

# Optimization, Randomized Approximability, and Boolean Constraint Satisfaction Problems

Tomoyuki Yamakami\*

**Abstract.** We give a unified treatment to optimization problems that can be expressed in the form of nonnegative-real-weighted Boolean constraint satisfaction problems. Creignou, Khanna, Sudan, Trevisan, and Williamson studied the complexity of approximating their optimal solutions whose optimality is measured by the sums of outcomes of constraints. To explore a wider range of optimization constraint satisfaction problems, following an early work of Marchetti-Spaccamela and Romano, we study the case where the optimality is measured by products of constraints' outcomes. We completely classify those problems into three categories: PO problems, NPO-hard problems, and intermediate problems that lie between the former two categories. To prove this trichotomy theorem, we analyze characteristics of nonnegative-real-weighted constraints using a variant of the notion of T-constructibility developed earlier for complex-weighted counting constraint satisfaction problems.

**keywords:** optimization problem, approximation algorithm, constraint satisfaction problem, PO, APX, approximation-preserving reducibility

## 1 Maximization by Multiplicative Measure

In the 1980s started extensive studies that have greatly improved our understandings of the exotic behaviors of various optimization problems within a scope of computational complexity theory. These studies have brought us deep insights into the approximability and inapproximability of optimization problems; however, many studies have targeted individual problems by cultivating different and independent methods for them. To push our insights deeper, we are focused on a collection of “unified” optimization problems, whose foundations are all formed in terms of *Boolean constraint satisfaction problems* (or *CSPs*, in short). Creignou is the first to have given a formal treatment to maximization problems derived from CSPs [3]. The *maximization constraint satisfaction problems* (or MAX-CSPs for succinctness) are, in general, optimization problems in which we seek a truth assignment  $\sigma$  of Boolean variables that maximizes an *objective value*<sup>†</sup> (or a *measure*) of  $\sigma$ , which equals the number of constraints being satisfied at once. Creignou presented three criteria (which are 0-validity, 1-validity, and 2-monotonicity) under which we can solve the MAX-CSPs in polynomial time; that is, the problems belong to PO.

Creignou's result was later reinforced by Khanna, Sudan, Trevisan, and Williamson [7], who gave a unified treatment to several types of CSP-based optimization problems, including MAX-CSP, MIN-CSP, and MAX-ONE-CSP. With constraints limited to “nonnegative” integer values, Khanna et al. defined MAX-CSP( $\mathcal{F}$ ) as the maximization problem in which constraints are all taken from constraint set  $\mathcal{F}$  and the maximization is measured by the “sum” of the objective values of

---

\*Present Affiliation: Department of Information Science, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan

<sup>†</sup>A function that associates an objective value (or a measure) to each solution is called an *objective function* or (a *measure function*).

constraints. For a later comparison, we call such a measure an *additive measure*. More formally,  $\text{MAX-CSP}(\mathcal{F})$  is defined as:

$\text{MAX-CSP}(\mathcal{F})$ :

- **INSTANCE:** a *finite* set  $H$  of elements of the form  $\langle h, (x_{i_1}, \dots, x_{i_k}) \rangle$  on Boolean variables  $x_1, \dots, x_n$ , where  $h \in \mathcal{F}$ ,  $\{i_1, \dots, i_k\} \subseteq [n]$ , and  $k$  is the arity of  $h$ .
- **SOLUTION:** a truth assignment  $\sigma$  to the variables  $x_1, \dots, x_n$ .
- **MEASURE:** the sum  $\sum_{\langle h, x' \rangle \in H} h(\sigma(x_{i_1}), \dots, \sigma(x_{i_k}))$ , where  $x' = (x_{i_1}, \dots, x_{i_k})$ .

For instance,  $\text{MAX-CSP}(\text{XOR})$  coincides with the optimization problem MAX-CUT, which is known to be MAX-SNP-complete [9]. Khanna et al. re-proved Creignou’s classification theorem that, for every set  $\mathcal{F}$  of constraints, if  $\mathcal{F}$  is one of 0-valid, 1-valid, and 2-monotone, then  $\text{MAX-CSP}(\mathcal{F})$  is in PO, and otherwise,  $\text{MAX-CSP}(\mathcal{F})$  is APX-complete<sup>‡</sup> under their approximation-preserving reducibility. This classification theorem was proven by an extensive use of a notion of *strict/perfect implementation*.

From a different and meaningful perspective, Marchetti-Spaccamela and Romano [8] made a general discussion on max NPSP problems whose maximization is measured by the “products” of the objective values of chosen (feasible) solutions. From their general theorem follows an early result of [1] that the maximization problem, MAX-PROD-KNAPSACK, has a *fully polynomial(-time) approximation scheme* (or FPTAS). This result can be compared with another result of Ibarra and Kim [5] that MAX-KNAPSACK (with additive measures) admits an FPTAS. The general theorem of [8] requires development of a new proof technique, called *variable partitioning*. A similar approach was taken in a study of linear multiplicative programming to minimize the “product” of two positive linear cost functions, subject to linear constraints [6]. In contrast with the previous additive measures, we call this different type of measures *multiplicative measures*, and we wish to study the behaviors of MAX-CSPs whose maximization is taken over multiplicative measures.

