Theoretical Computer Science 407 (2008) 97–109

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Universal relations and #P-completeness☆

Hervé Fournier[a,*], Guillaume Malod[b]

[a] Laboratoire PRiSM, Université de Versailles Saint-Quentin en Yvelines, 45 avenue des États-Unis, 78035 Versailles Cedex, France
[b] Institut de Mathématiques, Université de Mons-Hainaut, 6 avenue du Champ de Mars, 7000 Mons, Belgique

ARTICLE INFO

Article history:
Received 21 September 2006
Received in revised form 23 April 2008
Accepted 6 May 2008
Communicated by A. Razborov

Keywords:
Computational complexity
Counting problems

ABSTRACT

This paper follows the methodology introduced by Agrawal and Biswas in [Manindra Agrawal, Somenath Biswas, Universal relations, in: Structure in Complexity Theory Conference, 1992, pp. 207–220], based on a notion of universality for the relations associated with NP-complete problems. The purpose was to study NP-complete problems by examining the effects of reductions on the solution sets of the associated witnessing relations. This provided a useful criterion for NP-completeness while suggesting structural similarities between natural NP-complete problems. We extend these ideas to the class #P. The notion we find also yields a practical criterion for #P-completeness, as illustrated by a varied set of examples, and strengthens the argument for structural homogeneity of natural complete problems.

1. Introduction

Complexity classes such as P, NP or #P are great theoretical notions to further our knowledge of the resources needed to solve computational problems. Their usefulness, however, goes beyond the theoretical setting, because knowing the right class for a given problem is a precious hint as to the kind of algorithms one should look for.

Agrawal and Biswas [1] study the structure of NP-complete sets. In a decision problem one must determine for an instance x whether there exists a y such that $R(x, y)$ holds, where R is the solution checking relation. Agrawal and Biswas focus on the relations $R(x, y)$ to which any other relation can be reduced in a way which preserves solutions, roughly meaning that there is a mapping between solutions sets. They call these relations *universal*. In the setting of polynomial time checkable relations, they show that the decision problem corresponding to a universal relation is NP-complete. By giving a simple criterion for NP-completeness based on this definition and applying it to a varied set of examples, they argue that their result underlines a structural similarity between natural NP-complete sets, in the spirit of the work of Berman and Hartmanis [3]. Indeed Agrawal and Biswas show that their notion of universality is related to structural properties such as paddability and self-reducibility.

The notion of universality was subsequently used by Buhrman et. al. [4] to provide sufficient conditions for NP optimization problems that admit efficient approximation algorithms. It was then extended by Portier [9] to problems defined on an arbitrary structure, in the framework of Poizat [8], and recently Choudhary, Sinha and Biswas [6] defined it for non-deterministic logspace. Our aim in this paper is to adapt this notion to the class #P, which is the class of functions counting the number of solutions for relations checkable in polynomial time, a class at least as hard as the polynomial hierarchy, as shown by Toda [10]. Showing that a function is in #P is thus a convincing argument for its intractability, and a criteria for #P-completeness would be a useful tool. Universality for #P has been studied in [5], where an elaborate definition

☆ This work was supported by CEFIPRA grant number 2602-1; the second author also worked on this article as a JSPS fellow (fellowship P04801).
* Corresponding author. Tel.: +33 0139254331.
E-mail addresses: Herve.Fournier@prism.uvsq.fr (H. Fournier), Guillaume.Malod@umh.ac.be (G. Malod).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.05.003

of #P universality is given, based on Valiant's technique for showing #P-completeness (cf. [11]) by recovering the coefficients of a polynomial from its value at suitable points. We give here a definition of universality suited to #P which is both simpler, and closer to the definition used for NP by Agrawal and Biswas. The relative simplicity translates into a *usable* criterion for #P-completeness.

Section 2 introduces the background notions. Section 3 defines universality for #P and shows that it implies completeness (Proposition 1). Section 4 provides the main point of the article, with the #P-universality criterion and the proof that it implies universality (Theorem 1). Proposition 1 and Theorem 1 together become a practical criterion for #P-completeness, which we apply to examples from different backgrounds in Section 5.

2. Definitions and notations

We present here the framework with which we will work. For each instance x of a problem there is a set C_x of candidate solutions and the set S_x of actual solutions. This is another way of saying that we focus on the relation between an instance x and a candidate solution y which holds iff y belongs to S_x. One can for instance consider the set of 3CNF formulas over the variables $x_1, x_2, \ldots$; for a formula F, the set of candidate solutions is the set of truth-value assignments for the variables in F. The set of solutions is the set of satisfying assignments for F. Another example is the problem of finding a maximal independent set in a graph, i.e. a maximal subset of the vertices such that no two nodes have an edge between them. The set of instances is the set of graphs, candidate solutions are subsets of the vertices and solutions are maximal independent subgraphs.

The complexity of a computational question is the growth of some computational resource when the size of the instance increases, where the size usually means the length of the encoding. In this paper however we will need a slightly more general definition of size, which we will call a measure.

Definition 1. A *measure* for a problem is a mapping m from the set of instances into $\mathbb{N}^d$ for a given d, such that there is a polynomial $p(n_1, \ldots, n_d)$, with $p(m(x))$ bounding the length of the encoding of x. For $m(x) = (m_1, \ldots, m_d)$, we will write $m(x)+n$ for the tuple $(m_1+n, \ldots, m_d+n)$ if n is an integer, and $m(x)+m(y)$ for $(m_1+n_1, \ldots, m_d+n_d)$ if $m(y) = (n_1, \ldots, n_d)$.

For instance, the number of clauses is a valid measure for a 3CNF formula F, because it gives us a bound on the length of the formula and its number of variables. Encoding a graph of size n by giving its adjacency matrix yields a word of length $O(n^2)$, and therefore taking the number n of vertices is a valid measure for graphs. We could also have chosen to measure a graph with two integers, one being the number of vertices and the other being the number of edges. One may wonder why we need the notion of measure instead of using the standard length of the encoding. There are two reasons.

The first reason is that the length of the encoding may be non-linear in the intuitive "size" of an instance. In the case of graphs represented as adjacency matrices, the length of the encoding is quadratic in the number of vertices. Jumping ahead a little bit, this would be a problem with our Couple condition from Section 4: because we will need to iterate this operation, we need to impose a constant bound on the growth of the measure. However, if we add k vertices to a graph G to get a graph G', the difference between the length of the encoding of G' and the length of the encoding of G depends not only on the number k but also on the length of the encoding of G. Iterating such an operation may not yield a suitable bound. Note that this problem already arose for Agrawal and Biswas [1], who solved it by making the Couple growth condition more complex.

The second reason is that sometimes the size of an instance depends on several independent parameters, and the freedom to use an appropriate measure may help us when we need to show that a given relation satisfies the conditions defined in Section 4, in order to prove that it yields a #P-complete problem. An example of this is the knapsack problem in Section 5.5 where we use two parameters for the measure.

In any case, we believe the notion of measure reflects a natural way of thinking about the "size" of an instance: it seems more intuitive to think of the number of vertices of a graph than of the length of a particular encoding. Thus we hope that this definition does not hinder but rather facilitates our explanations.

