $k \in [n-1]$, $C_1, \dots, C_q$ be q ($q \geq 1$) different **e**-connected components of G_{k+1} ($q \geq 1$), $\{S_1, \dots, S_{p-1}, S_p, S_{p+1}\} = C_1 \uplus \dots \uplus C_q$, $\mathbb{S} = (S_1, \dots, S_{p-1})$, and $T_0, T_1, \dots, T_{p-1}$ be p non-empty subsets of V_H, and let $\mathbb{T} = (T_1, \dots, T_{p-1})$. Denote for all $j \in q$, $I_j \subseteq [p+1]$ such that $C_j = \{S_i, i \in I_j\}$. Assume that G_k is obtained from G_{k+1} by merging the two different vertices S_p and S_{p+1} to the vertex $S_0 = S_p \uplus S_{p+1}$. Then, we have:*

$$\Omega_W(\overline{\mathcal{R}}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0}) = \sum_{\substack{T_p \cup T_{p+1} = T_0 \\ (\mathbb{T}, T_p, T_{p+1}) \succeq_{\mathcal{R}} (\mathbb{S}, S_p, S_{p+1})}} \prod_{j=1}^{q} (\Omega_W(\overline{\mathcal{R}}^{(\mathbb{S}, S_p, S_{p+1})_{I_j}}_{(\mathbb{T}, T_p, T_{p+1})_{I_j}})$$

(where $\sum$ and $\prod$ refers to the sum and product of the semiring).

Proof. Using Lemma 28:

$$\Omega_W(\overline{\mathcal{R}}^{\mathbb{S}, S_0}_{\mathbb{T}, T_0}) = \Omega_W(\biguplus_{\substack{T_p \cup T_{p+1} = T_0 \\ (\mathbb{T}, T_p, T_{p+1}) \succeq_{\mathcal{R}} (\mathbb{S}, S_p, S_{p+1})}} \overline{\mathcal{R}}^{\mathbb{S}, S_p, S_{p+1}}_{\mathbb{T}, T_p, T_{p+1}})$$

using the first axiom of pre-morphisms:

$$\Omega_W(\overline{\mathcal{R}}^{\mathbb{S},S_0}_{\mathbb{T},T_0}) = \sum_{\substack{T_p \cup T_{p+1} = T_0 \\ (\mathbb{T},T_p,T_{p+1}) \succeq_{\mathcal{R}} (\mathbb{S},S_p,S_{p+1})}} \Omega_W(\overline{\mathcal{R}}^{\mathbb{S},S_p,S_{p+1}}_{\mathbb{T},T_p,T_{p+1}})$$

and we can iteratively apply the second axiom of pre-morphisms with $\mathcal{F} := \overline{\mathcal{R}}^{\mathbb{S},S_p,S_{p+1}}_{\mathbb{T},T_p,T_{p+1}}$ and $\forall j \in [q], \mathcal{F}_j := \overline{\mathcal{R}}^{(\mathbb{S},S_p,S_{p+1})_{I_j}}_{(\mathbb{T},T_p,T_{p+1})_{I_j}}$, thanks to Lemma 22 (since $(\mathbb{T},T_p,T_{p+1}) \succeq_{\mathcal{R}} (\mathbb{S},S_p,S_{p+1})$) in order to write:

$$\Omega_W(\overline{\mathcal{R}}^{\mathbb{S},S_0}_{\mathbb{T},T_0}) = \sum_{\substack{T_p \cup T_{p+1} = T_0 \\ (\mathbb{T},T_p,T_{p+1}) \succeq_{\mathcal{R}} (\mathbb{S},S_p,S_{p+1})}} \prod_{j=1}^{q} \Omega_W(\overline{\mathcal{R}}^{(\mathbb{S},S_p,S_{p+1})_{I_j}}_{(\mathbb{T},T_p,T_{p+1})_{I_j}}),$$

which was what we wanted to prove.

□

Lemma 30. *Let $(A,+,\times,0_A,1_A)$ be a semiring, B a set, and Ω a A-pre-morphism with weights in B (with A and B sets), $W \in B^{V_G \times V_H}$, G and H be two* **e***-free edge labelled graphs, $(G_n,\dots,G_1)$ be a contraction sequence of H.*

Then, for all $k \in [m]$, for all **e***-connected component $\{S_1,\dots,S_p\}$ of G_k, and for all $T_1,\dots,T_p$ non-empty subsets of V_H, after the iteration of index k (and before the iteration of index $k-1$) of the loop "for $k = m-1$ downto 1" of Algorithm 2, $\overline{OMEGA}\begin{bmatrix} S_1 & T_1 \\ \vdots & \vdots \\ S_p & T_p \end{bmatrix}$ contains $\Omega_W(\overline{\mathcal{R}}^{S_1,\dots,S_p}_{T_1,\dots,T_p})$.*

Proof. We proceed by induction over $k = m$ downto 1.