Our approach clearly fills a part of what Creignou [3] and Khanna et al. [7] left unexplored. Let us formalize our objectives for clarity. To differentiate our multiplicative measures from additive measures, we develop a new notation  $\text{MAX-PROD-CSP}(\cdot)$ , in which “PROD” refers to a use of “product” of objective values.

$\text{MAX-PROD-CSP}(\mathcal{F})$ :

- **INSTANCE:** a *finite* set  $H$  of elements of the form  $\langle h, (x_{i_1}, \dots, x_{i_k}) \rangle$  on Boolean variables  $x_1, \dots, x_n$ , where  $h \in \mathcal{F}$  and  $\{i_1, \dots, i_k\} \subseteq [n]$ .
- **SOLUTION:** a truth assignment  $\sigma$  to  $x_1, \dots, x_n$ .
- **MEASURE:** the product  $\prod_{\langle h, x' \rangle \in H} h(\sigma(x_{i_1}), \dots, \sigma(x_{i_k}))$ , where  $x' = (x_{i_1}, \dots, x_{i_k})$ .

There is a natural, straightforward way to relate  $\text{MAX-CSP}(\mathcal{F})$ ’s to  $\text{MAX-PROD-CSP}(\mathcal{F})$ ’s. For example, consider the problem MAX-CUT and its multiplicatively-measured counterpart, MAX-PROD-CUT. Given any cut for MAX-CUT, we set the weight of each vertex to be 2 if it belongs to the cut, and set the weight to be 1 otherwise. When the cardinality of a cut is maximized, the same cut incurs the maximum product value as well. In other words, an algorithm that “exactly” finds an optimal solution for MAX-CUT also computes an optimal solution for MAX-PROD-CUT. However, when an algorithm only tries to “approximate” such an optimal solution, its performance rates for MAX-CUT and for MAX-PROD-CUT significantly differ. In a case of

---

<sup>‡</sup>APX is the collection of optimization problems whose optimal solutions can be (deterministically) approximated to within a fixed constant in polynomial time.

geometric programming, a certain type of product objective function is known to be “reduced” to additive ones if function values are all positive (see an survey [2]). In our setting, a different complication comes in when some values of  $h$ ’s accidentally fall into zero and thus the entire product value vanishes. Thus, an approximation algorithm using an additive measure does not seem to lead to an approximation algorithm with a multiplicative measure. This circumstance indicates that multiplicative measures appear to endow their associated optimization problems with much higher approximation complexity than additive measures do. We then need to develop a quite different methodology for a study on the approximation complexity of multiplicatively-measured optimization problems.

Within the framework of MAX-PROD-CSPs, we can precisely express many maximization problems, such as MAX-PROD-CUT, MAX-PROD-SAT, MAX-PROD-IS (independent set), and MAX-PROD-BIS (bipartite independent set), which are naturally induced from their corresponding additively-measured maximization problems. Some of their formal definitions will be given in Section 2.2. In a context of approximability, since the multiplicative measures can provide a richer structure than the additive measures, classification theorems for MAX-CSPs and for MAX-PROD-CSPs are essentially different. The classification theorem for MAX-PROD-CSPs—our main result—is formally stated in Theorem 1.1, in which we use an abbreviation, MAX-PROD-CSP<sup>\*</sup>( $\mathcal{F}$ ), to mean a MAX-PROD-CSP whose unary constraints are provided for free<sup>§</sup> and other nonnegative-real-weighted constraints are drawn from  $\mathcal{F}$ .

**Theorem 1.1** *Let  $\mathcal{F}$  be any set of constraints. If either  $\mathcal{F} \subseteq \mathcal{AF}$  or  $\mathcal{F} \subseteq \mathcal{ED}$ , then MAX-PROD-CSP<sup>\*</sup>( $\mathcal{F}$ ) is in PO. Otherwise, if  $\mathcal{F} \subseteq \mathcal{IM}_{opt}$ , then MAX-PROD-CSP<sup>\*</sup>( $\mathcal{F}$ ) is APT-reduced from MAX-PROD-BIS and is APT-reduced to MAX-PROD-FLOW. Otherwise, MAX-PROD-CSP<sup>\*</sup>( $\mathcal{F}$ ) is APT-reduced from MAX-PROD-IS.*

Here, “APT” stands for “approximation preserving Turing” in the sence of [4]. Moreover,  $\mathcal{AF}$  is the set of “affine”-like constraints,  $\mathcal{ED}$  is related to the binary equality and disequality constraints, and  $\mathcal{IM}_{opt}$  is characterized by “implication”-like constraints. The problem MAX-PROD-FLOW is a maximization problem of finding a design that maximizes the amount of water flow in a given direct graph. See Sections 2.1–2.2 for their formal definitions.

The purpose of the rest of this paper is to prove Theorem 1.1. For this purpose, we will introduce a new notion of  $T_{max}$ -constructibility, which is a variant of the notion of T-constructibility invented in [10]. This  $T_{max}$ -constructibility is proven to be a powerful tool in dealing with MAX-PROD-CSPs.

