Electronic Colloquium on Computational Complexity, Report No. 44 (2008)

Complexity of Counting the Optimal Solutions*

Miki Hermann
LIX (CNRS, UMR 7161)
École Polytechnique
91128 Palaiseau, France
hermann@lix.polytechnique.fr

Reinhard Pichler
Institut für Informationssysteme
Technische Universität Wien
A-1040 Wien, Austria
pichler@dbai.tuwien.ac.at

Abstract

Following the approach of Hemaspaandra and Vollmer, we can define counting complexity classes $\#\cdot\mathcal{C}$ for any complexity class $\mathcal{C}$ of decision problems. In particular, the classes $\#\cdot\Pi_k\mathrm{P}$ with $k \geq 1$ corresponding to all levels of the polynomial hierarchy have thus been studied. However, for a large variety of counting problems arising from optimization problems, a precise complexity classification turns out to be impossible with these classes. In order to remedy this unsatisfactory situation, we introduce a hierarchy of new counting complexity classes $\#\cdot\mathrm{Opt}_k\mathrm{P}$ and $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$ with $k \geq 1$. We prove several important properties of these new classes, like closure properties and the relationship with the $\#\cdot\Pi_k\mathrm{P}$-classes. Moreover, we establish the completeness of several natural counting complexity problems for these new classes.

1 Introduction

Many natural decision problems are known to be complete for the class $\Theta_k\mathrm{P} = \Delta_k\mathrm{P}[\log n]$, defined by Wagner in [21], or for $\Delta_k\mathrm{P}$. In particular, they often occur in variants of $\Sigma_{k-1}\mathrm{P}$-complete problems when cardinality-minimality or weight-minimality (or, likewise, cardinality-maximality or weight-maximality) is imposed as an additional constraint. Two prototypical representatives of such problems are as follows (The completeness of these problems in $\Theta_2\mathrm{P}$ and $\Delta_2\mathrm{P}$, respectively, is implicit in [6]).

Problem: Min-Card-SAT
Input: A propositional formula φ in conjunctive normal form over variables X and a subset of variables $X' \subseteq X$.
Question: Are X' set to true in some cardinality-minimal model of φ?

Problem: Min-Weight-SAT
Input: A propositional formula φ in conjunctive normal form over variables in X together with a weight function $w\colon X \to \mathbb{N}$, and a subset of variables $X' \subseteq X$.
Question: Are X' set to true in some weight-minimal model of φ?

*A preliminary version without proofs appeared in COCOON 2008.

ISSN 1433-8092

A straightforward Θ_2P-algorithm for MIN-CARD-SAT computes the minimum cardinality of the models of φ by means of logarithmically many calls to an NP-oracle, asking questions of the type "Does φ have a model of size $\leq k$?". As soon as the minimum cardinality k_0 is known, we can proceed by a simple NP-algorithm, checking if the subset X' is true in some model of size k_0. Analogously, a Δ_2P-algorithm for MIN-WEIGHT-SAT first computes the minimum weight of all models of φ. In any reasonable representation, the weights are exponential with respect to their representation (e.g., they are represented in binary notation). Hence, the straightforward algorithm for computing the minimum weight needs logarithmically many calls to an NP-oracle with respect to the total weight of all variables. This comes down to polynomially many calls with respect to the representation of the weights.

Note that the membership in Θ_2P and Δ_2P recalled above is in great contrast to *subset-minimality*, i.e., minimality with respect to set inclusion (or, likewise, *subset-maximality*), which often raises the complexity one level higher in the polynomial hierarchy. E.g., the following problem is well-known to be Σ_2P-complete (cf. [14]).

Problem: MIN-SAT
Input: A propositional formula φ in conjunctive normal form over variables X and a subset $X' \subseteq X$.
Question: Are X' set to true in some subset-minimal model of φ?

As far as the complexity of the corresponding *counting problems* is concerned, only the counting problem corresponding to MIN-SAT has been satisfactorily classified so far. The following problem was shown to be #·coNP-complete in [3]: Given a propositional formula φ in conjunctive normal form, how many subset-minimal models does φ have? On the other hand, the counting complexity of the remaining aforementioned problems has remained obscure. The main goal of this paper is to introduce new counting complexity classes #·OptP and #·OptP[$\log n$], needed to pinpoint the precise complexity of these and many similar optimality counting problems. We will also show the relationship of these new classes with respect to the known classes in the counting hierarchy. Moreover, we will show that these new classes are not identical to already known ones, unless the polynomial hierarchy collapses. Finally, we will present several natural optimization counting problems, which turn out to be complete for one or the other introduced counting class. The definition of new natural counting complexity classes is by no means limited to the first level of the polynomial hierarchy. Indeed, we will show how the counting complexity classes #·OptP and #·OptP[$\log n$] can be generalized to #·Opt$_k$P and #·Opt$_k$P[$\log n$] for arbitrary $k \geq 1$ with #·OptP = #·Opt$_1$P and #·OptP[$\log n$] = #·Opt$_1$P[$\log n$].

The paper is organized as follows: In Section 2, we recall some basic notions and results from counting complexity. The new counting complexity classes #·Opt$_k$P and #·Opt$_k$P[$\log n$] are introduced and important properties of them are shown in Section 3. We present several complete problems for these new classes in Section 4. Finally, concluding remarks are given in Section 5.

2 Preliminaries

We recall the necessary concepts and definitions, but we assume that the reader is familiar with the basic notions in computational counting complexity. For more information, the interested reader is referred to Chapter 18 in the book [14] or the survey [4].

The study of *counting problems* was initiated by Valiant in [19,20]. While decision problems ask if at least one solution of a given problem instance exists, counting problems ask for the number of

different solutions. The most intensively studied counting complexity class is #P, which denotes the functions that count the number of accepting paths of a non-deterministic polynomial-time Turing machine. In other words, #P captures the counting problems corresponding to decision problems contained in NP. By allowing the non-deterministic polynomial-time Turing machine access to an oracle in NP, Σ_2P, Σ_3P, ..., we can define an infinite hierarchy of counting complexity classes.

Alternatively, a *counting problem* is presented using a suitable *witness* function which for every input x, returns a set of *witnesses* for x. Formally, a *witness* function is a function $A\colon \Sigma^* \to \mathcal{P}^{<\omega}(\Gamma^*)$, where Σ and Γ are two alphabets, and $\mathcal{P}^{<\omega}(\Gamma^*)$ is the collection of all finite subsets of Γ^*. Every such witness function gives rise to the following *counting problem*: given a string $x \in \Sigma^*$, find the cardinality $|A(x)|$ of the *witness* set $A(x)$. According to [7], if $\mathcal{C}$ is a complexity class of decision problems, we define $\#\cdot\mathcal{C}$ to be the class of all counting problems $\#\cdot A$ whose witness function A satisfies the following conditions.

1. There is a polynomial $p(n)$ such that for every x and every $y \in A(x)$, we have that $|y| \leq p(|x|)$, where $|x|$ is the length of x and $|y|$ is the length of y;
2. The decision problem "given x and y, is $y \in A(x)$?" is in $\mathcal{C}$.

It is easy to verify that #P = #·P. The counting hierarchy is ordered by linear inclusion [7]. In particular, we have that #P $\subseteq$ #·coNP $\subseteq$ #·Π_2P $\subseteq$ #·Π_3P, etc. Analogously, one can define the classes #·NP, #·Σ_2P, #·Σ_3P, etc. Toda and Ogiwara [17] determined the precise relationship between these classes as follows: #·Σ_kP $\subseteq$ #·$\mathrm{P}^{\Sigma_k\mathrm{P}}$ = #·Π_kP. Since the identity #·$\mathrm{P}^{\Sigma_k\mathrm{P}}$ = #·Δ_{k+1}P trivially holds, Toda and Ogiwara also showed that there are no new Δ-classes in the counting hierarchy.