Initialisation: For $k = m$, it comes the loop: "for $T \subseteq V_G$", noticing that the **e**-connected components of G_n are exactly its singletons of vertices, ie, the $\{s\}$ for $s \in V_G$, and noticing that, for all $T \subseteq V_H$, $\overline{R}^{\{s\}}_T = \{f^{\{s\}}_{\{a\}}\}$ if T is a singleton containing $T =: \{a\}$ such that $\forall (u,v) \in S^2, (l_G(s,s), l_H(a,a)) \in \mathcal{R}$, and $\overline{\mathcal{R}}^{\{s\}}_T = \emptyset$ otherwise, recalling that $\Omega_W(\emptyset) = 0_A$ (third axiom of pre-morphisms).

Hereditary: Assume the lemma is true for $k+1$. Assume that the vertices merged in the contraction $G_{k+1} \to G_k$ are called S_p and S_{p+1} and that the merged vertex is called S_0. Notice that the only **e**-connected component of G_k that is not also a **e**-connected component of G_{k+1} is the one that contains S_0. Call it $C = \{S_0, S_1, \dots, S_{p-1}\}$. Let $C_1 \uplus \dots \uplus C_q = \{S_1,\dots,S_{p-1},S_p,S_{p+1}\}$ be the partitionning of $(C \setminus \{S_0\}) \cup \{S_p, S_{p+1}\}$ into **e**-connected component in G_{k+1}. For every $j \in [q]$, recalling that $C_j = \{S_{I_j[1]},\dots,S_{I_j[p_j]}\}$ (by definition of I_j, and denoting $p_j := |I_j|$), we have by induction hypothesis: $\overline{\text{OMEGA}}\begin{bmatrix} S_{I_j[1]} & T_{I_j[1]} \\ \vdots & \vdots \\ S_{I_j[p_j]} & T_{I_j[p_j]} \end{bmatrix}$ contains $\Omega_W(\overline{\mathcal{R}}^{\mathbb{S}_{I_j}}_{\mathbb{T}_{I_j}})$, where, we recall, $\overline{\mathcal{R}}^{\mathbb{S}_{I_j}}_{\mathbb{T}_{I_j}}$ is a notation for $\overline{\mathcal{R}}^{S_{I_j[1]},\dots,S_{I_j[p_j]}}_{T_{I_j[1]},\dots,T_{I_j[p_j]}}$. Then, we see that, at the end of the loop of index k (and thus at the beginning of the loop of index $k-1$): "for all $T_0,\dots,T_{p-1}$ subsets of V_H": $\overline{\text{OMEGA}}\begin{bmatrix} S_0 & T_0 \\ S_1 & T_1 \\ \vdots & \vdots \\ S_{p-1} & T_{p-1} \end{bmatrix}$ contains $\sum_{\substack{T_p \cup T_{p+1} = T_0 \\ (\mathbb{T},T_p,T_{p+1}) \succeq_{\mathcal{R}} (\mathbb{S},S_p,S_{p+1})}} (\prod_{j=1}^{q} \Omega_W(\overline{\mathcal{R}}^{(\mathbb{S},S_p,S_{p+1})_{I_j}}_{(\mathbb{T},T_p,T_{p+1})_{I_j}}))$ which, by Theorem 29, equals $\Omega_W(\overline{\mathcal{R}}^{\mathbb{S},S_0}_{\mathbb{T},T_0})$. $\overline{\text{OMEGA}}\begin{bmatrix} S_0 & T_0 \\ S_1 & T_1 \\ \vdots & \vdots \\ S_{p-1} & T_{p-1} \end{bmatrix}$ contains $\Omega_W(\overline{\mathcal{R}}^{\mathbb{S},S_0}_{\mathbb{T},T_0})$ at the end of the loop of index k. Recall that $\overline{\mathcal{R}}^{\mathbb{S},S_0}_{\mathbb{T},T_0}$ is a notation for $\overline{\mathcal{R}}^{S_1,\dots,S_{p-1},S_0}_{T_1,\dots,T_{p-1},T_0} = \overline{\mathcal{R}}^{S_0,S_1,\dots,S_{p-1}}_{T_0,T_1,\dots,T_{p-1}}$. This concludes the proof.