## 2 Formal Definitions and Basic Properties

Let  $\mathbb{N}$  denote the set of all *natural numbers* (i.e., nonnegative integers) and let  $\mathbb{R}$  denote the set of all real numbers. For convenience,  $\mathbb{N}^+$  expresses  $\mathbb{N} - \{0\}$  and, for each  $n \in \mathbb{N}^+$ ,  $[n]$  denotes the integer set  $\{1, 2, \dots, n\}$ . Moreover, the notation  $\mathbb{R}^{\geq 0}$  stands for the set  $\{r \in \mathbb{R} \mid r \geq 0\}$ .

### 2.1 Constraints and Relations

Here, we use terminology given in [10]. A (*nonnegative-real-weighted*) *constraint* is a function mapping from  $\{0, 1\}^k$  to  $\mathbb{R}^{\geq 0}$ , where  $k$  is the *arity* of  $f$ . Assuming the standard lexicographic

---

<sup>§</sup>Allowing a free use of arbitrary unary constraints is a commonly used assumption for decision CSPs and counting CSPs.

order on  $\{0,1\}^k$ , we express  $f$  as a series of its output values. For instance, if  $k = 2$ , then  $f$  is  $(f(00), f(01), f(10), f(11))$ . We set  $EQ = (1, 0, 0, 1)$ ,  $\Delta_0 = (1, 0)$ , and  $\Delta_1 = (0, 1)$ .

A *relation* of arity  $k$  is a subset of  $\{0,1\}^k$ . Such a relation can be also viewed as a function mapping Boolean variables to  $\{0,1\}$  (i.e.,  $x \in R$  iff  $R(x) = 1$ , for every  $x \in \{0,1\}^k$ ) and it can be treated as a Boolean constraint. For instance, logical relations  $OR$ ,  $NAND$ ,  $XOR$ , and  $Implies$  are all expressed as appropriate constraints in the following manner:  $OR = (0, 1, 1, 1)$ ,  $NAND = (1, 1, 1, 0)$ ,  $XOR = (0, 1, 1, 0)$ , and  $Implies = (1, 1, 0, 1)$ . A relation  $R$  is *affine* if it is expressed as a set of solutions to a certain system of linear equations over  $GF(2)$ . An *underlying relation*  $R_f$  of  $f$  is defined as  $R_f = \{x \mid f(x) \neq 0\}$ .

We introduce the following six special sets of constraints.

1. Let  $\mathcal{U}$  denote the set of all unary constraints.
2. The notation  $\mathcal{NZ}$  expresses the set of all non-zero constraints.
3. Denote by  $\mathcal{DG}$  the set of all constraints that are expressed by products of unary constraints, each of which is applied to a different variable. Such a constraint is called *degenerate*.
4. Let  $\mathcal{ED}$  denote the set of all constraints that are expressed as products of some of unary constraints, the equality  $EQ$ , and the disequality  $XOR$ .
5. The set  $\mathcal{AF}$  is defined as the collection of all constraints of the form  $g(x_1, \dots, x_k) \prod_{j:j \neq i} R_j(x_i, x_j)$  for a certain fixed index  $i \in [k]$ , where  $g$  is in  $\mathcal{DG}$  and each  $R_j$  is an affine relation.
6. Define  $\mathcal{IM}_{opt}$  to be the collection of all constraints that are products of some of the following constraints: unary constraints and constraints of the form  $(1, 1, \lambda, 1)$  with  $0 \leq \lambda < 1$ . This is different from  $\mathcal{IM}$  defined in [10].

**Lemma 2.1** *For any constraint  $f = (x, y, z, w)$  with  $x, y, z, w \in \mathbb{R}^{\geq 0}$ , if  $xw > yz$ , then  $f$  belongs to  $\mathcal{IM}_{opt}$ .*

## 2.2 Optimization Problems with Multiplicative Measures

A (*combinatorial*) *optimization problem*  $P = (I, \text{sol}, m)$  takes input instances of “admissible data” to the target problem. We often write  $I$  to denote the set of all such instances and  $\text{sol}(x)$  denotes a set of (*feasible*) *solutions* associated with instance  $x$ . A *measure function* (or *objective function*)  $m$  associates a nonnegative real number to each solution  $y$  in  $\text{sol}(x)$ ; that is,  $m(x, y)$  is an *objective value* (or a *measure*) of the solution  $y$  on the instance  $x$ . We conveniently assume  $m(x, y) = 0$  for any element  $y \notin \text{sol}(x)$ . The goal of the problem  $P$  is to find a solution  $y$  in  $\text{sol}(x)$  that has an optimum value (such  $y$  is called an *optimal solution*), where the optimality is measured by either the maximization or minimization of an objective value  $m(x, y)$ , taken over all solutions  $y \in \text{sol}(x)$ . When  $y$  is an optimal solution, we set  $m^*(x)$  to be  $m(x, y)$ .