The prototypical #·Π_kP-complete problem for $k \in \mathbb{N}$ is #Π_kSAT [3], defined as follows. Given a formula $\psi(X) = \forall Y_1 \exists Y_2 \cdots Q Y_k\ \varphi(X, Y_1, \ldots, Y_k)$, where φ is a Boolean formula and X, Y_1, ..., Y_k are sets of propositional variables, count the number of truth assignments to the variables in X that satisfy ψ.

Completeness of counting problems is usually proved by means of polynomial-time Turing reductions, also called Cook reductions. However, these reductions do not preserve the counting classes #·Π_kP [18]. Hence, *parsimonious reductions* are usually considered instead. Consider two counting problems $\#\cdot A\colon \Sigma^* \to \mathbb{N}$ and $\#\cdot B\colon \Sigma^* \to \mathbb{N}$. We say that $\#\cdot A$ reduces to $\#\cdot B$ via a parsimonious reduction if there exists a polynomial-time function $f \in$ FP, such that for each $x \in \Sigma^*$ we have $\#\cdot A(x) = \#\cdot B(f(x))$. Parsimonious reductions are a special case of Karp reductions with a one-to-one relation between solutions for the corresponding instances of the problems $\#\cdot A$ and $\#\cdot B$. However, parsimonious reductions are not always strong enough to prove completeness of well-known problems in counting complexity classes. E.g., the problem #POSITIVE 2SAT [2, 20] of counting satisfying assignments to a propositional formula with positive literals only and with two literals per clause cannot be #P-complete under parsimonious reductions, unless P = NP. Therefore Durand, Hermann, and Kolaitis [3] generalized parsimonious reductions to *subtractive reductions* and showed that all the classes #·Π_kP are closed under them. Subtractive reductions are defined as follows. The counting problem $\#\cdot A$ reduces to $\#\cdot B$ via a *strong subtractive reduction* if there exist two polynomial-time computable functions f and g such that for each $x \in \Sigma^*$ we have

$$B(f(x)) \subseteq B(g(x)) \quad \text{and} \quad |A(x)| = |B(g(x))| - |B(f(x))| \ .$$

A *subtractive reduction* is a composition (transitive closure) of a finite sequence of strong subtractive reductions. Thus, a *parsimonious* reduction corresponds to the special case of a strong subtractive reduction with $B(f(x)) = \emptyset$. As it is mentioned in [3], subtractive reductions have been shown to be

strong enough to prove completeness of many interesting problems in #P and other counting classes, but their power remains tame enough to preserve several interesting counting classes between #P and #PSPACE.

3 Optimization Counting Complexity Classes

Recall that, according to [7], a counting complexity class $\#\cdot\mathcal{C}$ can in principle be defined for any decision complexity class $\mathcal{C}$. However, as far as the polynomial hierarchy is concerned, this definition does not yield the desired diversity of counting complexity classes. In fact, if we simply consider $\#\cdot\mathcal{C}$ for either $\mathcal{C} = \Delta_k\mathrm{P}$ or $\mathcal{C} = \Theta_k\mathrm{P}$, then we do not get any new complexity classes, since the relationship $\#\cdot\Theta_k\mathrm{P} = \#\cdot\Delta_k\mathrm{P} = \#\cdot\Pi_{k-1}\mathrm{P}$ is an immediate consequence of the aforementioned result by Toda and Ogiwara [17]. Hence a different approach is necessary if we want to obtain a more fine grained stratification of the counting hierarchy. For this reason we introduce in the sequel the counting classes $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$ and $\#\cdot\mathrm{Opt}_k\mathrm{P}$ for each $k \in \mathbb{N}$, which will be appropriate for optimization counting problems. Of special interest will be the classes $\#\cdot\mathrm{OptP}[\log n] = \#\cdot\mathrm{Opt}_1\mathrm{P}[\log n]$ and $\#\cdot\mathrm{OptP} = \#\cdot\mathrm{Opt}_1\mathrm{P}$. We will define the new counting complexity classes via the nondeterministic transducer model (see [15]), as well as by an equivalent predicate based definition following the approach from [7]. The following definition generalizes the definition of nondeterministic transducers [15] to oracle machines.

Definition 1 A *nondeterministic transducer* M is a nondeterministic polynomial-time bounded Turing machine, such that every accepting path writes a binary number. If M is equipped with an oracle from the complexity class $\mathcal{C}$, then it is called a *nondeterministic transducer with* $\mathcal{C}$*-oracle*. A $\Sigma_k\mathrm{P}$*-transducer* M is a nondeterministic transducer with a $\Sigma_{k-1}\mathrm{P}$ oracle. We identify nondeterministic transducers without oracle and $\Sigma_1\mathrm{P}$-transducers.

For $x \in \Sigma^*$, we write $\mathrm{opt}_M(x)$ to denote the *optimal* value, which can be either the *maximum* or the *minimum*, on any accepting path of the computation of M on x. If no accepting path exists then $\mathrm{opt}_M(x)$ is undefined.

The above definition of a nondeterministic transducer is similar to a metric Turing machine defined in [11] and its generalization in [12]. However, our definition deviates from the machine models in [11, 12] in the following aspects:

1. We take the optimum value only over the *accepting* paths, while in [11] every path is accepting. Our ultimate goal is to count the number of optimal solutions. Hence, above all, the objects that we want to count have to be *solutions*, i.e., correspond to an accepting computation, and only in the second place we are interested in the optimum.

2. In [11], only the maximum value is considered and it is mentioned that the minimum value is treated analogously. We prefer to make the applicability both to max and min explicit. The definition of the counting complexity classes below is not affected by this distinction.

3. In [12], NP-metric Turing machines were generalized to higher levels of the polynomial hierarchy by allowing alternations of minimum and maximum computations. However, for our purposes, in particular for the predicate-based characterization of the counting complexity classes below, the generalization via oracles is more convenient. Proving the equivalence of the two kinds of generalizations is straightforward.

It will be clear in the sequel that the generalization of nondeterministic transducers [15] to oracle machines is exactly the model we need. A similar idea but with a deterministic Turing transducer was used by Jenner and Torán in [8] to characterize the functional complexity classes $\mathrm{FP}_{\|}^{\mathrm{NP}}$, $\mathrm{FP}_{\log}^{\mathrm{NP}}$, and $\mathrm{FL}_{\log}^{\mathrm{NP}}$.

Definition 2 We say that a counting problem $\#\cdot A\colon \Sigma^* \to \mathbb{N}$ is in the class $\#\cdot\mathrm{Opt}_k\mathrm{P}$ for some $k \in \mathbb{N}$, if there is a $\Sigma_k\mathrm{P}$-transducer M, such that $\#\cdot A(x)$ is the number of accepting paths of the computation of M on x yielding the optimum value $\mathrm{opt}_M(x)$. If no accepting path exists then $\#\cdot A(x) = 0$. If the length of the binary number written by M is bounded by $O(z(|x|))$ for some function $z(n)$, then $\#\cdot A$ is in the class $\#\cdot\mathrm{Opt}_k\mathrm{P}[z(n)]$.

In this paper, we are only interested in $\#\cdot\mathrm{Opt}_k\mathrm{P}[z(n)]$ for two types of functions $z(n)$, namely the polynomial function $z(n) = n^{O(1)}$ and the logarithmic function $z(n) = \log n$. Clearly, $\#\cdot\mathrm{Opt}_k\mathrm{P}$ is the same as $\#\cdot\mathrm{Opt}_k\mathrm{P}[n^{O(1)}]$.

Distinguishing between max and min gives no additional computational power, as it is formalized by the following result.