□

Theorem 15. *Let $(A, +, \times, 0_A, 1_A)$ be a semiring, Ω a poly-time computable strong A-pre-morphism with weights in a set B, H an e-free edge-labelled graph, X a finite set, and $\mathcal{R} \subseteq X \times X_H$. Assume that Ω is corestriction independent. Then, for every instance G on $n \geq 1$ vertices, Algorithm 2 with fixed, H, $\mathcal{R}$ and Ω solves Ω(H-($\mathcal{R}$-MORPHISM)) in $O(((2^{|V_H|}-1)^{ctww(G)}) \times |V_G|^2)$ time, provided that an optimal contraction sequence $\mathbb{G}$ of G is given.*

Proof. The complexity comes from the fact that there are $(2^{|V_H|}-1)^{p+1}$ ways to choose $(T_0, T_1 \ldots, T_{p-1})$ and (T_p, T_{p+1}) such that $(T_0, T_1 \ldots, T_{p-1})$ are non-empty subsets of V_H and (T_p, T_{p+1}) are non-empty subsets of V_H with $T_p \cup T_{p+1} = T_0$. Note that, using the axiom relative to $\uplus$ and $+$ and the hypothesis that Ω is corestriction independent, it follows that for all $\mathcal{F} \in T^S$ with $\forall f \in \mathcal{F}, f(S) \subseteq T'$, denoting $\mathcal{F}|^{T'} = \{f|^{T'}, f \in \mathcal{F}\}$, we have $\Omega_W(\mathcal{F}) = \Omega_W(\mathcal{F}|^{T'})$.

The soundness of Algorithm 2 follows from the observation that, partitioning the set of solution SOL of the instance G of H-($\mathcal{R}$-MORPHISM):

$$\text{SOL} = \biguplus_{T \subseteq V_H} \{f \in \mathcal{R}_{V_H}^{V_G} \mid f(V_G) = T\}$$

which leads to

$$\Omega_W(\text{SOL}) = \sum_{T \subseteq V_H} \Omega_W(\{f \in \mathcal{R}_{V_H}^{V_G} \mid f(V_G) = T\})$$

Using the above remark:

$$\Omega_W(\text{SOL}) = \sum_{T \subseteq V_H} \Omega_W(\{f \in \mathcal{R}_{V_H}^{V_G} \mid f(V_G) = T\}|^T)$$

from where we deduce:

$$\Omega_W(\text{SOL}) = \sum_{T \subseteq V_H} \Omega_W(\overline{\mathcal{R}}_T^{V_G})$$

To obtain the soundness of Algorithm 2, it only remains to notice that, since V_G is a connected component of G_1, we deduce from Lemma 30 that $\overline{\text{OMEGA}}\begin{bmatrix} V_G & T \end{bmatrix}$ contains exactly $\Omega_W(\overline{\mathcal{R}}_T^{V_G})$ for all $T \subseteq V_H$. □

Appendix E. Examples of Pre-Morphisms and Associated Problems

In this section we introduce additional examples of pre-morphisms and explicitly show how to formulate various BINARY-CSP(Γ) problems in the Ω(H-($\mathcal{R}$-MORPHISM)) framework. One of the smallest example of a non-trivial pre-morphism is the following Boolean pre-morphism:

Lemma 31. *Let $\mathbf{2} = \{0, 1\}$ be the set of the two Booleans and $\vee$ and $\wedge$ denote disjunction and conjunction over $\mathbf{2}$. Let $\mathbf{1}_{\neq\emptyset}$ be the function that maps $\emptyset$ to 0 and every other set $\mathcal{F}$ to 1 (the weight matrix W is inessential). Then, $(\mathbf{2}, \vee, \wedge, 0, 1)$ is a semiring (even a dioid), and the function $\mathbf{1}_{\neq\emptyset}$ is a strong poly-time computable $\mathbf{2}$-pre-morphism ignoring weights.*