Let NPO denote the class of all optimization problems  $P$  such that (1) input instances and solutions can be recognized in polynomial time; (2) solutions are polynomially-bounded in input size; and (3) a measure function can be computed in polynomial time. Define PO as the class of all problems  $P$  in NPO such that there exists a deterministic algorithm that, for every instance  $x \in I$ , returns an optimal solution  $y$  in  $\text{sol}(x)$  in time polynomial in the size  $|x|$  of the instance  $x$ .

We say that, for a fixed real-valued function  $\alpha$  with  $\alpha(n) \geq 1$  for any input size  $n \in \mathbb{N}$ , an algorithm  $\mathcal{A}$  for an optimization problem  $P = (I, \text{sol}, m)$  is an  $\alpha$ -*approximation algorithm* if, for every instance  $x \in I$ ,  $\mathcal{A}$  produces a solution  $y \in \text{sol}(x)$  satisfying that  $1/\alpha(|x|) \leq |m(x, y)/m^*(x)| \leq \alpha(|x|)$ , except that, whenever  $m^*(x) = 0$ , we always demand that  $m(x, y) = 0$ . Such a  $y$  is called

an  $\alpha(n)$ -approximate solution for input instance  $x$  of size  $n$ . The class APX (resp., exp-APX) consists of all problems  $P$  in NPO such that there are a constant  $r \geq 1$  (resp., an exponentially-bounded<sup>¶</sup> function  $\alpha$ ) and a polynomial-time  $r$ -approximation (resp.,  $\alpha$ -approximation) algorithm for  $P$ . Approximation algorithms are often randomized. A *randomized approximation scheme* for  $P$  is a randomized algorithm that takes a standard input instance  $x \in I$  together with an error tolerance parameter  $\varepsilon \in (0, 1)$ , and outputs a  $2^\varepsilon$ -approximate solution  $y \in \text{sol}(x)$  with probability at least  $3/4$ .

Of numerous existing notions of approximation-preserving reducibilities, we choose a notion introduced recently by Dyer, Goldberg, Greenhill, and Jerrum [4], which can be viewed as a randomized variant of Turing reducibility, based on a mechanism of *oracle Turing machine*. Since the purpose of Dyer et al. is to solve counting problems approximately, we need to modify their notion so that we can deal with optimization problems. Given two optimization problems  $P = (I, \text{sol}, m)$  and  $Q = (I', \text{sol}', m')$ , a *polynomial-time (randomized) approximation-preserving Turing reduction* (or *APT-reduction*, in short) from  $P$  to  $Q$  is a randomized algorithm  $N$  that takes a pair  $(x, \varepsilon) \in I \times (0, 1)$  as input, uses an arbitrary randomized approximation scheme (not necessarily polynomial time-bounded)  $M$  for  $Q$  as *oracle*, and satisfies the following conditions: (i)  $N$  is a randomized approximation scheme for  $P$  for any choice of oracle  $M$  for  $Q$ ; (ii) every *oracle call* made by  $N$  is of the form  $(w, \delta) \in I' \times (0, 1)$  satisfying  $1/\delta \leq p(|x|, 1/\varepsilon)$ , where  $p$  is a certain absolute polynomial, and an oracle answer is an outcome of  $M$  on the input  $(w, \delta)$ ; and (iii) the running time of  $N$  is bounded from above by a certain polynomial in  $(|x|, 1/\varepsilon)$ , not depending on the choice of the oracle  $M$ . In this case, we write  $P \leq_{\text{APT}} Q$  and we also say that  $P$  is *APT-reducible* (or *APT-reduced*) to  $Q$ . Note that APT-reducibility composes. If  $P \leq_{\text{APT}} Q$  and  $Q \leq_{\text{APT}} P$ , then  $P$  and  $Q$  are said to be *APT-equivalent* and we use the notation  $P \equiv_{\text{APT}} Q$ .

In the definition of  $\text{MAX-PROD-CSP}(\mathcal{F})$  given in Section 1, we also write  $h(x_{i_1}, \dots, x_{i_k})$  to mean  $\langle h, (x_{i_1}, \dots, x_{i_k}) \rangle$  in  $H$ . For notational simplicity, we intend to write, for example,  $\text{MAX-PROD-CSP}(f, \mathcal{F}, \mathcal{G})$  instead of  $\text{MAX-PROD-CSP}(\{f\} \cup \mathcal{F} \cup \mathcal{G})$ . In addition, we abbreviate as  $\text{MAX-PROD-CSP}^*(\mathcal{F})$  the maximization problem  $\text{MAX-PROD-CSP}(\mathcal{F} \cup \mathcal{U})$ .

For any optimization problem  $P$  and any class  $\mathcal{C}$  of optimization problems, we write  $P \leq_{\text{APT}} \mathcal{C}$  if there exists a problem  $Q \in \mathcal{C}$  such that  $P \leq_{\text{APT}} Q$ . Our choice of APT-reducibility makes it possible to prove Lemma 2.2; however, it is not clear whether the lemma implies that  $\text{MAX-PROD-CSP}(\mathcal{F}) \in \text{exp-APX}$ .

**Lemma 2.2**  $\text{MAX-PROD-CSP}(\mathcal{F}) \leq_{\text{APT}} \text{exp-APX}$  for any constraint set  $\mathcal{F}$ .