Proposition 3 *Suppose that some counting problem $\#\cdot A\colon \Sigma^* \to \mathbb{N}$ is defined in terms of a $\Sigma_k\mathrm{P}$-transducer M with the optimum being the maximum (minimum). Then there exists a parsimonious reduction to a counting problem $\#\cdot A'$ defined via a $\Sigma_k\mathrm{P}$-transducer M' with the optimum value corresponding to the minimum (maximum).*

Proof. Let K denote the maximum length of the output on any computation path of M on input x. Denote the output on a computation path π of M as b_π. Construct M' producing exactly the same set of computation paths as M, where the output b'_π on path π is $2^{K+1} - b_\pi$. The paths in M producing the maximum (minimum) over all values b_π correspond one-to-one to the paths in M' producing the minimum (maximum) over all values b'_π. □

Krentel defined in [11] the class $\mathrm{OptP}[z(n)]$ of optimization problems for a given function $z(n)$. He showed that $\mathrm{OptP}[z(n)]$ essentially corresponds to $\mathrm{FP}^{\mathrm{NP}[z(n)]}$ (see also [9]). More precisely, for every "smooth" function[1] $z(n)$ (see [11]) we have the inclusion $\mathrm{OptP}[z(n)] \subseteq \mathrm{FP}^{\mathrm{NP}[z(n)]}$ and every function $f \in \mathrm{FP}^{\mathrm{NP}[z(n)]}$ can be represented as an $\mathrm{OptP}[z(n)]$-problem followed by a polynomial-time function h. This correspondence between $\mathrm{OptP}[z(n)]$ and $\mathrm{FP}^{\mathrm{NP}[z(n)]}$ can be generalized as follows:

Lemma 4 *For every nondeterministic transducer M with a $\Delta_k\mathrm{P}$ oracle there exists an* equivalent *$\Sigma_k\mathrm{P}$-transducer M' with the following properties: (i) For every input x there exists a bijective function h from the computation paths of M on x to the computation paths of M' on x, such that a path π in M is accepting if and only if the path $h(\pi)$ in M' is accepting and, (ii) the output written on path π is identical to the output written on path $h(\pi)$.*

Proof. By the definition of $\Delta_k\mathrm{P}$, the oracle of M is realized by a deterministic polynomial time Turing machine N with a $\Sigma_{k-1}\mathrm{P}$ oracle. Without loss of generality, the states in M and N are disjoint apart from the query state $q_?$ and the answer states q_Y and q_N. Then we construct the machine M' as follows: The input tape of M is also the input tape of M'. The work tapes of M'

[1] A function $f\colon \mathbb{N} \to \mathbb{N}$ is *smooth* if it is nondecreasing and its unary representation is computable in polynomial time.

are the work tapes of M plus the query tape of M plus the work tapes of N. Moreover, we identify the input tape of N with the query tape of M. As far as the states of M' are concerned, we replace the query state $q_?$ of M by the initial state q_0 of N. Moreover, the answer states q_Y and q_N of M are considered as "ordinary" states. The states of M' are thus the states of M (without its query state) plus the states of N. In addition, we will need some auxiliary states for clean-up tasks, which we do not describe in detail here. Finally, the transition relation of M' consists of the transition relation of M (where the query state $q_?$ is replaced by q_0 as stipulated before) plus the transition function of N. In addition, we need some transitions for clean up tasks, i.e.: when the machine is about to enter either the state q_Y or q_N of M (i.e., the former answer states which are now treated as ordinary states), then it first enters some auxiliary state and erases the contents of the tapes of N first. Clearly, the resulting Turing machine is a nondeterministic transducer with a $\Sigma_{k-1}\mathrm{P}$ oracle. Moreover, on any input x, any path π of M is now extended to exactly one path π' of M', since N is deterministic. Also π is accepting if and only if π' is accepting and the output written by M' on the path π' is the same as the output written by M on π. □

In other words, Lemma 4 shows that replacing the $\Sigma_{k-1}\mathrm{P}$ oracle in a $\Sigma_k\mathrm{P}$-transducer by a $\Delta_k\mathrm{P}$ oracle does not increase the expressive power.

We show next that the definition of $\#\cdot\mathrm{Opt}_k\mathrm{P}[z(n)]$ via Turing machines (see Definition 1) has an equivalent definition via predicates. The basic idea is to decompose the computation of a $\Sigma_k\mathrm{P}$-transducer M into a predicate B, which associates inputs x with computations y, and a function f which computes the number written by the transducer M following the computation path y.

Theorem 5 *For any function $z(n)$, a counting problem $\#\cdot A\colon \Sigma^* \to \mathbb{N}$ is in the class $\#\cdot\mathrm{Opt}_k\mathrm{P}[z(n)]$ if and only if there exist an alphabet Γ, a predicate B on $\Sigma^* \times \Gamma^*$, and a polynomial-time computable function $f\colon \Gamma^* \to \mathbb{N}$ satisfying the following conditions.*

(i) There is a polynomial $p(n)$ such that every pair of strings $(x, y) \in B$ satisfies the relation $|y| \leq p(|x|)$;
(ii) The predicate B is decidable by a $\Delta_k\mathrm{P}$ algorithm;
(iii) The length of $f(y)$ is bounded by $O(z(|x|))$ for every (x, y);
(iv) $\mathrm{opt}_f^B(x) = \mathrm{opt}(\{f(y) \mid (x, y) \in B\})$ with $\mathrm{opt} \in \{\max, \min\}$;
(v) $A(x) = \{y \mid (x, y) \in B \wedge f(y) = \mathrm{opt}_f^B(x)\}$.

Proof. [⇒]: Let $\#\cdot A$ be a counting problem in $\#\cdot\mathrm{OptP}[z(n)]$ and let M be the corresponding $\Sigma_k\mathrm{P}$-transducer. Without loss of generality, we assume that the nondeterministic Turing machine M makes on every input x only binary choices denoted by 0 and 1. Let $b = p(|x|)$ be the polynomial computation bound of the Turing machine M on every given input x. Each computation path π of M on an input x corresponds to exactly one run $a_1 \cdots a_b$ of nondeterministic choices with $a_i \in \{0, 1\}$. We define a binary predicate $B \subseteq \Sigma^* \times \Gamma^*$, where $\Gamma = \{0, 1\}$, with the intended meaning that every string $y \in \{0, 1\}^*$ describes the computation path of M on x, such that a pair $(x, y) \in \Sigma^* \times \Gamma^*$ belongs to B if and only if y describes an accepting computation path. The predicate B clearly satisfies the condition (i) and is decidable by a $\Delta_k\mathrm{P}$ algorithm, i.e., satisfying condition (ii). Finally, we define $f(y)$ as the output on the computation path y, setting $f(y) = 0$ for any non-accepting path. Then the predicate B and the function f fulfill the conditions (iii) to (v). In particular, the equality $A(x) = \{y \mid (x, y) \in B \wedge f(y) = \mathrm{opt}_f^B(x)\}$ holds.

[⇐]: Suppose that $\#\cdot A\colon \Sigma^* \to \mathbb{N}$ is a counting problem such that $A(x) = \{y \mid (x, y) \in B \wedge f(y) = \mathrm{opt}_f^B(x)\}$ holds for a predicate $B \subseteq \Sigma^* \times \Gamma^*$ decidable by a $\Delta_k\mathrm{P}$ algorithm and a polynomial-time

computable function f. We define a nondeterministic transducer M with a Δ_kP oracle as follows. The machine M on input x has a computation path $y \in \Gamma^*$ satisfying $|y| \leq p(|x|)$, where $p(|x|)$ is the polynomial bound on the length of all strings y with $(x, y) \in B$. We stipulate that a path y of M on x is accepting if and only if $(x, y) \in B$. By condition (ii), the latter test can be done by a Δ_kP oracle. Moreover, on every accepting path y the machine M outputs the binary representation of $f(y)$ computable in polynomial time. Hence, M is a nondeterministic transducer with a Δ_kP oracle, such that $A(x) = \{y \mid y \text{ is an accepting path yielding the optimal value } \mathrm{opt}_f^B(x)\}$ holds. By Lemma 4, there exists a Σ_kP-transducer M' equivalent to M. Hence, the counting problem $\#\cdot A$ is defined via a Σ_kP-transducer, namely M'. □

As far as complete problems for these new complexity classes are concerned, we propose the following natural generalizations of minimum cardinality and minimum weight counting satisfiability problems to quantified Boolean formulas.