Note that computing the value by the function $\mathbf{1}_{\neq\emptyset}$ of the set of solutions of a BINARY-CSP (or a MORPHISM) instance is equivalent to solving its decision version. Therefore, the $\mathbf{1}_{\neq\emptyset}$(BINARY-CSP($\Gamma$)) and $\mathbf{1}_{\neq\emptyset}$($H$-($\mathcal{R}$-MORPHISM)) problems are exactly the BINARY-CSP(Γ) and H-($\mathcal{R}$-MORPHISM) problems. We can do better and implement the list version:

Lemma 32. *Let list be the function that associates to every $\mathcal{F} \in \mathcal{P}(T^S)$ (for all sets S and T) and $W \in \mathbf{2}^{S \times T}$: 0 if $\{f \in \mathcal{F} \mid \forall u \in S, W(u, f(u)) = 1\}$ is empty, and 1 otherwise. Then list is a poly-time computable strong $\mathbf{2}$-pre-morphism with weights in $\mathbf{2}$.*

It is easy to see that the list(BINARY-CSP(Γ)) and list(H-($\mathcal{R}$-MORPHISM)) problems are exactly the list-BINARY-CSP(Γ) and list-H-($\mathcal{R}$-MORPHISM) problems. We can also define a pre-morphism around the function computing the cardinal of the set involved:

Lemma 33. *Let # be the function that associate to every set $\mathcal{F}$ the cardinal $|\mathcal{F}|$ of $\mathcal{F}$ (the weight matrix W is inessential). Then $(\mathbb{N}, +, \times, 0_{\mathbb{N}}, 1_{\mathbb{N}})$ is a semiring (it is even a dioid), and # is a poly-time computable $\mathbb{N}$-pre-morphism ignoring weights.*

It follows that #(BINARY-CSP(Γ)) and #(H-($\mathcal{R}$-MORPHISM)) are alternative formulations of the #BINARY-CSP(Γ) and #H-($\mathcal{R}$-MORPHISM) counting problems.

To go even further, we can even add a list to the counting version.

Lemma 34. *Let $\#_{list}$ be the function that associates to every $\mathcal{F} \in \mathcal{P}(T^S)$ and $W \in \mathbf{2}^{S\times T}$ (for all sets S and T and $W \in \mathbf{2}^{S\times T}$) the cardinal of the set of the functions f of $\mathcal{F}$ satisfying $\forall u \in S, W(u, f(u)) = 1$: $\#_{list_W}(\mathcal{F}) = |\{f \in \mathcal{F} \mid \forall u \in S, W(u, f(u)) = 1\}|$. Then $\#_{list}$ is a poly-time computable strong $\mathbb{N}$-pre-morphism with weights in $\mathbf{2}$.*

The next semiring and pre-morphism are more subtle, since they actually takes into account the weight matrix with weights in $\overline{\mathbb{R}}$, and additionaly make use of a Cartesian product. Define

$$\oplus : \begin{array}{ccc} (\overline{\mathbb{R}} \times \mathbb{N})^2 & \mapsto & \overline{\mathbb{R}} \times \mathbb{N} \\ (m_1, c_1), (m_2, c_2) & \mapsto & (\min(m_1, m_2), \left\{ \begin{array}{l} c_1 \text{ if } m_1 < m_2 \\ c_2 \text{ if } m_2 < m_1 \\ c_1 + c_2 \text{ if } m_1 = m_2 \end{array} \right\}) \end{array}$$

and

$$\otimes : \begin{array}{ccc} (\overline{\mathbb{R}} \times \mathbb{N})^2 & \mapsto & \overline{\mathbb{R}} \times \mathbb{N} \\ (m_1, c_1), (m_2, c_2) & \mapsto & (m_1 + m_2, c_1 \times c_2) \end{array}$$

We immediately obtain the following lemma.

Lemma 35. *$((\overline{\mathbb{R}} \times \mathbb{N}), \oplus, \otimes, (+\infty, 0_{\mathbb{N}}), (0_{\overline{\mathbb{R}}}, 1_{\mathbb{N}}))$ is a semiring (it is even a dioid).*

Definition 36. We define the following functions.