Hereafter, we introduce the maximization problems stated in Theorem 1.1.

MAX-PROD-IS: (maximum product independent set)

- INSTANCE: an undirected graph  $G = (V, E)$  and a series  $\{w_x\}_{x \in V}$  of vertex weights with  $w_x \in \mathbb{R}^{\geq 0}$ ;
- SOLUTION: an independent set  $A$  on  $G$ ;
- MEASURE: the product  $\prod_{x \in A} w_x$ .

This maximization problem MAX-PROD-IS literally coincides with  $\text{MAX-PROD-CSP}(NAND, \mathcal{G}_0)$ , where  $\mathcal{G}_0 = \{[1, \lambda] \mid \lambda \geq 0\}$ , and it can be easily shown to be NPO-complete. When all input graphs are limited to bipartite graphs, the corresponding problem is called MAX-PROD-BIS.

---

<sup>¶</sup>This function  $\alpha$  must satisfy that there exists a positive polynomial  $p$  for which  $1 \leq \alpha(n) \leq 2^{p(n)}$  for any number  $n \in \mathbb{N}$ .

MAX-PROD-BIS: (maximum product bipartite independent set)

- In MAX-PROD-IS, all input graphs are limited to bipartite graphs.

Since the above two problems can be expressed in the form of MAX-PROD-CSP\*( $\cdot$ ), we can draw the following important conclusion, which becomes part of the proof of the main theorem.

**Lemma 2.3** 1. MAX-PROD-IS  $\leq_{\text{APT}}$  MAX-PROD-CSP\*(OR).

2. MAX-PROD-BIS  $\leq_{\text{APT}}$  MAX-PROD-CSP\*(Implies).

Next, we introduce a special maximization problem, called MAX-PROD-FLOW, whose intuitive setting is explained as follows. Suppose that water flows from point  $u$  to point  $v$  through a one-way pipe at flow rate  $\rho_{(u,v)}$ . A value  $\sigma(x)$  expresses an elevation (indicating either the *bottom level* or the *top level*) of point  $x$  so that water runs from point  $u$  to point  $v$  whenever  $\sigma(u) \geq \sigma(v)$ . More water is added at influx rate  $w_v$  at point  $v$  for which  $\sigma(v) = 1$ .

MAX-PROD-FLOW: (maximum product flow)

- INSTANCE: a directed graph  $G = (V, E)$ , a series  $\{\rho_e\}_{e \in E}$  of flow rates with  $\rho_e \geq 1$ , and a series  $\{w_x\}_{x \in V}$  of influx rates with  $w_x \geq 0$ ;
- SOLUTION: a Boolean assignment  $\sigma$  to  $V$ ;
- MEASURE: the product  $\left( \prod_{(x,y) \in E, \sigma(x) \geq \sigma(y)} \rho_{(x,y)} \right) \left( \prod_{z \in V, \sigma(z)=1} w_z \right)$ .

From the above definition, it is not difficult to prove the following statement.

**Lemma 2.4** For any constraint set  $\mathcal{F} \subseteq \mathcal{IM}_{\text{opt}}$ , MAX-PROD-CSP\*( $\mathcal{F}$ ) is APT-reducible to MAX-PROD-FLOW.

## 2.3 $T_{\text{max}}$ -Constructibility

To pursue notational succinctness, we use the following notations. Let  $f$  be any arity- $k$  constraint. For any two distinct indices  $i, j \in [k]$  and any bit  $c \in \{0, 1\}$ , let  $f^{x_i=c}$  denote the function  $g$  satisfying that  $g(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k) = f(x_1, \dots, x_{i-1}, c, x_{i+1}, \dots, x_k)$  and let  $f^{x_i=x_j}$  be the function  $g$  defined as  $g(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k) = f(x_1, \dots, x_{i-1}, x_j, x_{i+1}, \dots, x_k)$ . Moreover, we denote by  $\max_{y_1, \dots, y_d}(f)$  the function  $g$  defined as  $g(x_1, \dots, x_k) = \max_{(y_1, \dots, y_d) \in \{0,1\}^d} \{f(x_1, \dots, x_k, y_1, \dots, y_d)\}$ , where  $y_1, \dots, y_d$  are all distinct and different from  $x_1, \dots, x_k$ , and let  $\lambda \cdot f$  denote the function satisfying  $(\lambda \cdot f)(x_1, \dots, x_k) = \lambda f(x_1, \dots, x_k)$ .

A helpful tool invented in [10] for counting CSPs is a notion of *T-constructibility*. For our purpose of proving the main theorem, we wish to modify this notion and introduce a notion of  $T_{\text{max}}$ -constructibility. We say that an arity- $k$  constraint  $f$  is  *$T_{\text{max}}$ -constructible* (or  *$T_{\text{max}}$ -constructed*) from a constraint set  $\mathcal{G}$  if  $f$  can be obtained, initially from constraints in  $\mathcal{G}$ , by applying recursively a finite number (possibly zero) of seven functional operations described below.