In the following, $|A|$ denotes the cardinality of the set A. We will consider *balanced* relations X, in the sense that all elements y in relation with an instance x of X have the same length denoted by $q^X(x)$; moreover, we ask that $|q^X(x)| \leq |x|^{O(1)}$ (this constant depending on the relation X). Thus, any potential solution for the instance x is a word of length $q^X(x)$ on the alphabet $\Sigma = \{0, 1\}$, i.e. it belongs to $\Sigma^{q^X(x)}$, which we will call the set of candidate solutions and denote as C_x^X. We will use the notation S_x^X to denote the set of solutions for the instance x via the relation X:

$$S_x^X = \{y \in C_x^X \mid (x, y) \in X\}.$$

Definition 2. A (p, q)-projection is a sequence $\alpha = (\alpha[1], \ldots, \alpha[q])$, where the $\alpha[i]$ are distinct integers in $\{1, \ldots, p\}$. Such a sequence naturally defines a function from Σ^p to Σ^q: to a word $u_1 \cdots u_p$ we associate the word $u_{\alpha[1]} \cdots u_{\alpha[q]}$. We will slightly abuse notations and also write α for the function defined by α.

We also need to define the *concatenation* of two projections. Given a (p, q)-projection α and a (p', q')-projection α', their concatenation $\beta = \alpha \cdot \alpha'$ is the $(p + p', q + q')$-projection defined by $\beta[i] = \alpha[i]$ for $1 \leq i \leq q$ and $\beta[i] = p + \alpha'[i - q]$ for $q + 1 \leq i \leq q + q'$.

3. Universality for #P

We call X a P-relation if it can be decided in polynomial time. Let us now give a definition of universality adapted to #P.

Definition 3. Let X be a balanced P-relation. X is #P-universal if for any balanced P-relation Y there exists a polynomial time computable function which, given an instance y of Y, computes an instance x of X, two integers $k, M > 0$ such that $M \cdot |\Sigma|^{q^Y(y)} < 2^k$, and a $\left(q^X(x), q^Y(y)\right)$-projection α such that if t is a solution for y, then

$$\left|\{s \in S_x^X \mid \alpha(s) = t\}\right| \equiv M \pmod{2^k},$$

and otherwise (if t is not a solution for y), then

$$\left|\{s \in S_x^X \mid \alpha(s) = t\}\right| \equiv 0 \pmod{2^k}.$$

For example, suppose that 3SAT is known to be universal, as we will see below. Then if we consider the relation associated with maximal independent sets in a graph, this means that given a graph G we can compute a 3CNF formula ϕ and a projection α which sends an assignment to the variables of ϕ to a subset of the vertices of G such that: for any maximal independent set I of G, the number satisfying assignments of ϕ which are projected to I via α is exactly M, and the number of remaining satisfying assignments for ϕ is 0, all these numbers being modulo an adequate power of 2. Now if we know the number of solutions of ϕ we can compute the number of maximal independent sets of G. In the general case this gives us Proposition 1 below.

We recall that $f \in \mathrm{PF}$ is reducible to $g \in \mathrm{PF}$ via a Cook reduction if f can be computed in polynomial time by a Turing machine using g as an oracle. Cook[1]-reductions are Cook reductions using at most one call to the oracle. The reader not familiar with the notion of oracle may consult [7].

Proposition 1. *The counting problem associated to a* #P*-universal relation is* #P*-complete for Cook[1]-reductions.*

Proof. Let X be a #P-universal relation and g the associated counting function: $g(x) = \left|S_x^X\right|$. The function g is obviously in #P.

Let Y be a P-relation and h the associated counting function. By definition of universality, there is a computable time function which given an instance y of Y computes two integers $k, M > 0$ and a projection α with the above properties. Thus,

$$\begin{aligned}\left|S_x^X\right| &= \sum_{t \in C_y^Y} |\{s \in S_x \mid \alpha(s) = t\}| \\ &= \sum_{t \in S_y^Y} \left|\{s \in S_x^X \mid \alpha(s) = t\}\right| + \sum_{t \in C_y^Y \setminus S_y^Y} \left|\{s \in S_x^X \mid \alpha(s) = t\}\right| \\ &\equiv M \cdot \left|S_y^Y\right| \pmod{2^k}.\end{aligned}$$

As $M \cdot |\Sigma|^{q^Y(y)} < 2^k$, we have $M \cdot \left|S_y^Y\right| < 2^k$. Thus $h(y) = (g(x) \bmod 2^k)/M$, and this computation can be done in polynomial time. □

4. Sufficient conditions for universality

We shall say that a balanced P-relation X has Block, Join and Couple conditions if there exists an integer $k_0 \in \mathbb{N} \setminus \{0\}$ such that the following three properties hold:

Block. There exist $M_b \in \mathbb{N} \setminus \{0\}$ and a polynomial time computable function which, given an integer $k \geqslant k_0$ in unary encoding, computes an instance b of X, a word $t_b \in \Sigma^3$ and a $\left(q^X(x), 3\right)$-projection β such that:

- for any word $u \in \Sigma^3$ different from t_b, $\left|\{s \in S_b^X \mid \beta(s) = u\}\right| \equiv M_b \pmod{2^k}$.
- $\left|\{s \in S_b^X \mid \beta(s) = t_b\}\right| \equiv 0 \pmod{2^k}$.
- $m(x) = k^{O(1)}$.

Join. There exist $M_j \in \mathbb{N} \setminus \{0\}$ and a polynomial time computable function which, given two instances x_1, x_2 of X and $k \geqslant k_0$ in unary encoding, computes an instance y of X, a $\left(q^X(y), q^X(x_1) + q^X(x_2)\right)$-projection ρ such that:

- for all $s_1 \in C_{x_1}^X$, for all $s_2 \in C_{x_2}^X$, if $s_1 \in S_{x_1}^X$ and $s_2 \in S_{x_2}^X$ then:

  $$\left|\{t \in S_y^X \mid \rho(t) = s_1 s_2\}\right| \equiv M_j \pmod{2^k},$$

  otherwise

  $$\left|\{t \in S_y^X \mid \rho(t) = s_1 s_2\}\right| \equiv 0 \pmod{2^k}.$$

- $m(y) \leqslant m(x_1) + m(x_2) + k^{O(1)}$.

Couple. There exist $M_c \in \mathbb{N} \setminus \{0\}$ and a polynomial time computable function which, given an integer $k \geqslant k_0$ in unary encoding, an instance x of X, and integers i, j with $1 \leq i < j \leq q^X(x)$, computes an instance y of X and a $\left(q^X(y), q^X(x)\right)$-projection γ such that:

- for all $s \in C_x^X$, if $s \in S_x^X$ and if the ith and jth bits of s have different values (i.e. their XOR is 1), then

$$\left|\{t \in S_y^X \mid \gamma(t) = s\}\right| \equiv M_c \pmod{2^k},$$

  otherwise:

$$\left|\{t \in S_y^X \mid \gamma(t) = s\}\right| \equiv 0 \pmod{2^k},$$

- $m(y) \leqslant m(x) + k^{O(1)}$.

The example of 3SAT should help understand these conditions and show that they can be very intuitive in the case of specific examples. We consider the three properties in turn.

Block. Consider the clause $\phi = d_1 \vee d_2 \vee d_3$, then a solution is a Boolean word of length 3 and 000 is the only word which does not yield a solution for ϕ; all the other words of Σ^3 yield exactly one solution. The general case extends this in the following ways: computations must hold only modulo a given power of 2; the word which is not a solution may be different from 000; the other words are not constrained to yielding exactly one solution but a constant number M_b instead.

Join. Given two 3CNF formulas ϕ_1 and ϕ_2, we can rename variables to ensure that ϕ_1 and ϕ_2 have no common variables. The conjunction $\psi = \phi_1 \wedge \phi_2$ is then a 3CNF formula such that: a solution for ϕ_1 coupled with a solution for ϕ_2 yields exactly one solution for ψ; the number of other solutions for ψ is 0; the measure of ψ is bounded by the sum of the measures of ϕ_1 and ϕ_2. Differences in the general case: computing modulo a given 2^k, getting a constant number of solutions M_j for each couple, allowing the measure to increase polynomially in k.