Problem: #Min-Card-Π_kSAT
Input: A Π_kSAT formula $\psi(X) = \forall Y_1 \exists Y_2 \cdots Q Y_k \ \varphi(X, Y_1, \ldots, Y_k)$ with $k \in \mathbb{N}$, where φ is a quantifier-free formula and $X, Y_1, \ldots, Y_k$ are sets of propositional variables, such that Q is either $\exists$ (for k even) or $\forall$ (for k odd).
Output: Number of cardinality-minimal models of $\psi(X)$ or 0 if $\psi(X)$ is unsatisfiable.

Problem: #Min-Weight-Π_kSAT
Input: A Π_kSAT formula $\psi(X) = \forall Y_1 \exists Y_2 \cdots Q_k Y_k \ \varphi(X, Y_1, \ldots, Y_k)$ with $k \in \mathbb{N}$, where φ is a quantifier-free formula and $X, Y_1, \ldots, Y_k$ are sets of propositional variables, and a weight function $w \colon X \to \mathbb{N}$ assigning positive values to each variable $x \in X$.
Output: Number of weight-minimal models of $\psi(X)$ or 0 if $\psi(X)$ is unsatisfiable.

We define the classes #Min-Card-SAT and #Min-Weight-SAT to be the #Min-Card-Π_0SAT and #Min-Weight-Π_0SAT, respectively. Moreover, we can assume, following the ideas of Wrathal [22], that the formula φ is in CNF for k even and in DNF for k odd. Notice that for k *even* (*odd*), the formula φ has an *odd* (*even*) number of variable vectors, since the first variable block X remains always unquantified.

Theorem 6 *For every $k \in \mathbb{N}$, the following problems are complete via parsimonious reductions.* #Min-Weight-Π_kSAT *is* $\#\cdot\mathrm{Opt}_{k+1}$P*-complete and* #Min-Card-Π_kSAT *is* $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$*-complete.*

Proof. For the membership, we show that for both counting problems there exist an appropriate predicate $B \in \Delta_{k+1}$P (actually, we shall even show $B \in \Sigma_k$P) and a polynomial-time computable function f according to Theorem 5. Let B be a binary predicate defined on pairs (x, y), where x is a Π_kSAT formula and y is a truth assignment to the variables in x. Clearly, there exists a polynomial function p, such that $|y| \leq p(|x|)$ holds for all $(x, y) \in B$. Moreover, B can be decided by a Π_kP algorithm using a Σ_kP oracle and reversing its answer. For the problem #Min-Weight-Π_kSAT, we define $f(y)$ as the total weight of the variables which are evaluated to true in y. This function value requires polynomial space with respect to the input. For #Min-Card-Π_kSAT, we define $f(y)$ as the number of variables evaluated to true in y. The latter function value requires logarithmic space with respect to the input. In both cases, the function f can be computed in polynomial time. The set of weight-minimal respectively cardinality-minimal models of a given formula x corresponds

to the set $\{y \mid (x,y) \in B \wedge f(y) = \mathrm{opt}_f^B(x)\}$ for the respective function f. By Theorem 5, #Min-Weight-Π_kSAT is in $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}$ and #Min-Card-Π_kSAT is in $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$.

Hardness of #Min-Weight-Π_kSAT. Let $\#\cdot A$ be an arbitrary counting problem in $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}$ and let x be an instance of $\#\cdot A$. By Definition 2, there exists a $\Sigma_{k+1}\mathrm{P}$-transducer M, such that $\#\cdot A(x)$ is the number of computation paths of M on x which yield the optimum value $\mathrm{opt}_M(x)$. By Proposition 3 we may assume that the optimum value is the minimum. By Galil's construction in [5], the computation of a polynomially time bounded nondeterministic Turing machine M can be parsimoniously reduced in polynomial time to a propositional formula φ, such that there is a one-to-one correspondence between the accepting computation paths of M on x and the satisfying truth assignments of φ. Likewise, if M is equipped with a $\Sigma_k\mathrm{P}$ oracle, then the computation of M can be parsimoniously reduced in polynomial time to a Π_kSAT formula ψ, such that there is a one-to-one correspondence between the accepting computation paths of M on x and the satisfying truth assignments of ψ.

Suppose that the length of the output of M on any path is bounded by $m = p(|x|)$ for some polynomial $p(n)$. We may assume that every output of M on x has precisely this length. This can be easily achieved by padding of leading zeros. Suppose that ψ is of the form $\psi(y_1, \ldots, y_\ell, z_1, \ldots, z_m)$, such that $z_1, \ldots, z_m$ correspond to the output of a computation path and $y_1, \ldots, y_\ell$ are the remaining variables needed to encode the computation of M on x. Note that every propositional variable must get a positive weight. However, we can make sure that the truth assignment to $y_1, \ldots, y_\ell$ has no effect on the computation of the optimum by transforming ψ to the formula $\psi' = \psi(y_1, \ldots, y_\ell, y'_1, \ldots, y'_\ell, z_1, \ldots, z_m) = \psi \wedge (y_1 \equiv \neg y'_1) \wedge \cdots \wedge (y'_\ell \equiv \neg y'_\ell)$. In other words, any satisfying truth assignment of ψ' will assign the value true to exactly ℓ variables out of the 2ℓ variables $y_1, \ldots, y_\ell, y'_1, \ldots, y'_\ell$.

Now the desired instance of #Min-Weight-Π_kSAT consists of the propositional formula ψ' and the following weights: $w(y_i) = w(y'_i) = 1$ for all $i \in \{1, \ldots, \ell\}$ and $w(z_j) = 2^{m-j}$ for all $j \in \{1, \ldots, m\}$. The $\Sigma_{k+1}\mathrm{P}$-transducer M has no accepting computation path on x if and only if ψ (and thus also ψ') is unsatisfiable. Therefore, in this case, both counting problems yield the same result, namely 0. Suppose now that M does have an accepting computation path on x. The number $z_1 \cdots z_m$ (written in binary notation) output on an accepting path of M corresponds to $\sum_{j=1}^m z_j \cdot w(z_j)$, i.e., the total weight of the variables evaluated to true. Hence, there is a one-to-one correspondence between the accepting computation paths of M on x which yield the minimum value $\mathrm{opt}_M(x) = z_1 \cdots z_m$ and the satisfying truth assignments of ψ' with minimal weight $\ell + \sum_{j=1}^m z_j \cdot w(z_j)$.

Hardness of #Min-Card-Π_kSAT. Adapting the above proof to the counting problem #Min-Card-Π_kSAT is easy. Note that now every unquantified variable has the weight 1. On the other hand, the length m of the binary representation of the numbers output by M is logarithmically bounded. Let $w'(z_j) = 2^{m-j}$ for $j \in \{1, \ldots, m\}$. We transform the previous formula ψ' into the desired instance ψ'' of #Min-Card-Π_kSAT by producing $w'(z_j) - 1 = 2^{m-j} - 1$ copies of z_j for each $j \in \{1, \ldots, m\}$. For each z_j we construct the formula $\rho_j = (z_j \equiv z_{j1}) \wedge (z_j \equiv z_{j2}) \wedge \cdots \wedge (z_j \equiv z_{jK})$ with $K = 2^{m-j} - 1$. Then we set $\psi'' = \psi' \wedge \rho_1 \wedge \cdots \wedge \rho_m$. It is straightforward to verify that there exists a one-to-one correspondence between the accepting computation paths of M on x yielding the optimum value $\mathrm{opt}_M(x) = z_1 \cdots z_m$ and the satisfying assignments of ψ'' with minimal Hamming weight $\ell + \sum_{j=1}^m z_j \cdot w'(z_j)$. □

As usual, also the versions of #Min-Weight-Π_kSAT and #Min-Card-Π_kSAT restricted to 3 literals per clause are $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}$-complete and $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$-complete, respectively, since

there exists a parsimonious reduction to them, presented e.g. in [10].