1. PERMUTATION: for two indices  $i, j \in [k]$  with  $i < j$ , by exchanging two columns  $x_i$  and  $x_j$  in  $(x_1, \dots, x_i, \dots, x_j, \dots, x_k)$ , transform  $g$  into  $g'$ , where  $g'$  is defined as  $g'(x_1, \dots, x_i, \dots, x_j, \dots, x_k) = g(x_1, \dots, x_j, \dots, x_i, \dots, x_k)$ .
2. PINNING: for an index  $i \in [k]$  and a bit  $c \in \{0, 1\}$ , build  $g^{x_i=c}$  from  $g$ .
3. LINKING: for two distinct indices  $i, j \in [k]$ , build  $g^{x_i=x_j}$  from  $g$ .
4. EXPANSION: for an index  $i \in [k]$ , introduce a new “free” variable, say,  $y$  and transform  $g$  into  $g'$  that is defined by  $g'(x_1, \dots, x_i, y, x_{i+1}, \dots, x_k) = g(x_1, \dots, x_i, x_{i+1}, \dots, x_k)$ .

5. MULTIPLICATION: from two constraints  $g_1$  and  $g_2$  of arity  $k$  that share the same input variable series  $(x_1, \dots, x_k)$ , build the constraint  $g_1 \cdot g_2$ , where  $(g_1 \cdot g_2)(x_1, \dots, x_k) = g_1(x_1, \dots, x_k)g_2(x_1, \dots, x_k)$ .
6. MAXIMIZATION: build  $\max_{y_1, \dots, y_d}(g)$  from  $g$ , where  $y_1, \dots, y_d$  are not shared with any other constraint other than this particular constraint  $g$ .
7. NORMALIZATION: for a positive constant  $\lambda$ , build  $\lambda \cdot g$  from  $g$ .

When  $f$  is  $T_{\max}$ -constructible from  $\mathcal{G}$ , we use the notation  $f \leq_{\text{con}}^{\max} \mathcal{G}$ . In particular, when  $\mathcal{G}$  is a singleton  $\{g\}$ , we also write  $f \leq_{\text{con}}^{\max} g$  instead of  $f \leq_{\text{con}}^{\max} \{g\}$ .

It holds that  $T_{\max}$ -constructibility between constraints guarantees APT-reducibility between their corresponding MAX-PROD-CSP\*( $\cdot$ )'s.

**Lemma 2.5** *If  $f \leq_{\text{con}}^{\max} \mathcal{G}$ , then  $\text{MAX-PROD-CSP}^*(f, \mathcal{F})$  is APT-reducible to  $\text{MAX-PROD-CSP}^*(\mathcal{G}, \mathcal{F})$  for any constraint set  $\mathcal{F}$ .*

### 3 Proof of the Main Theorem

Our main theorem—Theorem 1.1—states that all maximization problems of the form of  $\text{MAX-PROD-CSP}^*(\cdot)$  can be classified into three categories. This trichotomy theorem sheds a clear contrast with the dichotomy theorem of Khanna et al. [7] for MAX-CSPs. Hereafter, we will present the proof of Theorem 1.1.

#### 3.1 First Step Toward the Proof

We begin with  $\text{MAX-PROD-CSP}^*(\cdot)$ 's that can be solved in polynomial time.

**Proposition 3.1** *If either  $\mathcal{F} \subseteq \mathcal{AF}$  or  $\mathcal{F} \subseteq \mathcal{ED}$ , then  $\text{MAX-PROD-CSP}^*(\mathcal{F})$  belongs to PO.*

**Proof Sketch.** For every target problem, as in the proof of [10, Lemma 6.1], we can greatly simplify the structure of each input instance so that it depends only on polynomially many solutions. By examining all such solutions deterministically, we surely find its optimal solution. Hence, the target problem belongs to PO. ■

It thus remains to deal with only the case where  $\mathcal{F} \not\subseteq \mathcal{AF}$  and  $\mathcal{F} \not\subseteq \mathcal{ED}$ . In this case, we first make the following key claim that leads to the main theorem.

**Proposition 3.2** *Let  $f$  be any constraint and assume that  $f \notin \mathcal{AF} \cup \mathcal{ED}$ . Let  $\mathcal{F}$  be any set of constraints.*

1. *If  $f \in \mathcal{IM}_{\text{opt}}$ , then  $\text{MAX-PROD-CSP}^*(\text{Implies}, \mathcal{F})$  is APT-reducible to  $\text{MAX-PROD-CSP}^*(f, \mathcal{F})$ .*
2. *If  $f \notin \mathcal{IM}_{\text{opt}}$ , then there exists a constraint  $g \in \{\text{OR}, \text{NAND}\}$  such that  $\text{MAX-PROD-CSP}^*(g, \mathcal{F})$  is APT-reducible to  $\text{MAX-PROD-CSP}^*(f, \mathcal{F})$ .*

We postpone the proof of the above proposition and, meanwhile, we want to prove Theorem 1.1 using the proposition.