Couple. Given a 3CNF formula ϕ and two variables a and b in ϕ (which correspond to the indexes i and j in a solution for ϕ), the formula $\psi = \phi \wedge (a \vee b \vee b) \wedge (\neg a \vee \neg b \vee \neg b)$ is such that any solution of ϕ which satisfies (a XOR b) yields exactly one solution for ψ and the number of other solutions for ψ is 0. Differences in the general case: computations modulo a given 2^k, getting a constant number of solutions M_j for each solution of ϕ, allowing the measure of the new instance to increase polynomially in k.

Theorem 1. *If a balanced* P*-relation has the Block, Join and Couple conditions, it is* #P*-universal.*

Proof. Before giving the general ideas, let us first define some notions and notations we will use in this proof:

- We will use the notation $\mu_1(d_1) \vee \mu_2(d_2) \vee \mu_3(d_3)$ for a clause over the variables d_1, d_2, d_3, where $\mu_i(d_i)$ may be either d_i or $\neg d_i$. If the variables d_1, d_2, d_3 are distinct, we define the *type* of $C = \mu_1(d_1) \vee \mu_2(d_2) \vee \mu_3(d_3)$ as the only Boolean word $u \in \Sigma^3$ which is not a satisfying assignment of the clause, and we will say that u *defines* C. Note that the type of a clause can also be defined as a word whose ith letter is 0 if the ith literal of the clause is positive and 1 otherwise. For example, the type of the clause $d_1 \vee \neg d_2 \vee d_3$ is the Boolean word 010.
- The Couple condition for bit positions i and j ($i \neq j$) enables us to restrict our attention to solutions where the bits i and j take different values (i.e. one is 0 and the other is 1). We will call these solutions *coupled* for i and j. We call a set I of pairs of bits (i, j) with $i \neq j$ a *coupling set*, and we say that a solution is I*-coupled* if it is coupled for all pairs $(i, j) \in I$, in other words if for all $(i, j) \in I$ the bits i and j take different values. If it is obvious from the context, we may drop the set I and just talk of coupled solutions.

Let X be a balanced P-relation which has the properties detailed above. We wish to show that it is #P-universal. Consider another P-relation Y. Given an instance y of Y, we will describe how to compute an instance x of X, two integers k, M and a projection α satisfying the conditions stated in the definition of universality. This will be done via the following steps:

(1) From the instance y we build a 3CNF formula F_1 and a $\left(q^{3SAT}(F_1), q^Y(y)\right)$-projection α_1 such that α_1 induces a bijection from the satisfying assignments of F_1 to the solutions of y for the relation Y.

(2) We then build a 3CNF formula $F_2 = D_1 \wedge \cdots \wedge D_n$ on $3n$ distinct variables where each clause D_i is of the type given by the word t_b from the Block condition of the relation X, a $\left(q^{3SAT}(F_2), q^{3SAT}(F_1)\right)$-projection α_2 and a coupling set I such that α_2 induces a bijection between the I-coupled satisfying assignments of F_2 and the solutions of F_1.

(3) We now have expressed the solution set for the instance y as the coupled solutions of a 3CNF-formula which is made from clauses which are all of the type defined by the Block condition. We therefore use the Block condition to create instances which simulate each clause, then simulate their conjunction by bringing these instances together with the Join condition, and finally use the Couple condition to simulate the coupled assignment of F_2 by an instance x of X which has the properties required in the definition of universality.

Step 1. Recall that 3SAT is #P-complete for parsimonious reductions. This means that for any P-relation Y, there exists a polynomial time computable function which, given an instance y of Y, computes a 3CNF-formula F such that the number of solutions of y, i.e. $\left|\{z \in C_y^Y \mid Y(y, z)\}\right|$, is equal to the number of satisfying assignments of the formula F. But in fact, we have

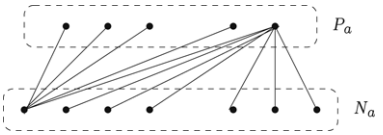

Fig. 1. Coupling the variables in F_2.

the stronger following property: we can additionally compute a $\left(q^{3SAT}(F), q^Y(y)\right)$-projection α_1 which induces a bijection from the set of satisfying assignments of F to the set of solutions of y.

This paragraph gives some details on how to compute α_1 as above. Since this argument cannot be separated from the construction of F, we explain below how to construct both F and α_1 at the same time. For this, let us recall briefly the proof of NP-completeness of 3SAT (see [2] or [7] for further details). By definition of a P-relation, there exists a Turing machine which, given an instance y and a potential solution $z = z_1 z_2 \dots z_p$ where $p = q^Y(y)$, computes $Y(y, z)$. The computation carried by this Turing machine can be transformed into a circuit $C(z_1, \dots, z_p)$ — although it does not appear in the notation, the circuit C of course depends on y. Now it is a standard technique to transform this circuit $C(z_1, \dots, z_p)$ into an equivalent existential 3CNF-formula $\exists v_{p+1} \dots v_r F(v_1, \dots, v_r)$, where the tuple of additional variables $v_{p+1}, \dots, v_r$ is needed to perform this transformation: some of these variables are used to represent the nodes of the circuit in order to write it in the form of an equivalent existential formula, while other variables are then needed to put this formula in 3CNF form. Moreover, for each value of the variables $v_1, \dots, v_p$ the following holds: if $v_1 \dots v_p \in S_y^Y$, there exists a unique assignment of the variables $v_{p+1}, \dots, v_r$ such that $F(v_1, \dots, v_r)$ evaluates to *true*; if $v_1 \dots v_p \notin S_y^Y$, the formula $F(v_1, \dots, v_r)$ evaluates to *false* for all assignments of $v_{p+1}, \dots, v_r$. The proof of #P-completeness of 3SAT for parsimonious reductions follows at once, and the $\left(q^{3SAT}(F), q^Y(y)\right)$-projection $\alpha_1 = (1, 2, \dots, p)$ has the required properties.

Recall that there is a word $t_b \in \Sigma^3$ from the definition of the Block condition. This word defines a clause and we will call ν_1, ν_2, ν_3 the associated signs. We add clauses $\nu_1(a) \vee \neg\nu_2(a) \vee \epsilon(a)$ for each variable a in F, where $\epsilon(a)$ is a if $\nu_1(a)$ is $\neg a$ and $\neg a$ otherwise. The reason for these additional clauses will be explained later. The measure of the formula F_1 thus obtained is still polynomial in the measure of y and it has the same set of solutions as F, because these additional clauses are always satisfied.

Step 2. Suppose $F_1 = C_1 \wedge \dots \wedge C_n$, and suppose each clause C_i is of the form $\mu_{i,1}(c_{i,1}) \vee \mu_{i,2}(c_{i,2}) \vee \mu_{i,3}(c_{i,3})$, where the $c_{i,j}$ are variables and $\mu_{i,j}(a)$ is either $\neg a$ or a. We now consider a 3CNF-formula F_2 over $3n$ distinct variables $d_{i,j}$ which has the same number of clauses as F_1 (and thus the same measure), but which is such that the type of each of these clauses is the word t_b given by the Block condition of X: F_2 is of the form $D_1 \wedge \dots \wedge D_n$, with $D_i = \nu_1(d_{i,1}) \vee \nu_2(d_{i,2}) \vee \nu_3(d_{i,3})$.