Apart from containing natural complete problems, a complexity class should also be closed with respect to an appropriate type of reductions. We consider the closure of the considered counting classes under subtractive reductions. Note that we cannot expect the class $\#\cdot\mathrm{Opt}_k\mathrm{P}[z(n)]$ to be closed under subtractive reductions for any function $z(n)$ since we can always get an arbitrary polynomial speed-up simply by padding the input. We show in the sequel that the two most interesting cases, namely $\#\cdot\mathrm{Opt}_k\mathrm{P}$ and $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$ for each $k \in \mathbb{N}$, are indeed closed under subtractive reductions.

Theorem 7 *The complexity classes $\#\cdot\mathrm{Opt}_k\mathrm{P}$ and $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$ are closed under subtractive reductions for all $k \in \mathbb{N}$.*

Proof. Suppose that some counting problem $\#\cdot A$ belongs to $\#\cdot\mathrm{Opt}_k\mathrm{P}$ (respectively to $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$) and that another counting problem $\#\cdot A'$ reduces to $\#\cdot A$ via a strong subtractive reduction. We need to show that $\#\cdot A'$ also belongs to $\#\cdot\mathrm{Opt}_k\mathrm{P}$ (respectively to $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$). Following Theorem 5, there exists a binary predicate B decidable by a $\Delta_k\mathrm{P}$ algorithm and a polynomial p, such that every pair $(x, y) \in B$ satisfies the relation $|y| \leq p(|x|)$, together with a polynomial-time computable function $f\colon \Gamma^* \to \mathbb{N}$, such that $A(x) = \{y \mid (x, y) \in B \wedge f(y) = \mathrm{opt}_f^B(x)\}$ and the length of $f(y)$ is polynomially (resp. logarithmically) bounded with respect to $|x|$.

Since $\#\cdot A'$ reduces to $\#\cdot A$ via a strong subtractive reduction, there are two polynomial-time computable functions g_1 and g_2 such that for each $x \in \Sigma^*$ we have $A(g_1(x)) \subseteq A(g_2(x))$ and $|A'(x)| = |A(g_2(x))| - |A(g_1(x))|$. Now let B' be a binary predicate over $\Sigma^* \times (\Sigma^* \cdot \{*\} \cdot \Sigma^* \cdot \{*\} \cdot \Gamma^*)$, where $*$ is a delimiter symbol not occurring in $\Sigma \cup \Gamma$. The predicate B' contains the pairs (x, y') where $y' = g_1(x) * g_2(x) * y$ such that $(g_2(x), y) \in B$ and $(g_1(x), y) \notin B$. We define the function f' on the strings y' as $f'(g_1(x) * g_2(x) * y) = f(y)$. Thus, a pair (x, y') belongs to B' if and only if it is accepted by the following algorithm:

1. extract $g_1(x)$, $g_2(x)$, and y from y';
2. check that $(g_2(x), y)$ belongs to B;
3. check that $(g_1(x), y)$ does not belong to B.
4. check that $f'(y') = f(y)$.

Since $\Delta_k\mathrm{P}$ is closed under complement, the predicate B' is decidable by a $\Delta_k\mathrm{P}$ algorithm and there exists a polynomial function p, such that the relation $|y'| \leq p(|x|)$ is satisfied for each $(x, y') \in B'$. The function f' is computable in polynomial time and the length of the function values $f'(g_1(x) * g_2(x) * y)$ is polynomially (logarithmically) bounded with respect to x, since $f(y)$ is polynomially (logarithmically) bounded with respect to $g_1(x)$ and $g_2(x)$. Function f' is computable in polynomial time and the length of the function values $f'(g_1(x) * g_2(x) * y)$ is polynomially (logarithmically) bounded with respect to x, since $f(y)$ is polynomially (logarithmically) bounded with respect to $g_i(x)$ with $i \in \{1, 2\}$.

It remains to show that A' is indeed defined by B' and f', i.e.: $A'(x) = \{w \in \Sigma^* \cdot \{*\} \cdot \Sigma^* \cdot \{*\} \cdot \Gamma^* \mid B'(x, w) \wedge f'(w) = \mathrm{opt}_{f'}^{B'}(x)\}$.

The cases $A(g_1(x)) = \emptyset$ and $A(g_1(x)) = A(g_2(x))$ are easy. We only consider the case that both $A(g_1(x)) \neq \emptyset$ and $A(g_1(x)) \subset A(g_2(x))$ hold. By $A(g_1(x)) \neq \emptyset$, there exists some $y' \in \Gamma^*$ with $y' \in A(g_1(x))$ and, of course, also $y' \in A(g_2(x))$. Hence, $B(g_i(x), y')$ holds and also $f(y') = \mathrm{opt}_f^B(g_i(x))$ for $i \in \{1, 2\}$. Thus, in particular, $\mathrm{opt}_f^B(g_1(x)) = \mathrm{opt}_f^B(g_2(x))$, since both optimal values are identical to $f(y')$.

By the assumption $A(g_1(x)) \subset A(g_2(x))$, there also exists some $y'' \in \Gamma^*$ with $y'' \in A(g_2(x))$ and $y'' \notin A(g_1(x))$. By $y'' \in A(g_2(x))$ we know that $B(g_2(x), y'')$ holds and also $f(y'') = \mathrm{opt}_f^B(g_2(x))$. But then, since $y'' \notin A(g_1(x))$ and $\mathrm{opt}_f^B(g_1(x)) = \mathrm{opt}_f^B(g_2(x))$, we know that $B(g_1(x), y'')$ does not hold. Hence, by the definition of B' and f', also $\mathrm{opt}_{f'}^{B'}(x) = \mathrm{opt}_f^B(g_2(x))$ holds.

To conclude the proof, it suffices to show that for all strings $w = g_1(x) * g_2(x) * y$, the following equivalence holds: $B'(x, w) \wedge f'(w) = \mathrm{opt}_{f'}^{B'}(x)$ if and only if $y \in A(g_2(x)) \wedge y \notin A(g_1(x))$.

[$\Rightarrow$]: $B'(x, w)$ implies that $(g_2(x), y)$ belongs to B while $(g_1(x), y)$ does not. Hence, $y \notin A(g_1(x))$. On the other hand, since $f(y) = f'(w) = \mathrm{opt}_{f'}^{B'}(x) = \mathrm{opt}_f^B(g_2(x))$ holds, we have $y \in A(g_2(x))$.

[$\Leftarrow$]: Let $y \in A(g_2(x))$ and $y \notin A(g_1(x))$. By the former condition, we have $B(g_2(x), y)$ and $f(y) = \mathrm{opt}_f^B(g_2(x))$. Then also $f(y) = \mathrm{opt}_f^B(g_1(x))$ holds, by the equality $\mathrm{opt}_f^B(g_1(x)) = \mathrm{opt}_f^B(g_2(x))$. Hence, $y \notin A(g_1(x))$ implies that $(g_1(x), y)$ does not belong to B. Thus, (x, w) belongs to B' and $f'(w) = f(y) = \mathrm{opt}_f^B(g_2(x)) = \mathrm{opt}_{f'}^{B'}(x)$ holds. □

Our new considered classes $\#\cdot\mathrm{Opt}_k\mathrm{P}$ and $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$ need to be confronted with the already known counting hierarchy. We will present certain inclusions of the new classes with respect to already known counting complexity classes and show that the inclusions are proper, unless the polynomial hierarchy collapses.