**Proof of Theorem 1.1.** If  $\mathcal{F} \subseteq \mathcal{AF}$  or  $\mathcal{F} \subseteq \mathcal{ED}$ , then Proposition 3.1 implies that  $\text{MAX-PROD-CSP}^*(\mathcal{F})$  belongs to PO. Henceforth, we assume that  $\mathcal{F} \not\subseteq \mathcal{AF}$  and  $\mathcal{F} \not\subseteq \mathcal{ED}$ . If  $\mathcal{F} \subseteq \mathcal{IM}_{\text{opt}}$ , then Lemma 2.4 helps APT-reduce  $\text{MAX-PROD-CSP}^*(\mathcal{F})$  to MAX-PROD-FLOW.

Next, we choose a constraint  $f \in \mathcal{F}$  for which  $f \notin \mathcal{AF} \cup \mathcal{ED}$ . Proposition 3.2(1) then yields an APT-reduction from  $\text{MAX-PROD-CSP}^*(\text{Implies})$  to  $\text{MAX-PROD-CSP}^*(f)$ . By Lemma 2.3(2), we obtain  $\text{MAX-PROD-BIS} \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(\text{Implies})$ . Since  $\text{MAX-PROD-CSP}^*(f) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(\mathcal{F})$ , it follows that  $\text{MAX-PROD-BIS}$  is APT-reducible to  $\text{MAX-PROD-CSP}^*(\mathcal{F})$ .

Finally, we assume that  $\mathcal{F} \not\subseteq \mathcal{IM}_{\text{opt}}$ . Take a constraint  $f \in \mathcal{F}$  satisfying that  $f \notin \mathcal{AF} \cup \mathcal{ED} \cup \mathcal{IM}_{\text{opt}}$ . In this case, Proposition 3.2(2) yields an APT-reduction from  $\text{MAX-PROD-CSP}^*(\text{OR})$  to  $\text{MAX-PROD-CSP}^*(f)$ , since  $\text{MAX-PROD-CSP}^*(\text{OR}) \equiv_{\text{APT}} \text{MAX-PROD-CSP}^*(\text{NAND})$ . From  $\text{MAX-PROD-CSP}^*(f) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(\mathcal{F})$ , it immediately follows that  $\text{MAX-PROD-CSP}^*(\text{OR})$  is APT-reducible to  $\text{MAX-PROD-CSP}^*(\mathcal{F})$ . By Lemma 2.3(1),  $\text{MAX-PROD-IS} \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(\text{OR})$ . Therefore, we conclude that  $\text{MAX-PROD-IS}$  is APT-reducible to  $\text{MAX-PROD-CSP}^*(\mathcal{F})$ .  $\square$

### 3.2 Second Step Toward the Proof

To finish the proof of Theorem 1.1, we still need to prove Proposition 3.2. Proving this proposition requires three properties. To describe them, we first review two existing notions from [10]. We say that a constraint  $f$  has *affine support* if  $R_f$  is an affine relation and that  $f$  has *imp support* if  $R_f$  is logically equivalent to a conjunction of a certain “positive” number of relations of the form  $\Delta_0(x)$ ,  $\Delta_1(x)$ , and  $\text{Implies}(x, y)$ . The notation *AFFINE* denotes the set of all affine relations.

In the following three statements,  $\mathcal{F}$  denotes an arbitrary set of constraints.

**Lemma 3.3** *If  $f$  is a non-degenerate constraint in  $\mathcal{IM}_{\text{opt}}$  and has no imp support, then  $\text{MAX-PROD-CSP}^*(\text{Implies}, \mathcal{F}) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(f, \mathcal{F})$ .*

**Proposition 3.4** *Let  $f$  be any constraint having imp support. If either  $f$  has no affine support or  $f \notin \mathcal{ED}$ , then  $\text{MAX-PROD-CSP}^*(\text{Implies}, \mathcal{F})$  is APT-reducible to  $\text{MAX-PROD-CSP}^*(f, \mathcal{F})$ .*

**Proposition 3.5** *Let  $f \notin \mathcal{NZ}$  be any constraint. If  $f$  has neither affine support nor imp support, then there exists a constraint  $g \in \{\text{OR}, \text{NAND}\}$  such that  $\text{MAX-PROD-CSP}^*(g, \mathcal{F}) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(f, \mathcal{F})$ .*

With a help of the above statements, we can prove Proposition 3.2 as follows.

**Proof Sketch of Proposition 3.2.** Let  $f \notin \mathcal{AF} \cup \mathcal{ED}$  be any constraint. We proceed our proof by induction on the arity  $k$  of  $f$ . For the proposition’s claims, (1) and (2), the basis case  $k = 1$  is trivial since  $\mathcal{ED}$  contains all unary constraints. Next, we prove the induction step  $k \geq 3$ . In the remainder of this proof, as our induction hypothesis, we assume that the proposition holds for any arity less than  $k$ . The claims (1) and (2) will be shown separately.

(1) Assume that  $f$  is in  $\mathcal{IM}_{\text{opt}}$ . If  $f$  has imp support, since  $f \notin \mathcal{ED}$ , we can apply Proposition 3.4 and immediately obtain the desired APT-reduction  $\text{MAX-PROD-CSP}^*(\text{Implies}, \mathcal{F}) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(f, \mathcal{F})$ . Otherwise, by Lemma 3.3, we have the desired APT-reduction.