For $1 \leq i \leq n$, and for $j \in \{1, 2, 3\}$, if $\nu_j(d_{i,j})$ and $\mu_{i,j}(c_{i,j})$ are both positive or both negative literals, we say that $d_{i,j}$ represents $c_{i,j}$; otherwise we say that $d_{i,j}$ represents $\neg c_{i,j}$. Of course, a variable in F_1 may be represented by several variables in F_2: for each variable a in F_1 let P_a be the set of variables of F_2 which represent a and N_a the set of variables which represent $\neg a$. Both sets are non-empty because of the clauses we added to F to obtain F_1 at the end of Step 1. Indeed, if a is a variable in F_1, then we added the clause $\nu_1(a) \vee \neg\nu_2(a) \vee \epsilon(a)$. If this clause is the ith clause of F_1, then the corresponding clause in F_2 is $\nu_1(d_{i,1}) \vee \nu_2(d_{i,2}) \vee \nu_3(d_{i,3})$, and the variable $d_{i,1}$ represents a and the variable $d_{i,2}$ represents $\neg a$.

There is no link between the satisfying assignments of F_1 and those of F_2. However we will define a coupling set such that the coupled satisfying assignment of F_2 are in bijection with the satisfying assignments of F_1 via a projection. For each variable a appearing in F_1, we consider the couplings shown on the graph in Fig. 1, where the top vertices are the variables in P_a, the bottom vertices are the variables in N_a and an edge means that the variables corresponding to the vertices are coupled. All the variables in P_a are therefore coupled to the first variable in N_a and all the variables in N_a are coupled to the last variable in P_a. This means that in a coupled assignment, all the variables in P_a will take the same value $\delta \in \Sigma$ and all the variables in N_a will take the same value $\neg\delta$. Call I the set of all the above couplings done for each variable a appearing in F_1. From now on when we say that an assignment is coupled, we will mean that it is coupled for the set I.

Let m be the number of variables appearing in F_1. We define a $(3n, m)$-projection α_2 which gives to a variable a in F_1 the value of any variable in F_2 which represents a. More formally, an assignment to the variables of F_1 can be encoded as a Boolean word of length m, where each bit in the word corresponds to the value of a variable in F_1, and similarly an assignment to the variables of F_2 can be encoded as a Boolean word of length $3n$. If the ith bit of an assignment to the variables of F_1 corresponds to the value of the variable a, let d be the first variable appearing in F_2 which represents a, and j the position of the bit giving the value of d in an encoding of an assignment of F_2. We then let $\alpha_2[i] = j$.

We will now show that the projection α_2 given above induces a bijection from the set of coupled satisfying assignments of F_2 to the set of satisfying assignments of F_1:

- *Claim: The image of a coupled satisfying assignment of F_2 is a satisfying assignment of F_1.* If $w \in \Sigma^{3n}$ is a satisfying assignment for F_2, then it satisfies each clause D_i, for $1 \leq i \leq n$. To satisfy the clause $D_i = \nu_1(d_{i,1}) \vee \nu_2(d_{i,2}) \vee \nu_3(d_{i,3})$, it must satisfy at least one of the literals, say $\nu_1(d_{i,1})$ (the reasoning is the same in the other cases). Consider the ith clause of F_1, $C_i = \mu_{i,1}(c_{i,1}) \vee \mu_{i,2}(c_{i,2}) \vee \mu_{i,3}(c_{i,3})$. If both literals are positive or both are negative, then the variable $d_{i,1}$ represents the variable $c_{i,1}$ and the value of the variable $c_{i,1}$ by the projection is the same as the value of the variable $d_{i,1}$. Because the literal $\nu_1(d_{i,1})$ is satisfied, the literal $\mu_{i,1}(c_{i,1})$ is also satisfied. If one of the literals is positive while the other is negative, then the variable $d_{i,1}$ represents the negation of the variable $c_{i,1}$, and the value of the variable $c_{i,1}$ by the projection is the

negation of the value of the variable $d_{i,1}$. The fact that the literal $\nu_1(d_{i,1})$ is satisfied still implies that the literal $\mu_{i,1}(c_{i,1})$ is satisfied. In any case, the clause C_i is satisfied. Repeating this argument for all i shows that the projection of a coupled satisfying assignment of F_2 is a satisfying assignment of F_1.

- *Claim: The projection α_2 induces a surjective application.* If u is a satisfying assignment for F_1, define an assignment for F_2 by giving to each variable in P_a the value given to a, and to each variable in N_a the negation of that value, doing this for each variable a appearing in F_1. The assignment thus defined is obviously coupled, and can be shown to satisfy F_2 by an argument similar to the one above.
- *Claim: The projection α_2 induces an injective application.* Consider two assignments u and w to the variables of F_2 such that $\alpha_2(u) = \alpha_2(v)$. If d is a variable in F_2, it belongs to P_a for a variable a in F_1. The value of d given by the assignment u is then the value of a given by the assignment $\alpha_2(u)$, which is the value of a given by the assignment $\alpha_2(v)$ and therefore equal to the value of d given by the assignment v.

Step 3. Let us now choose the smallest integer k such that $M_b^n M_j^{n-1} M_c^{3n} \cdot 2^{q^Y(y)} < 2^k$. As stated at the beginning of the proof, we will build n copies of the block instance, bring them together with the Join condition and then restrict the solutions to coupled solutions with the Couple condition. We describe these steps in succession.

(a) *Building the blocks.* We first build n copies $b_1, \ldots, b_n$ of the instance from the Block condition for the integer k, one instance for each clause D_i in F_2. Suppose that $t \in \Sigma^3$ is a solution for D_i, i.e. t is different from t_b, the word which defines each clause D_i. Then

$$\left|\{s \in S^X_{b_i} \mid t = \beta(s)\}\right| \equiv M_b \pmod{2^k}. \tag{1}$$

Otherwise (if t is not a solution for D_i, i.e. if $t = t_{b_i}$), then

$$\left|\{s \in S^X_{b_i} \mid t = \beta(s)\}\right| \equiv 0 \pmod{2^k}.$$

Indeed, this is just a restatement of the Block condition for b_i.

(b) *Joining the blocks.* We join all of the blocks b_i using the Join condition. Let us start by showing how it works for the blocks b_1 and b_2. Using the Join condition on instances b_1 and b_2, we get an instance x_1 and a projection ρ. We will use the projection $\pi_1 = \beta_1 \circ \rho$ with $\beta_1 = \beta \cdot \beta$. We wish to compute the cardinality of the following set:

$$\{s \in S^X_{x_1} \mid \pi_1(s) = t\} = \{s \in S^X_{x_1} \mid \beta_1 \circ \rho(s) = t\}.$$

If $t \in C^{3\mathrm{SAT}}_{D_1 \wedge D_2}$, then t can be seen as the concatenation of a word $t_1 \in C^{3\mathrm{SAT}}_{D_1}$ and a word $t_2 \in C^{3\mathrm{SAT}}_{D_2}$ — this is because D_1 and D_2 involve distinct variables. We partition the previous set as follows:

$$\{s \in S^X_{x_1} \mid \pi_1(s) = t_1 t_2\} = \bigcup_{\substack{s_1 \in C^X_{b_1},\ \beta(s_1) = t_1 \\ s_1 \in C^X_{b_2},\ \beta(s_2) = t_2}} \{s \in S^X_{x_1} \mid \rho(s) = s_1 s_2\}. \tag{2}$$

If s_1 is a solution of b_1 and s_2 is a solution of b_2, then the set under the union in Eq. (2) is congruent to M_j modulo 2^k, because of the Join condition. Otherwise, it is congruent to 0. The cardinality of the set $\{s \in S^X_{x_1} \mid \pi_1(s) = t\}$ is thus congruent to M_j multiplied by the following product of cardinals:

$$\left|\{s_1 \in S^X_{b_1} \mid \beta(s_1) = t_1\}\right| \cdot \left|\{s_2 \in S^X_{b_2} \mid \beta(s_2) = t_2\}\right|. \tag{3}$$

Suppose now that t is a satisfying assignment of the 3CNF-formula $D_1 \wedge D_2$. Then t is a concatenation of a satisfying assignment t_1 of D_1 and a satisfying assignment t_2 of D_2, and all satisfying assignments of $D_1 \wedge D_2$ are of this form. Eq. (1) shows that the product of cardinals in (3) is therefore congruent to M_b^2 modulo 2^k. If t is not a satisfying assignment of $D_1 \wedge D_2$, then either t_1 is not a satisfying assignment of D_1 or t_2 is not a satisfying assignment of D_2, and the product of cardinals in the expression (2) is therefore congruent to M_b^2 modulo 2^k. We have thus shown that for all $t \in C^{3\mathrm{SAT}}_{D_1 \wedge D_2}$, if $t \in S^{3\mathrm{SAT}}_{D_1 \wedge D_2}$, then the cardinality of the set $\{s \in S_{x_1} \mid \pi_1(s) = t\}$ is congruent to $M_b^2 M_j$ modulo 2^k and otherwise it is congruent to 0.