- Let MinCost be the function that maps, for G and H two edge-labelled graphs, $S \subseteq V_G$, $T \subseteq V_H$, any $(\mathcal{F}, W)$ to $\min W_{\sum}(\mathcal{F})$, for $\mathcal{F} \in \mathcal{P}(T^S)$ and $W \in \overline{\mathbb{R}}^{V_G \times V_H}$ denoting, for all $f \in T^S$, $W_{\sum}(f) = \sum_{u\in S} W(u, f(u))$, and $W_{\sum}(\mathcal{F}) = \{W_{\sum}(f) \mid f \in \mathcal{F}\}$: MinCost outputs values in $\overline{\mathbb{R}}$).
- Let ArgMinCost be the function that maps, for G and H two edge-labelled graphs, $S \subseteq V_G$, $T \subseteq V_H$, any $(\mathcal{F}, W)$ to the set $\left\{ \begin{array}{l} \emptyset \text{ if } \min W_{\sum}(\mathcal{F}) = +\infty \\ \{f \in \mathcal{F} \mid W_{\sum}(f) = \min W_{\sum}(\mathcal{F})\} \text{ otherwise} \end{array} \right\}$ with $\mathcal{F} \in \mathcal{P}(T^S)$ and $W \in \overline{\mathbb{R}}^{V_G \times V_H}$
- Last, let #ArgMinCost be the composition of # and ArgMinCost (i.e., the function that outputs the cardinal of the set described above): #ArgMinCost outputs values in $\mathbb{N}$.

Lemma 37. *(MinCost, #ArgMinCost) is a poly-time computable strong $(\overline{\mathbb{R}} \times \mathbb{N})$-pre-morphism with weights in $\overline{\mathbb{R}}$.*

Notice that computing the value by the function $(MinCost, \#ArgMinCost)$ of the set of solutions of a BINARY-CSP or a MORPHISM instance means computing the minimal weight $\sum_{u\in V_G} w(u, f(u))$ of the solutions f satisfying $\forall (u, v) \in V_G \times V_H, w(u, v) = +\infty \implies f(u) \neq v$, and determining the number of such solutions of minimal weights, which answers a problem that subsumes together the counting, the list, and a weighted version of BINARY-CSP(Γ) and H-($\mathcal{R}$-MORPHISM). We also remark that instead of considering the weights in $\overline{\mathbb{R}}$, considering any totally ordered set B with an increasing, binary, associative and commutative operation $b : B^2 \mapsto B$ (instead of $+$) would have been possible. We present a variant of this pre-morphism, with the goal of modeling another weighted version of BINARY-CSP(Γ) and H-($\mathcal{R}$-MORPHISM).

Definition 38. Let MinWeight be the function that maps, for G and H two edge-labelled graphs, $S \subseteq V_G$, $T \subseteq V_H$, any $(\mathcal{F}, W)$ to $\min W_{\max}(\mathcal{F})$, for $\mathcal{F} \in \mathcal{P}(T^S)$ and $W = (w(u, v))_{u\in V_G, v\in V_H} \in \overline{\mathbb{R}}^{V_G \times V_H}$ denoting, for all $f \in T^S$, $W_{\max}(f) = \sum_{v\in T} \max_{u\in f^{-1}(v)} W(u, v)$, and $W_{max}(\mathcal{F}) = \{W_{\max}(f) \mid f \in \mathcal{F}\}$: MinWeight outputs values in $\overline{\mathbb{R}}$).

Let ArgMinWeight be the function that maps, for G and H two edge-labelled graphs, $S \subseteq V_G$, $T \subseteq V_H$, any $(\mathcal{F}, W)$ to the set $\left\{ \begin{array}{l} \emptyset \text{ if } \min W_{\max}(\mathcal{F}) = +\infty \\ \{f \in \mathcal{F} \mid W_{\max}(f) = \min W_{\max}(\mathcal{F})\} \text{ otherwise} \end{array} \right\}$ with $\mathcal{F} \in \mathcal{P}(T^S)$ and $W \in \overline{\mathbb{R}}^{V_G \times V_H}$.

and let #ArgMinWeight be the composition of # and ArgMinWeight (ie the function that outputs the cardinal of the set described above): #ArgMinWeight outputs values in $\mathbb{N}$.

Similarly, computing the value by MinWeight of the set of solutions of an instance of BINARY-CSP(Γ) or H-($\mathcal{R}$-MORPHISM), solves another weighted version of the problems, which is the one defined by Escoffier et al. [25] (restricted to the less general case of q-COLORING) while computing both the minimal weight of a solution and the number of solutions with such a minimal weight.