(2) Since  $f$  has no imp support, if  $R_f$  is not affine, then Proposition 3.5 implies that, for a certain  $g_0 \in \{\text{OR}, \text{NAND}\}$ ,  $\text{MAX-PROD-CSP}^*(g_0, \mathcal{F}) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(f, \mathcal{F})$ ; therefore, the desired result follows. To finish the proof, we hereafter assume the affine property of  $R_f$ .

[Case:  $f \in \mathcal{NZ}$ ] Recall that  $f \notin \mathcal{ED}$  and  $R_f \in \text{AFFINE}$ . Since  $f \in \mathcal{NZ}$ , we have  $|R_f| = 2^k$ , and thus  $f$  should be in *clean form* (i.e.,  $f$  contains no factor of the form:  $\Delta_0(x)$ ,  $\Delta_1(x)$ , and  $\text{EQ}(x, y)$ ). As shown in [10, Lemma 7.5], there exists a constraint  $p = (1, x, y, z) \notin \mathcal{ED}$  with



$xyz \neq 0$ ,  $z \neq xy$ , and  $p \leq_{\text{con}}^{\text{max}} f$ . When  $z < xy$ , we can prove that, for a certain  $g_0 \in \{OR, NAND\}$ ,  $\text{MAX-PROD-CSP}^*(g_0, \mathcal{F}) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(p, \mathcal{F})$ . In the case where  $z > xy$ , Lemma 2.1 implies  $p \in \mathcal{IM}_{\text{opt}}$ . Since  $p$  is obtained from  $f$  by pinning operations only, we conclude that  $f \in \mathcal{IM}_{\text{opt}}$ , a contradiction. This finishes the induction step.

[Case:  $f \notin \mathcal{NZ}$ ] We first claim that  $k \geq 3$ . Assume otherwise that  $k = 2$ . Since  $R_f \in \text{AFFINE}$ , it is possible to write  $f$  in the form  $f(x_1, x_2) = \xi_A(x_1, x_2)g(x_1)$  after appropriately permuting variable indices. This places  $f$  within  $\mathcal{AF}$ , a contradiction against the choice of  $f$ . Hence,  $k \geq 3$  holds. Moreover, we can prove the existence of a constraint  $g \notin \mathcal{AF}$  of arity  $m$  for which  $2 \leq m < k$ ,  $g \leq_{\text{con}}^{\text{max}} f$ , and either  $g \in \mathcal{NZ}$  or  $R_g \notin \text{AFFINE}$ . If we can show that (\*) there exists a constraint  $g_0 \in \{OR, NAND\}$  satisfying  $\text{MAX-PROD-CSP}^*(g_0, \mathcal{F}) \leq_{\text{APT}} \text{MAX-PROD-CSP}^*(g, \mathcal{F})$ , then the proposition immediately follows from  $g \leq_{\text{con}}^{\text{max}} f$ . The claim (\*) is split into two cases: (i)  $R_g \in \text{AFFINE}$  and (ii)  $R_g \notin \text{AFFINE}$ . For (i), we apply the induction hypothesis. For (ii), we apply Propositions 3.4–3.5. Thus, we have completed the induction step.  $\square$

In this end, we have completed the proof of the main theorem. The detailed proofs omitted in this extended abstract will be published shortly. We hope that our systematic treatment of MAX-PROD-CSP\*s would lead to a study on a far wider class of optimization problems.

## References

- [1] G. Ausiello, A. Marchetti-Spaccamela, and M. Protasi. Full approximability of a class of problems over power sets. In: *Proc. of CAAP 81*, LNCS, vol. 112, pp. 76–87. Springer, Berlin (1981)
- [2] S. Boyd, S. J. Kim, L. Vandenbergh, and A. Hassibi. A tutorial on geometric programming. *Optm. Eng.* 8, 67–127 (2007)
- [3] N. Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *J. Comput. System Sci.* 51, 511–522 (1995)
- [4] M. Dyer, L. A. Goldberg, C. Greenhill, and M. Jerrum. The relative complexity of approximating counting problems. *Algorithmica* 38, 471–500 (2003)
- [5] O. H. Ibarra and C. E. Kim. Fast approximation for the knapsack and sum of subset problems. *J. ACM* 22, 463–468 (1975)
- [6] H. Konno and T. Kuno. Linear multiplicative programming. *Math. Program.* 56, 51–64 (1992)
- [7] S. Khanna, M. Sudan, L. Trevisan, and D. P. Williamson. The approximability of constraint satisfaction problems. *SIAM J. Comput.* 30, 1863–1920 (2001)
- [8] A. Marchetti-Spaccamela and S. Romano. On different approximation criteria for subset product problems. *Inform. Process. Lett.* 21, 213–218 (1985)
- [9] C. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes. *J. Comput. System Sci.* 43, 425–440 (1991)
- [10] T. Yamakami. Approximate counting for complex-weighted Boolean constraint satisfaction problems. Available at arXiv:1007.0391. An older version appeared in the *Proc. of WAOA 2010*, LNCS, vol. 6534, pp. 261–272. Springer, Heidelberg (2011)