We bring the instances $b_1, \ldots, b_n$ together using the Join condition $n - 1$ times to get an instance x_2 and a projection π_2. Following the argument above, we can show by induction that this projection is such that if $t \in C^{3\mathrm{SAT}}_{F_2}$ is a satisfying assignment of F_2, then

$$\left|\{s \in S^X_{x_2} \mid \pi_2(s) = t\}\right| \equiv M_b^n M_j^{n-1} \pmod{2^k},$$

and otherwise (if t is not a satisfying assignment of F_2), then

$$\left|\{s \in S^X_{x_2} \mid \pi_2(s) = t\}\right| \equiv 0 \pmod{2^k}.$$

This means that we can relate the solutions of x_2 to the satisfying assignments of F_2 by a projection in a way which is consistent with the definition of universality. However, as we said when we were building the formula F_2, what we are interested in are the satisfying assignments of F_2 which are coupled according to the coupling set I defined in step 2.

(c) *Coupling.* We now use the Couple condition. Again, we will first observe the effect of applying it once. Let $1 \leq i < j \leq 3n$ denote the bit positions of two variables in F_2. We use the Couple condition on the instance x_2 for the integers $\pi_2[i]$ and $\pi_2[j]$ to get an instance x_3 and a projection γ (note that $\pi_2[i] \neq \pi_2[j]$ by definition of a projection, so that we can use the

Couple condition). Let $\pi_3 = \pi_2 \circ \gamma$. For a given $t \in C_{F_2}^{3SAT}$, we wish to evaluate the cardinality of the set $\{s \in S_{x_3}^X \mid \pi_3(s) = t\}$. We partition the set to express it as the following union:

$$\left(\bigcup_{\substack{u \in S_{x_2}^X \\ \pi_2(u)=t}} \{s \in S_{x_3}^X \mid \gamma(s) = u\} \right) \cup \left(\bigcup_{\substack{u \in C_{x_2}^X \setminus S_{x_2}^X \\ \pi_2(u)=t}} \{s \in S_{x_3}^X \mid \gamma(s) = u\} \right). \tag{4}$$

The cardinality of each set $\{s \in S_{x_3}^X \mid \gamma(s) = u\}$ in the second union is congruent to 0, by the Couple condition. We are therefore left with the following union:

$$\bigcup_{\substack{u \in S_{x_2}^X \\ \pi_2(u)=t}} \{s \in S_{x_3}^X \mid \gamma(s) = u\}. \tag{5}$$

Suppose that t is a satisfying assignment of F_2 which is coupled for i and j (i.e. it is an assignment which gives the value 1 to exactly one of the variables corresponding to i and j). Then since $t = \pi_2(u)$, the word u is $(\pi_2[i], \pi_2[j])$-coupled. In this case the cardinality of each set $\{s \in S_{x_3}^X \mid \gamma(s) = u\}$ above is congruent to M_c modulo 2^k, and the cardinality of the set $\{s \in S_{x_3}^X \mid \pi_3(s) = t\}$ is thus congruent to $M_b^n M_j^{n-1} M_c$ modulo 2^k by using the congruences at the end of the previous paragraph. Using a similar argument, one can show that if t is not coupled or if t is not a satisfying assignment of F_2, then the cardinality of the set $\{s \in S_{x_3}^X \mid \pi_3(s) = t\}$ is congruent to 0.

By applying the Couple condition for each of the couples in the set I described in step 2 of the proof, we get an instance x_4 and a projection π_4 such that if t is an I-coupled satisfying assignment of F_2, then:

$$\left|\{s \in S_{x_4} \mid \pi_4(s) = t\}\right| \equiv M_b^n M_j^{n-1} M_c^{|I|} \pmod{2^k}, \tag{6}$$

and otherwise:

$$\left|\{s \in S_{x_4} \mid \pi_4(s) = t\}\right| \equiv 0 \pmod{2^k}.$$

We can now bring everything together. Consider the $\left(q^X(x_4), q^Y(y)\right)$-projection $\pi = \alpha_1 \circ \alpha_2 \circ \pi_4$. We wish to compute the cardinality of the following set:

$$\begin{aligned} \{s \in S_{x_4}^X \mid \pi(s) = t\} &= \{s \in S_{x_4}^X \mid (\alpha_1 \circ \alpha_2) \circ \pi_4(s) = t\} \\ &= \bigcup_{\substack{u \in C_{F_2}^{3SAT} \\ (\alpha_1 \circ \alpha_2)(u)=t}} \{s \in S_{x_4}^X \mid \pi_4(s) = u\}. \end{aligned}$$

Suppose that $t \in S_y^Y$. Then since the projection $(\alpha_1 \circ \alpha_2)$ induces a bijection from the set of coupled solutions of F_2 to the set S_y^Y, t has a unique antecedent u_0 by $(\alpha_1 \circ \alpha_2)$ such that u_0 is a coupled satisfying assignment of F_2. The cardinality of the set $\{s \in S_{x_4}^X \mid \pi_4(s) = u_0\}$ is then congruent to $M_b^n M_j^{n-1} M_c^{|I|}$ modulo 2^k, by Eq. (6). All other antecedents u of t by $(\alpha_1 \circ \alpha_2)$ are not coupled satisfying assignments of F_2, and thus the cardinality of each set $\{s \in S_{x_4}^X \mid \pi_4(s) = u\}$ in the union is congruent to 0 modulo 2^k. Therefore, if t is a solution of y, the cardinality of the set $\{s \in S_{x_4}^X \mid \pi(s) = t\}$ is congruent to $M_b^n M_j^{n-1} M_c^{|I|}$. Otherwise, if $t \in C_y^Y \setminus S_y^Y$, then none of its antecedents by $(\alpha_1 \circ \alpha_2)$ are coupled assignments of F_2, and a similar argument shows that the cardinality of the set $\{s \in S_{x_4}^X \mid \pi(s) = t\}$ is congruent to 0.

We have thus defined an instance x_4, integers k and $M = M_b^n M_j^{n-1} M_c^{|I|}$ and a mapping π which satisfy the universality condition for X (the condition $M \cdot 2^{q^Y(y)} \leq 2^k$ holds true because of the choice of k and the fact that at most $3n$ couplings were needed). We must still check that the computation can be done in polynomial time. The integer k is polynomially bounded in the size of y because Y is a P-relation. We should also check the size of the resulting instance x_4. Thanks to the growth conditions in the three properties, its measure is bounded by $\left(\sum_{i=1}^n m(b_i)\right) + (n-1)k^{O(1)} + 3nk^{O(1)}$; of course $n = |y|^{O(1)}$. The measure of each instance b_i is also polynomially bounded in k. From all this, we get $m(x_4) = |y|^{O(1)}$ and thus $|x_4| = |y|^{O(1)}$. □

5. Examples

The first obvious example would be 3SAT: the necessary arguments have been given just after the criterion for universality. Further examples are given in this section. Monotone 2SAT is a logical problem and an example of a relation whose decision problem is in P but whose counting problem is #P-complete. Maximal independent set and Hamiltonian cycles are graph problems; the proofs that they satisfy the criterion are very short and boil down to finding the right graph gadgets. Cycle covers illustrate how the classical proof of completeness for the permanent of 0–1 matrices fits perfectly in our framework. Knapsack is an application of the criterion in yet another setting. We also include the cases of counting maximal cliques, minimal vertex covers and s–t-paths.