Lemma 39. *$((\overline{\mathbb{R}}_+ \times \mathbb{N}), \oplus, \otimes, (+\infty, 0_{\mathbb{N}}), (0_{\overline{\mathbb{R}}_+}, 1_{\mathbb{N}}))$ is a semiring, and (MinWeight, #ArgMinWeight) is a poly-time computable $(\overline{\mathbb{R}}_+ \times \mathbb{N})$-pre-morphism (restricting and corestricting $\oplus$ and $\otimes$ to $(\overline{\mathbb{R}}_+ \times \mathbb{N})^2$ and $(\overline{\mathbb{R}}_+ \times \mathbb{N})$). It is not a strong pre-morphism.*

Finally, we describe a last dioid, that adds a constraint on the preimages of the vertices of the target graph.

Definition 40. Let $\#restrictive-list$ be the function that maps, for G and H two edge-labelled graphs, $S \subseteq V_G$, $T \subseteq V_H$, any $(\mathcal{F}, W)$ to $|\{f \in \mathcal{F} \mid \forall v \in T, w(v) \neq +\infty \implies |f^{-1}(\{v\})| = W_1(v), \forall (u,v) \in S \times T, f(u) = v \implies W_2(u,v) = 1\}|$, for $\mathcal{F} \in \mathcal{P}(T^S)$ and $W = (W_1(v), W_2(u,v))_{u \in V_G, v \in V_H} \in (\overline{\mathbb{N}} \times \mathbf{2})^{V_G \times V_H}$ with $W_1 = (W_1(v))_{u \in V_G, v \in V_H} \in \overline{\mathbb{N}}^{V_G \times V_H}$ a matrix of constant columns and $W_2 \in \mathbf{2}^{V_G \times V_H}$.

Lemma 41. *The $\#restrictive-list$ is a poly-time computable $\mathbb{N}$-pre-morphism with weights in $\overline{\mathbb{N}} \times \mathbf{2}$.*

Computing the value by $\#restrictive-list$ of the set of solutions of an instance of BINARY-CSP(Γ) or H-($\mathcal{R}$-MORPHISM), solves a stronger version of the list generalisation, which is the one defined by Diaz et. al [22] (restricted to the less general case of q-COLORING), while also counting the number of solutions. Notice that, by taking weights in $\overline{\mathbb{N}} \times \overline{\mathbb{R}}$ (instead of simply $\overline{\mathbb{N}} \times \mathbf{2}$), we could have, similarly as previously, encoded a weighted version.

References

[1] L. Barto, J. Bulín, A. Krokhin, and J. Opršal. Algebraic approach to promise constraint satisfaction. *Journal of the ACM (JACM)*, 68(4):1–66, 2021.

[2] P. Bergé, É. Bonnet, and H. Déprés. Deciding twin-width at most 4 is np-complete. In M. Bojanczyk, E. Merelli, and D. P. Woodruff, editors, *Proceedings of the 49th International Colloquium on Automata, Languages, and Programming, (ICALP-2022)*, volume 229 of *LIPIcs*, pages 18:1–18:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[3] S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint satisfaction and optimization. *Journal of the ACM (JACM)*, 44(2):201–236, 1997.

[4] A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM Journal on Computing*, 39(2):546–563, 2009.

[5] H. L. Bodlaender, E. J. v. Leeuwen, J. M. Van Rooij, and M. Vatshelle. Faster algorithms on branch and clique decompositions. In *International Symposium on Mathematical Foundations of Computer Science*, pages 174–185. Springer, 2010.

[6] É. Bonnet, D. Chakraborty, E. J. Kim, N. Köhler, R. Lopes, and S. Thomassé. Twin-width VIII: delineation and win-wins. *arXiv preprint arXiv:2204.00722*, 2022.

[7] É. Bonnet and H. Déprés. Twin-width can be exponential in treewidth. *arXiv preprint arXiv:2204.07670*, 2022.

[8] É. Bonnet, C. Geniet, E. J. Kim, S. Thomassé, and R. Watrigant. Twin-width ii: small classes. In *Proceedings of the 32nd ACM-SIAM Symposium on Discrete Algorithms (SODA-2021)*, pages 1977–1996. SIAM, 2021.

[9] E. Bonnet, C. Geniet, E. J. Kim, S. Thomassé, and R. Watrigant. Twin-width III: Max Independent Set, Min Dominating Set, and Coloring. In N. Bansal, E. Merelli, and J. Worrell, editors, *Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP-2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[10] É. Bonnet, C. Geniet, R. Tessera, and S. Thomassé. Twin-width VII: groups. *arXiv preprint arXiv:2204.12330*, 2022.