Theorem 8 *The following inclusions hold for each $k \in \mathbb{N}$:*

$$\#\cdot\Pi_k\mathrm{P} \subseteq \#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n] \subseteq \#\cdot\mathrm{Opt}_{k+1}\mathrm{P} \subseteq \#\cdot\Pi_{k+1}\mathrm{P}$$

Proof. The inclusion $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n] \subseteq \#\cdot\mathrm{Opt}_k\mathrm{P}$ is clear since any output that fits into logarithmic space also fits into polynomial space.

[$\#\cdot\Pi_k\mathrm{P} \subseteq \#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$]: Let $\#\cdot A$ be a counting problem in $\#\cdot\Pi_k\mathrm{P}$. Since $\#\cdot\Pi_k\mathrm{P} = \#\cdot\mathrm{P}^{\Sigma_k\mathrm{P}}$ holds, there exists a nondeterministic polynomial-time Turing machine M with a $\Sigma_k\mathrm{P}$ oracle, such that $\#\cdot A(x)$ corresponds to the number of accepting paths of M on x. We transform M into a $\Sigma_k\mathrm{P}$-transducer M' by requesting M' to write the same output (say, the number 1) on every accepting path y. Hence, every accepting path of M' trivially writes the optimal value. Then $\#\cdot A$ can indeed be considered as the $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$-problem defined by M', i.e., $\#\cdot A(x)$ corresponds to the number of accepting paths of M' on x yielding the optimal value.

[$\#\cdot\mathrm{Opt}_{k+1}\mathrm{P} \subseteq \#\cdot\Pi_{k+1}\mathrm{P}$]: Let $\#\cdot A \in \#\cdot\mathrm{Opt}_{k+1}\mathrm{P}$. By Theorem 5, there exists a binary predicate B decidable by a $\Delta_{k+1}\mathrm{P}$ algorithm and a polynomial p, such that for each pair $(x, y) \in B$ we have $|y| \leq p(|x|)$, together with a polynomial-time computable function $f\colon \Gamma^* \to \mathbb{N}$, such that $A(x) = \{y \mid (x, y) \in B \wedge f(y) = \mathrm{opt}_f^B(x)\}$ and the length of $f(y)$ is polynomially bounded with respect to $|x|$. Without loss of generality, suppose that $\mathrm{opt}_f^B(x)$ is defined as the maximum. Then $\#\cdot A$ can also be defined as the counting problem $\#\cdot B'$ corresponding to the following predicate B', where $y \in B'(x)$ if and only if $(x, y) \in B$ and for all strings $y' \in \Gamma^*$ with $(x, y') \in B$ we have $f(y') \leq f(y)$. The latter condition can be tested in $\mathrm{coNP}^{\Delta_{k+1}\mathrm{P}}$. Since $(x, y) \in B$ is decidable in $\Delta_{k+1}\mathrm{P}$ and the equalities $\mathrm{coNP}^{\Delta_{k+1}\mathrm{P}} = \mathrm{coNP}^{\Sigma_k\mathrm{P}} = \Pi_{k+1}\mathrm{P}$ hold for each k, we have that $B' \in \Pi_{k+1}\mathrm{P}$. Hence, $\#\cdot B'$ is in $\#\cdot\Pi_{k+1}\mathrm{P}$. Moreover, the identity $\#\cdot B' = \#\cdot A$ holds, since the equality $A(x) = \{y \mid (x, y) \in B \wedge f(y) = \mathrm{opt}_f^B(x)\} = \{y \mid y \in B'(x)\}$ is satisfied. □

Finally we prove the robustness of the new classes.

Theorem 9 *If $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$ or $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}$ coincides with either $\#\cdot\Pi_k\mathrm{P}$ or $\#\cdot\Pi_{k+1}\mathrm{P}$ for some $k \in \mathbb{N}$, then the polynomial hierarchy collapses to the k-th or $(k+1)$-st level, respectively.*

Proof. Obviously, it suffices to separate the classes $\#\cdot\Pi_k\mathrm{P}$ from $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$ and $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}$ from $\#\cdot\Pi_{k+1}\mathrm{P}$.

Suppose that $\#\cdot\Pi_k\mathrm{P} = \#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$ holds. Consider a predicate A on $\Pi_k\mathrm{SAT}$ formulas φ containing truth assignments I. We define $I \in A(\varphi)$ if I is a cardinality-minimal model of φ. The counting problem $\#\cdot A$ is the familiar $\#$Min-Card-Π_kSAT problem. Following Theorem 6, $\#$Min-Card-Π_kSAT and therefore also $\#\cdot A$ belong to $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$. From the equality $\#\cdot\Pi_k\mathrm{P} = \#\cdot\mathrm{Opt}_{k+1}\mathrm{P}[\log n]$ it follows that $\#\cdot A$ is contained in $\#\cdot\Pi_k\mathrm{P}$. Hence, it can be tested by a $\Pi_k\mathrm{P}$ algorithm if $I \in A(\varphi)$ holds for a given formula φ. But then we get the following $\Pi_k\mathrm{P}$-decision procedure for the Min-Card–Π_kSAT problem, a generalization of Min-Card-SAT where we consider Π_kSAT formulas $\forall Y_1 \cdots Q Y_k\ \psi(X, Y_1, \ldots, Y_k)$ instead of propositional ones $\psi(X)$. Let an instance of Min-Card–Π_kSAT be given by a Π_kSAT formula $\varphi(X) = \forall Y_1 \cdots Q Y_k\ \psi(X, Y_1, \ldots, Y_k)$ and a subset of variables $X' \subseteq X$. We guess an assignment I to X, check that all variables X' are evaluated to true in I, and that I is a cardinality-minimal model of φ. The latter check is equivalent to checking if $I \in A(\varphi)$ holds, which can be done by a $\Pi_k\mathrm{P}$ algorithm. It can be easily proved that Min-Card–Π_kSAT is $\Theta_{k+1}\mathrm{P}$-complete by a generalization of the $\Theta_2\mathrm{P}$-completeness proof of Max-Card-SAT in [6] along the lines presented in [16,22]. This implies that $\Pi_k\mathrm{P} = \Theta_{k+1}\mathrm{P}$ holds, hence we have the inclusion $\Sigma_k\mathrm{P} \subseteq \mathrm{P}^{\Sigma_k\mathrm{P}[1]} \subseteq \Delta_{k+1}\mathrm{P}[\log n] = \Pi_k\mathrm{P}$, and therefore the polynomial hierarchy collapses to the k-th level.