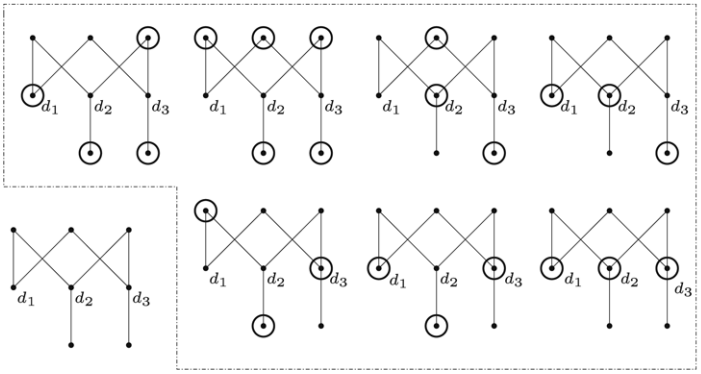

Fig. 2. Block for maximal independent set and its solutions.

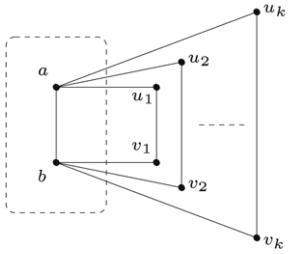

Fig. 3. Couple for maximal independent set.

5.1. Monotone 2SAT

Monotone 2SAT is similar to 3SAT, but it demands a little more work. Monotone 2SAT instances are 2CNF formulas without negative literals. Let $k_0 = 1$.

Block. For $k \geqslant k_0$, the formula corresponding to the block (modulo 2^k) is the following one: $B = \bigwedge_{i=1}^{k}(d_1 \vee u_i) \wedge (d_2 \vee u_i) \wedge (d_3 \vee u_i)$. If at least one of the variables d_1, d_2 or d_3 is false, then all the u_i must be true for an assignment to satisfy B. On the other hand, if all three d_i are true, then there are 2^k ways to satisfy B. This corresponds to the definition of Block with $t_b = 111$.

Join. Given two formulas F and G, we first rename the variables to ensure they have no variables in common. The join of F and G is then the formula $F \wedge G$.

Couple. Coupling x_a and x_b in the formula $F(\bar{x})$ (modulo 2^k) is done by the formula $F(\bar{x}) \wedge (x_a \vee x_b) \wedge \bigwedge_{i=1}^{k}(x_a \vee u_i) \wedge (x_b \vee u_i)$. This ensures first that x_a and x_b cannot both be false. If both are true there are 2^k possible assignments for the variables u_i.

5.2. Maximal independent set

We focus now on a graph problem. The definition of measure has been given in the introduction. Let $k_0 = 2$.

Block. $M_b = 1$. For any $k \geqslant k_0$, consider the graph G of Fig. 2. This figure also shows all maximal independent subsets of G: none of them contain d_2 and d_3 but not d_1, but all other subsets of $\{d_1, d_2, d_3\}$ correspond to exactly one solution.

Join. Suppose we have an integer $k \geqslant k_0$ and two disjoint graphs G_1 and G_2. Consider the graph G which is the union of G_1 and G_2. There is a bijection between the solution set of G and the Cartesian product of the solution sets of G_1 and G_2, so that the required cardinality condition holds for $M_j = 1$. The growth condition is obvious.

Couple. Suppose we have an integer $k \geqslant k_0$, a graph G, and two vertices a and b appearing in G. Consider the graph G' obtained from G in the following manner. We first add the edge (a, b) if it is not already present. Then we add $2k$ vertices $\{u_1, \ldots, u_k\}$ and $\{v_1, \ldots, v_k\}$. At last, we add the $3k$ edges (a, u_i), (u_i, v_i) and (v_i, b) for $1 \leqslant i \leqslant k$, as shown on Fig. 3.

The set of solutions for G' can be partitioned in the following way. Any solution for G which contains neither a nor b yields exactly 2^k solutions of G', because any maximal subset of G' which contains neither a nor b must contain exactly one vertex in each pair (u_i, v_i). Any solution for G which contains both a and b cannot be extended to a maximal independent set for G' because a and b are linked by an edge in G'. Any solution for G which contains a and not b can be extended into a unique maximal independent set for G' by adding all the vertices v_i. The symmetrical situation for a and b is similar. Therefore if a solution for G is not coupled with regard to vertices a and b, then it yields either 0 or a multiple 2^k solutions of G'. A coupled solution for G yields exactly one solution for G' . Therefore $M_c = 1$. As for the growth condition, we have added $2k$ vertices to G, so that the measure of G' is bounded by the measure of G plus k^2 for $k \geqslant k_0 = 2$.

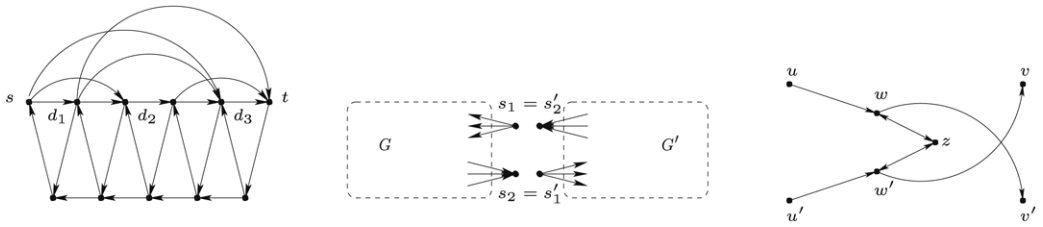

Fig. 4. Block, Join and XOR for Hamiltonian cycles.

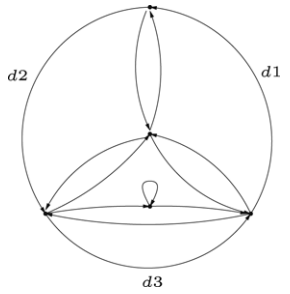

Fig. 5. Block for cycle covers.

5.3. Hamiltonian cycles

An instance for this problem is a graph and a solution is a subset of the edges which is a Hamiltonian cycle. Fig. 4 gives the block, the Join construction and a XOR gadget which we will use to couple.

Block. One should check that there is exactly one Hamiltonian cycle for each non-empty subset of $\{d_1, d_2, d_3\}$, and no Hamiltonian cycle avoiding these three edges.

Join. Suppose we now wish to join two instances G and G'. We choose a vertex s in G and split into two vertices s_1 and s_2. All the outgoing edges of s become outgoing edges of s_1 and all the incoming edges of s become incoming edges of s_2, so that there is now a one-to-one correspondence between Hamiltonian cycles of G and Hamiltonian paths from s_1 to s_2 in the new graph. We modify G' in the same way, splitting a node s' into s'_1 and s'_2. We then identify s_2 and s'_1, and s'_2 and s_1. Any Hamiltonian cycle of this graph is made of a Hamiltonian path of G from s_1 to s_2 and a Hamiltonian path of G' from s'_1 to s'_2.

Couple. If we wish to couple edges (u, v) and (u', v') in a graph G, we start by deleting these edges and connect the vertices u, v, u' and v' with the XOR gadget. This gadget is such that the Hamiltonian cycles of the resulting graph must contain one of the edges (u, w) or (u', w') but not both.