[11] É. Bonnet, U. Giocanti, P. Ossona de Mendez, P. Simon, S. Thomassé, and S. Toruńczyk. Twin-width IV: ordered graphs and matrices. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC-2022)*, pages 924–937, 2022.

[12] É. Bonnet, E. J. Kim, A. Reinald, and S. Thomassé. Twin-width vi: the lens of contraction sequences. In *Proceedings of the 33rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA-2022)*, pages 1036–1056. SIAM, 2022.

[13] É. Bonnet, E. J. Kim, A. Reinald, S. Thomassé, and R. Watrigant. Twin-width and polynomial kernels. *Algorithmica*, pages 1–38, 2022.

[14] É. Bonnet, E. J. Kim, S. Thomassé, and R. Watrigant. Twin-width I: tractable FO model checking. In *Proceedings of the 61 61st Annual Symposium on Foundations of Computer Science (FOCS-2020)*, pages 601–612. IEEE, 2020.

[15] É. Bonnet, O.-j. Kwon, D. R. Wood, et al. Reduced bandwidth: a qualitative strengthening of twin-width in minor-closed classes (and beyond). *arXiv preprint arXiv:2202.11858*, 2022.

[16] É. Bonnet, J. Nešetřil, P. O. de Mendez, S. Siebertz, and S. Thomassé. Twin-width and permutations. *arXiv preprint arXiv:2102.06880*, 2021.

[17] A. A. Bulatov. A dichotomy theorem for nonuniform csps simplified. *CoRR*, abs/2007.09099, 2020.

[18] J.-Y. Cai and X. Chen. Complexity of counting csp with complex weights. *Journal of the ACM (JACM)*, 64(3), jun 2017.

[19] B. Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990.

[20] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000.

[21] J. Díaz, M. Serna, and D. M. Thilikos. Counting h-colorings of partial k-trees. *Theoretical Computer Science*, 281(1-2):291–309, 2002.

[22] J. Díaz, M. Serna, and D. M. Thilikos. The restrictive h-coloring problem. *Discrete Applied Mathematics*, 145(2):297–305, 2005.

[23] R. G. Downey and M. R. Fellows. *Parameterized complexity*. Springer Science & Business Media, 2012.

[24] M. Dyer and C. Greenhill. The complexity of counting graph homomorphisms. *Random Structures & Algorithms*, 17(3-4):260–289, 2000.

[25] B. Escoffier, J. Monnot, and V. T. Paschos. Weighted coloring: further complexity and approximability results. *Information Processing Letters*, 97(3):98–103, 2006.

[26] F. V. Fomin, P. Heggernes, and D. Kratsch. Exact algorithms for graph homomorphisms. *Theory of Computing Systems*, 41(2):381–393, 2007.

[27] R. Ganian, T. Hamm, V. Korchemna, K. Okrasa, and K. Simonov. The fine-grained complexity of graph homomorphism parameterized by clique-width. In M. Bojanczyk, E. Merelli, and D. P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 66:1–66:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[28] P. Hell and J. Nešetřil. On the complexity of h-coloring. *Journal of Combinatorial Theory, Series B*, 48(1):92–110, 1990.

[29] D. Kobler and U. Rotics. Edge dominating set and colorings on graphs with fixed clique-width. *Discrete Applied Mathematics*, 126(2-3):197–221, 2003.

[30] K. Okrasa and P. R. ażewski. Fine-grained complexity of the graph homomorphism problem for bounded-treewidth graphs. *SIAM Journal on Computing*, 50(2):487–508, 2021.

[31] F. Rossi, P. Van Beek, and T. Walsh. *Handbook of constraint programming*. Elsevier, 2006.

[32] M. Wahlström. New plain-exponential time classes for graph homomorphism. *Theory of Computing Systems*, 49(2):273–282, 2011.

[33] N. Wilson. Decision diagrams for the computation of semiring valuations. In L. P. Kaelbling and A. Saffiotti, editors, *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-2005)*, pages 331–336. Professional Book Center, 2005.

[34] D. Zhuk. A proof of the CSP dichotomy conjecture. *Journal of the ACM (JACM)*, 67(5):30:1–30:78, 2020.

(A. Baril) Université de Lorraine, CNRS, LORIA, F-54000 Nancy, France

(M. Couceiro) Université de Lorraine, CNRS, LORIA, F-54000 Nancy, France

(V. Lagerkvist) Dep. Computer and Information Science,, Linköpings Universitet, Sweden