Suppose now that $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P} = \#\cdot\Pi_{k+1}\mathrm{P}$ holds. From [3] we know that the generic counting problem $\#\Pi_{k+1}\mathrm{SAT}$ belongs to $\#\cdot\Pi_{k+1}\mathrm{P}$ (in fact, the problem is $\#\cdot\Pi_{k+1}\mathrm{P}$-complete). From the identity $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P} = \#\cdot\Pi_{k+1}\mathrm{P}$ it follows that the problem $\#\Pi_{k+1}\mathrm{SAT}$ also belongs to $\#\cdot\mathrm{Opt}_{k+1}\mathrm{P}$. Hence, there exists a $\Sigma_{k+1}\mathrm{P}$-transducer M, such that for every $\#\Pi_{k+1}\mathrm{SAT}$-formula

$$\psi(X) \quad = \quad \forall Y_1 \exists Y_2 \cdots Q Y_{k+1}\ \ \varphi(X, Y_1, \ldots, Y_{k+1})$$

the number of satisfying assignments of $\psi(X)$ corresponds to the number of accepting paths of M on input $\psi(X)$ yielding the optimal value. But then we get a $\Sigma_{k+1}\mathrm{P}$-decision procedure for QSAT_{k+2} as follows. Let $\chi = \exists X \psi(X)$ be an instance of QSAT_{k+2}, where $\psi(X)$ is defined as before. Then χ is satisfiable if and only if $\psi(X)$ has at least one satisfying assignment if and only if the computation of M on input $\psi(X)$ has at least one accepting path. There exists an accepting path if and only if there exists an accepting path yielding the optimal value. From $\Sigma_{k+2}\mathrm{P}$-completeness of QSAT_{k+2} it follows that $\Sigma_{k+1}\mathrm{P} = \Sigma_{k+2}\mathrm{P}$, collapsing the polynomial hierarchy to the $(k+1)$-st level. □

4 Further Optimization Counting Problems

The most interesting optimization counting problems are of course those belonging to the classes on the first level of the optimization counting hierarchy, namely $\#\cdot\mathrm{OptP}$ and $\#\cdot\mathrm{OptP}[\log n]$. In this section we will focus on such problems of particular interest.

Gasarch *et al.* presented in [6] a plethora of optimization problems complete for OptP and $\mathrm{OptP}[\log n]$. Either their lower bound is already proved by a parsimonious reduction or the presented reduction can be transformed into a parsimonious one similarly to Galil's construction in [5]. The counting version of virtually all these problems can therefore be proved to be complete for $\#\cdot\mathrm{OptP}$ or $\#\cdot\mathrm{OptP}[\log n]$. Likewise, Krentel presented in [12] several problems belonging to higher levels of the optimization hierarchy. They give rise to counting problems complete for $\#\cdot\mathrm{Opt}_k\mathrm{P}$ or $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$ with $k > 1$.

Let us investigate the following optimization variant of the usual counting problem related to satisfiability of propositional formulas.

Problem: #Min-Card-SAT
Input: A propositional formula φ in conjunctive normal form over the variables X.
Output: Number of models of φ with minimal Hamming weight.

The dual problem #Max-Card-SAT asks for the number of models with *maximal* Hamming weight. The problems #Min-Weight-SAT and #Max-Weight-SAT are the corresponding weighted versions of the aforementioned problems.

Following Theorem 6, both counting problems #Min-Card-SAT and #Max-Card-SAT are $\#\cdot\mathrm{OptP}[\log n]$-complete, whereas #Min-Weight-SAT and #Max-Weight-SAT are $\#\cdot\mathrm{OptP}$-complete. We consider only the cardinality-minimal problems in the sequel.

It is also interesting to investigate special cases of the optimization counting problems involving the following restrictions on the formula φ. As usual, a literal is a propositional variable (positive literal) or its negation (negative literal), whereas a clause is a disjunction of literals, and a formula in conjunctive normal form is a conjunction of clauses. We say that a clause c is

Horn	if	it contains at most one positive literal,
dual Horn	if	it contains at most one negative literal,
Krom	if	it contains at most two literals.

A formula $\varphi = c_1 \wedge \cdots \wedge c_n$ in conjunctive normal form is Horn, dual Horn, or Krom if all clauses c_i for $i = 1, \ldots, n$ satisfy the respective condition. Formulas restricted to conjunctions of Horn, dual Horn, or Krom clauses are often investigated in computational problems related to artificial intelligence, in particular to closed world reasoning [1]. We denote by the specification in brackets the restriction of the counting problem #Min-Card-SAT to the respective class of formulas.

The models of Horn formulas are closed under conjunction, i.e., for two models m and m' of a Horn formula φ, also the Boolean vector $m \wedge m' = (m[1] \wedge m'[1], \ldots, m[k] \wedge m'[k])$ is a model of φ. Hence there exists a unique model with minimal Hamming weight if and only if φ is satisfiable. Therefore a Horn formula φ has either one cardinality-minimal model or none, depending on the satisfiability of φ. A similar situation arises for #Min-Card-DNF, the problem of counting the number of assignments with minimal Hamming weight to a propositional formula in disjunctive normal form. These considerations imply the following results.

Proposition 10 *#Min-Card-SAT[horn] and #Min-Card-DNF are in* FP.

Vertex covers, cliques, and independent sets have a particular relationship. The set X is a smallest vertex cover in $G = (V, E)$ if and only if $V \setminus X$ is a largest independent set in G if and only if $V \setminus X$ is a largest clique in the complement graph $\bar{G} = (V, V \times V \setminus E)$. The size of the largest clique has been investigated by Krentel [11] and proved to be $\mathrm{OptP}[\log n]$-complete. (the same proof is also given in [14]). Using this knowledge, we can determine the complexity of the following problems.

Problem: #max-card-independent set
Input: Graph $G = (V, E)$.
Output: Number of independent sets in G with maximum cardinality, i.e., number of subsets $V' \subseteq V$ where $|V'|$ is maximal and for all $u, v \in V'$ we have $(u, v) \notin E$.

Problem: #MAX-CARD-CLIQUE
Input: Graph $G = (V, E)$.
Output: Number of cliques in G with maximum cardinality, i.e., number of subsets $V' \subseteq V$ where $|V'|$ is maximal and $(u, v) \in E$ holds for all $u, v \in V'$ such that $u \neq v$.

Problem: #MIN-CARD-VERTEX COVER
Input: Graph $G = (V, E)$.
Output: Number of vertex covers of G with minimal cardinality, i.e., number of subsets $V' \subseteq V$ where $|V'|$ is minimal and $(u, v) \in E$ implies $u \in V'$ or $v \in V'$.

Theorem 11 *The problems* #MAX-CARD-INDEPENDENT SET*,* #MAX-CARD-CLIQUE*, and* #MIN-CARD-VERTEX COVER *are* $\#\cdot\mathrm{OptP}[\log n]$*-complete. Their weighted versions are* $\#\cdot\mathrm{OptP}$*-complete.*

Proof. The membership in the appropriate classes is straightforward. The lower bound is proved by the usual reduction to #MAX-CARD-INDEPENDENT SET, from #MAX-CARD-SAT, as it is mentioned in the proof of Theorem 17.6 in [14] or in [11]. This reduction is parsimonious and extends by the usual construction to cliques. Hence, there exists a parsimonious reduction from #MIN-CARD-SAT, using Proposition 3, to all three problems. The weighted version is a reduction from #MIN-WEIGHT-SAT. □

We can easily transform the counting problem #MIN-CARD-VERTEX COVER to both #MIN-CARD-SAT[DUAL HORN] and #MIN-CARD-SAT[KROM]. Indeed, we can represent an edge $(u, v) \in E$ of a graph $G = (V, E)$ by a clause $(u \vee v)$ which is both Krom and dual Horn. Hence a cardinality-minimal vertex cover of a graph $G = (V, E)$ corresponds to a cardinality-minimal model of a corresponding formula $\varphi_G = \bigwedge_{(u,v)\in E}(u \vee v)$.

Corollary 12 *The counting problems* #MIN-CARD-SAT[DUAL HORN] *and* #MIN-CARD-SAT[KROM] *are* $\#\cdot\mathrm{OptP}[\log n]$*-complete via parsimonious reductions.*

The following problem is a classic in optimization theory. It is usually formulated as the maximal number of clauses that can be satisfied. We can also ask for the number of truth assignments that satisfy the maximal number of clauses.

Problem: #MAX2SAT
Input: A propositional formula φ in conjunctive normal form over the variables X with at most two variables per clause.
Output: Number of assignments to φ that satisfy the maximal number of clauses.