5.4. Cycle covers (or bipartite matchings, or the permanent of 0–1 matrices)

An instance for this problem is a graph and a solution is a subset of the edges which is a cycle cover. We shall explain how the proof of #P-completeness of the permanent given in [7] fits in a straightforward manner in our framework. Let $k_0 = 1$.

Block. Fig. 5 gives the block. One should check that there is exactly one cycle cover taking all edges of each strict subset of $\{d_1, d_2, d_3\}$, and no cycle cover using these three edges.

Join. Let G_1 and G_2 two instances of the cycle cover problem. It is easily checked that the disjoint union G of the two graphs realizes the join of G_1 and G_2. Indeed, any cycle cover of G is made of a cycle cover of G_1 and a cycle cover of G_2.

Couple. Let $k \geqslant k_0$. Coupling two edges (u, v) and (u', v') in a graph G is achieved by plugging the XOR gadget as described in [7]. As in the original proof, the edges with weight 2 or 3 are replaced with sequence of edges (and self-loop on the the intermediate edges), while the edge with weight -1 is now replaced with a subgraph simulating an edge of weight $2^k - 1$. This is done with $k^{O(1)}$ edges. Note also that both edges (u, v) and (u', v') disappear when they are coupled, but one can find equivalent edges on the XOR gadget.

5.5. Knapsack

Here is an example from a different setting. An instance of Knapsack is given by a set of integer weights $c_1, \ldots, c_n$ and an integer b called the sum, where the c_i and b are all strictly positive. A solution is a subset s of $\{1, \ldots, n\}$ such that $\sum_{i \in s} c_i = b$. The measure of an instance will be given by two integers: n (the number of integers c_i) and the bit-size of $b + \sum_{i \in \{1,\ldots,n\}} c_i$.

Block. Consider the instance with weights 1, 1, 1, 1, 2 and sum 4. Solutions for this instance cannot omit the first three weights. Moreover, for any non-empty subset of these three weights, there is only one way to complement the sum to 4.

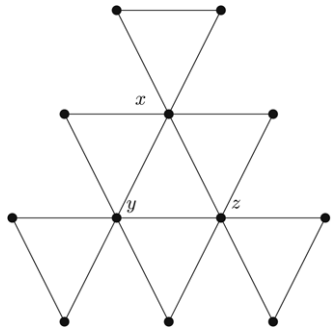

Fig. 6. Block for Max clique.

Join. Let us take two instances $a_1, \ldots, a_j, a$ and $b_1, \ldots, b_k, b$. Call S and T respectively the sums $a + \sum_{i=1}^{j} a_i$ and $b + \sum_{i=1}^{k} b_i$. Let our new instance be composed of the weights $a_1, \ldots, a_j, Sb_1, \ldots, Sb_k$ and sum $a + Sb$. The first integer in the measure of this new instance is $j + k$. The second is the bit-size of the following integer:

$$b + \sum_{i=1}^{j} a_i + S \cdot \left(\sum_{i=1}^{k} b_i \right) + S \cdot b = S(T + 1).$$

The measure can thus be bounded by the sum of the measures of the two initial instances plus a constant. It is easy to see that there is a bijection between couples of solutions for the initial instances and solutions for the new one. Indeed, suppose we have a subset J of $\{1, \ldots, j\}$ and a subset K of $\{1, \ldots, k\}$ such that $\sum_{i\in J} a_i + S \cdot \left(\sum_{i\in K} b_i\right) = a + Sb$, then we have the following equation: $\sum_{i\in J} a_i - a = S \cdot \left(b - \sum_{i\in K} b_i\right)$. The absolute value of the left-hand side is strictly smaller than S, while on the right- hand side it is either 0 or greater than S. Therefore it must be 0 on both sides, and J and K yield solutions for the initial instances.

Couple. Suppose we have an instance $a_1, \ldots, a_k, a$ and we wish to couple a_i and a_j. Let S be the sum $a + \sum_{l=1}^{k} a_l$. We consider the new instance obtained by replacing the weight a_i is with $a_i + S$, the weight a_j with $a_j + S$ and the sum a with $a + S$. Any solution for this new instance cannot omit or include both $a_i + S$ and $a_j + S$. Now suppose we have a solution which includes only one of them, for instance we have a subset K of $\{1, \ldots, k\} \setminus \{i\}$ such that $a_i + S + \sum_{l\in K} a_l = a + S$. Then $a_i + \sum_{l\in K} a_l = a$ and we get a solution for the initial instance. The measure of this new instance is composed of k and the bit-size of $\sum_{l=1}^{k} a_l + a + 3S = 4S$, and therefore can be bounded by the initial measure plus a constant.

5.6. Maximal clique

The proof that *maximal clique* is #P-universal follows a similar pattern as the previous one. This relation is $R(G, S)$, where G is a graph and S is a subset of the vertices which is a maximal clique in G.

Block. We take the blocks already appearing in [5] (cf. Fig. 6). Any vertex different from x, y and z is connected to at least one of x, y and z. Thus a clique cannot avoid these three vertices at once. Using the appropriate triangle, one can find maximal cliques containing one, two or three vertices from the set $\{x, y, z\}$.

Join. The join of two graphs is obtained by taking their disjoint union.

Couple. To couple vertices a and b in the graph $G = (V, E)$ modulo 2^k, we proceed as follows. First we remove the edge (a, b) if there is one. Then we add two sets of vertices $\{u_1, \ldots, u_k\}$ and $\{v_1, \ldots, v_k\}$. Each vertex u_i is connected to all other vertices in the graph except b and v_i. Each vertex v_i is connected to all other vertices in the graph except a and u_i. Because there is no edge between a and b, both vertices cannot be present in a clique of the resulting graph. If a maximal clique of G contains neither a nor b, then we can and must add one of u_i or v_i but not both, for all i, which yields 2^k maximal cliques of the resulting graphs. If a maximal clique of G contains a and not b, then we can and must add all the vertices u_i, which are connected to a, but cannot add any of the v_i because they are not connected to a.

5.7. Minimal vertex cover

We now show that the canonical relation associated to *Minimal Vertex Cover* is #P-universal. This relation is $R(G, S)$, where G is a graph and S is a subset of the vertices which is a minimal vertex cover for G.

Block. For the blocks, we take the graph shown in Fig. 7: it corresponds to the clause $\neg x \vee \neg y \vee \neg z$. If a cover contains vertices x, y and z, then we have 2^k choices because for each i we must take either u_i or v_i. For any other subset of $\{x, y, z\}$ there is only one minimal vertex cover.

Join. Joining two graphs is performed by taking their disjoint union.

Couple. To couple the vertices a and b in the graph G modulo 2^k, we modify the graph as in the case of maximal independent set (cf. Fig. 3). Any minimal vertex cover must contain at least one of a or b. As in the case of maximal

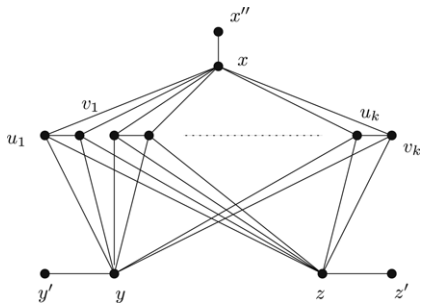

Fig. 7. Block for minimal vertex cover.

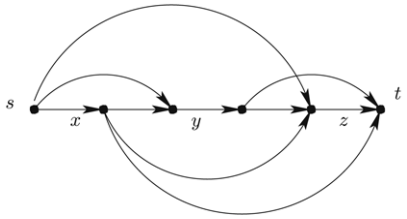

Fig. 8. Block for s–t-paths.

independent set, a minimal vertex cover of the original graph containing exactly one of a or b can be extended in a unique way, whereas a minimal vertex cover containing both has 2^k extensions.

5.8. Paths from s to t

We wish to show that the canonical relation associated with s–t-paths is #P-universal. This relation is $R(G, E)$, where G is a graph and E is a subset of the edges which is a path from s to t in G.

Block. The block is the same as the one given in [5] and is shown in Fig. 8. One should check that there are no s–t-paths which avoid edges x, y and z, and that for each non-empty subset S of these edges there is exactly one s–t-path going through edges of S and not going through edges of $\{x, y, z\} \setminus S$.

Join. To join a graph G_1 with vertices s_1 and t_1 with a graph G_2 with vertices s_2 and t_2 we just identify t_1 and s_2 and consider the resulting graph G with $s = s_1$ and $t = t_2$.

Couple. To couple we will first need an iff gadget, as presented in Fig. 9, in which all edges have weight 1 except for the edge of weight -1. The weight of a path is the product of the weights of its edges. Suppose G is a graph with edges u and v, and that we delete edges u and v and replace them with the iff gadget. The sum of the weights of the s–t-paths in the resulting graph is the number of s–t-paths of G which either went through both edges u and v or through none. Indeed, one can first notice that if a path in the new graph goes from α to β' through edge a, there is an equivalent path which will go through edge b and have weight -1, so that the two cancel out: the new paths introduced by the gadget do not count. Now if a path in the original graph went through u but not through v, it yields two paths in the new graph, one going through a and the other going through b, thus canceling each other out. Therefore only paths from the original graph which went through both u and v or neither of them are counted.

But we now have a graph with weights. We will simulate -1 by counting modulo 2^k, thus replacing an edge with -1 by the graph from Fig. 11, which has exactly $2^k - 1 = 1 + 2 + \cdots + 2^{k-1}$ paths from α to β. We now come back to our instance G, where we wish to couple the two edges u and v. We start by adding two new vertices t' and t'', with the edges (t, t'), (t, t'') and (t'', t') a shown in Fig. 10. A path from s to t' in the resulting graph must go either trough (t, t') or through (t'', t') exactly. We then use our iff gadget between edges u and (t'', t') and between edges v and (t, t'). The number of vertices added to the graph is of the order of k^2.

6. Conclusion

There are two ways to see the work done in this paper. On a theoretical level, it argues for the existence of structural similarities between difficult problems, an idea which has been studied for NP and which in here is applied to #P. As such it is an attempt to better understand why some problems are easy and some are difficult. Our work shows basically two ingredients for a relation to yield a #P-complete problem. One is a kind of building block/inductive structure, already noticed by Agrawal and Biswas in the case of NP, with stricter conditions on solution sets in order to adapt it to #P. The other is the possibility to compute modulo a big integer, as is crucial for instance in the proof of the completeness of the Permanent. If we consider the first ingredient, finding the closest possible (w.r.t. NP) structural criteria for #P-universality is a good way to study the famous question of whether all relations which yield NP-complete decision problems also yield #P-complete

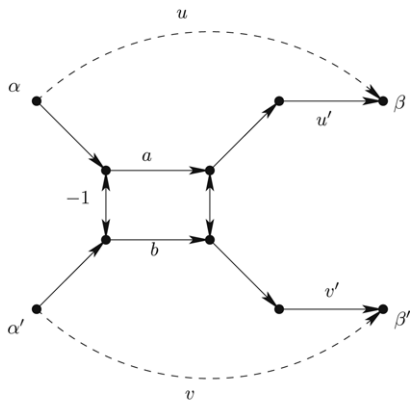

Fig. 9. Iff gadget for s–t-paths.

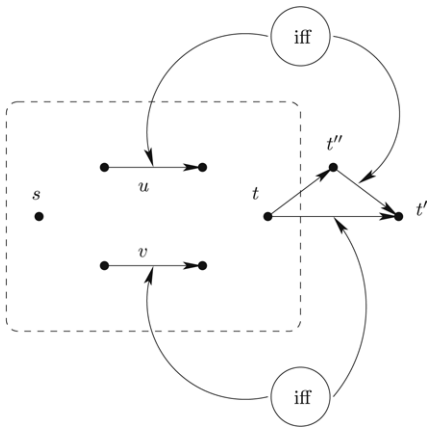

Fig. 10. Couple for s–t paths.

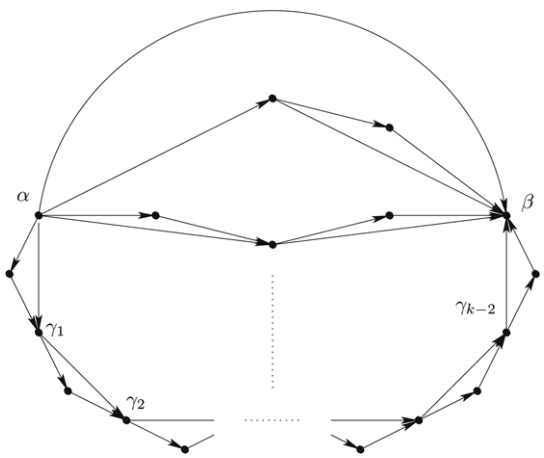

Fig. 11. $2^k - 1$ paths.

counting problems. The second ingredients is a rough explanation of why some easy decision problems have #P-complete counting equivalents. The interplay between these two ingredients makes the class #P rich and interesting.

The other focus of this work is to give a useful criterion for #P-completeness. When one wishes to prove that a problem is #P-hard, one often tries to find a known #P-hard problem which seems "near" enough, so that the reduction will be easier to exhibit. Our criterion takes advantage of the argument from the previous paragraph, namely the existence of common structure, to eliminate the search for a suitable known #P-hard problem. In other words the universality criterion plays the role of a generic #P-hard problem, but one which should be "close" enough in most cases, because the distance is bridged by Theorem 1. There may well be a #P-hard problem more suitable for a given example, i.e. yielding a simpler reduction,

but we believe that the universality criterion corresponds to a large class of natural problems, as hinted at by the variety of examples, for which proofs can be built in a systematic way.

References

[1] Manindra Agrawal, Somenath Biswas, Universal relations, in: Structure in Complexity Theory Conference, 1992, pp. 207–220.
[2] J.L. Balcazar, J. Diaz, J. Gabarro, Structural Complexity 1, Springer-Verlag New York, Inc., New York, NY, USA, 1988.
[3] Leonard Berman, Juris Hartmanis, On isomorphisms and density of NP and other complete sets, SIAM J. Comput. 6 (2) (1977) 305–322.
[4] Harry Buhrman, Jim Kadin, Thomas Thierauf, Functions computable with nonadaptive queries to NP, Theory Comput. Syst. 31 (1) (1998) 77–92.
[5] G. Chakravorty, R. Kumar, #P universality, Technical report, Indian Institute of Technology, Kanpur, 2003.
[6] V. Chaudhary, A.K. Sinha, S. Biswas, Universality for Nondeterministic Logspace, presented at Indo-German Workshop on Algorithms, Bangalore, 2004.
[7] Christos H. Papadimitriou, Computational Complexity, Addison-Wesley Publishing Company, Reading, MA, 1994.
[8] Bruno Poizat, Les Petits Cailloux, in: Nur Al-Mantiq Wal-Ma'rifah, vol. 3, Aléas, Lyon, 1995.
[9] Natacha Portier, Résolutions universelles pour des problèmes NP-complets, Theor. Comput. Sci. 201 (1–2) (1998) 137–150.
[10] Seinosuke Toda, PP is as hard as the polynomial-time hierarchy, SIAM J. Comput. 20 (5) (1991) 865–877.
[11] Leslie G. Valiant, The complexity of enumeration and reliability problems, SIAM J. Comput. 8 (3) (1979) 410–421.