The optimization variant of the following counting problem is presented in [6] under the name CHEATING SAT. We can interpret it as a satisfiability problem in a 3-valued logic, where the middle value τ is a "don't-know". In this setting it is interesting to investigate the minimal size of uncertainty we need to satisfy a formula for the optimization variant, as well as the number of satisfying assignments with the minimal size of uncertainty.

Problem: #MIN-SIZE UNCERTAINTY SAT
Input: A propositional formula φ in conjunctive normal form over the variables X.
Output: Number of satisfying assignments $m\colon X \to \{0, \tau, 1\}$ of the formula φ, where $m(x) = \tau$ satisfies both literals x and $\neg x$, with minimal cardinality of the set $\{x \in X \mid m(x) = \tau\}$.

Theorem 13 #MAX2SAT *and* #MIN-SIZE UNCERTAINTY SAT *are* $\#\cdot\mathrm{OptP}[\log n]$*-complete.*

Proof. The membership in $\#\cdot\mathrm{OptP}[\log n]$ is clear for all three problems from the fact that witness testing belongs to $\mathrm{OptP}[\log n]$, since the corresponding optimization variants of the problems are $\mathrm{OptP}[\log n]$-complete [6]. For the lower bound, the reductions presented in [6] are parsimonious, therefore they also represent the appropriate reductions for the counting variants. □

Even though the complete problems for the classes $\#\cdot\mathrm{OptP}$ and $\#\cdot\mathrm{OptP}[\log n]$ are the most interesting ones, there also exist some interesting complete problems in the classes $\#\cdot\mathrm{Opt}_k\mathrm{P}$ and $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$ for $k > 1$. The following problem is an example of such a case.

Problem: #MAXIMUM k-QUANTIFIED CIRCUIT
Input: A Boolean circuit $C(\vec{x}, \vec{y}_1, \ldots, \vec{y}_k)$ over variable vectors $\vec{x}$, $\vec{y}_1$, $\ldots$, $\vec{y}_k$.
Output: Number of maximum values $\vec{x} \in \{0,1\}^n$ in binary notation satisfying the quantified expression $\forall\vec{y}_1\exists\vec{y}_2 \cdots Q\vec{y}_k\ (C(\vec{x}, \vec{y}_1, \ldots, \vec{y}_k) = 1)$, where Q is either $\forall$ or $\exists$ depending on the parity of k.

Theorem 14 #MAXIMUM k-QUANTIFIED CIRCUIT *is* $\#\cdot\mathrm{Opt}_k\mathrm{P}$*-complete.*

The proof of Theorem 14 follows that of Theorem 4.2 from [12].

5 Concluding Remarks

In the scope of the result from [18] showing that all classes between #P and #PH, the counting equivalent of the polynomial hierarchy, collapse to #P under 1-Turing reductions, it is necessary (1) to find suitable reductions strong enough to prove completeness of well-known counting problems, but tame enough to preserve at least some counting classes, (2) to identify counting classes with interesting complete problems preserved under the aforementioned reduction. The first problem was mainly addressed in [3], whereas in this paper we focused on the second point. We introduced a new hierarchy of optimization counting complexity classes $\#\cdot\mathrm{Opt}_k\mathrm{P}$ and $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$. These classes allowed us to pinpoint the complexity of many natural optimization counting problems which had previously resisted a precise classification. Moreover, we have shown that these new complexity classes have several desirable properties and they interact well with the counting hierarchy defined by Hemaspaandra and Vollmer in [7]. Nevertheless, the Hemaspaandra-Vollmer counting hierarchy does not seem to be sufficiently detailed to capture all interesting counting problems. Therefore an even more fine-grained stratification of the counting complexity classes is necessary, which started with the contribution of Pagourtzis and Zachos [13] and has been pursued in this paper.

Finally, further decision problems in $\Delta_k\mathrm{P}$ (respectively $\Theta_k\mathrm{P}$) with $k \in \mathbb{N}$ and corresponding counting problems should be inspected. It should be investigated if the complexity of the latter can be precisely identified now that we have the new counting complexity classes $\#\cdot\mathrm{Opt}_k\mathrm{P}$ (respectively $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$) at hand. Moreover, we would also like to find out more about the nature of the problems that are complete for these new counting complexity classes. In particular, it would be very interesting to find out if there also exist "easy to decide, hard to count" problems, i.e., problems whose counting variant is complete for $\#\cdot\mathrm{Opt}_k\mathrm{P}$ (respectively $\#\cdot\mathrm{Opt}_k\mathrm{P}[\log n]$) while the corresponding decision problem is below $\Delta_k\mathrm{P}$ (respectively $\Theta_k\mathrm{P}$). Clearly, such a phenomenon can

only exist if we consider completeness with respect to reductions stronger than the parsimonious ones. Hence, the closure of our new counting classes under *subtractive reductions* (rather than just under parsimonious reductions) in Theorem 7 is an indispensable prerequisite for further research in this direction.

Acknowledgment: We thank Arnaud Durand and Yann Strozecki for their remarks and for the discussion on the proof of Theorem 9.

References

[1] M. Cadoli and M. Lenzerini. The complexity of propositional closed world reasoning and circumscription. *Journal of Computer and System Sciences*, 48(2):255–310, 1994.

[2] N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125(1):1–12, 1996.

[3] A. Durand, M. Hermann, and P. G. Kolaitis. Subtractive reductions and complete problems for counting complexity classes. *Theoretical Computer Science*, 340(3):496–513, 2005.

[4] L. Fortnow. Counting complexity. In L. A. Hemaspaandra and A. L. Selman, editors, *Complexity theory retrospective II*, pages 81–107. Springer-Verlag, 1997.

[5] Z. Galil. On some direct encodings of nondeterministic Turing machines operating in polynomial time into P-complete problems. *SIGACT News*, 6(1):19–24, January 1974.

[6] W. I. Gasarch, M. W. Krentel, and K. J. Rappoport. OptP as the normal behavior of NP-complete problems. *Mathematical Systems Theory*, 28(6):487–514, 1995.

[7] L. A. Hemaspaandra and H. Vollmer. The satanic notations: Counting classes beyond #P and other definitional adventures. *SIGACT News, Complexity Theory Column 8*, 26(1):2–13, March 1995.

[8] B. Jenner and J. Torán. Computing functions with parallel queries to NP. *Theoretical Computer Science*, 141(1-2):175–193, 1995.

[9] J. Köbler, U. Schöning, and J. Torán. On counting and approximation. *Acta Informatica*, 26(4):363–379, 1989.

[10] D. C. Kozen. *The design and analysis of algorithms*, chapter 26: Counting problems and #P, pages 138–143. Springer-Verlag, 1992.

[11] M. W. Krentel. The complexity of optimization problems. *Journal of Computer and System Sciences*, 36(3):490–509, 1988.

[12] M. W. Krentel. Generalizations of OptP to the polynomial hierarchy. *Theoretical Computer Science*, 97(2):183–198, 1992.

[13] A. Pagourtzis and S. Zachos. The complexity of counting functions with easy decision version. In R. Královič and P. Urzyczyn, editors, *Proceedings 31st International Symposium on Mathematical Foundations of Computer Science (MFCS 2006), Stará Lesná (Slovakia)*, volume 4162 of *Lecture Notes in Computer Science*, pages 741–752. Springer-Verlag, 2006.

[14] C. H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[15] A. Selman, X. Mei-Rui, and R. Book. Positive relativizations of complexity classes. *SIAM Journal on Computing*, 12(3):565–579, 1983.

[16] L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.

[17] S. Toda and M. Ogiwara. Counting classes are at least as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 21(2):316–328, 1992.

[18] S. Toda and O. Watanabe. Polynomial-time 1-Turing reductions from #PH to #P. *Theoretical Computer Science*, 100(1):205–221, 1992.

[19] L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.

[20] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[21] K. Wagner. Bounded query classes. *SIAM Journal on Computing*, 19(5):833–846, 1990.

[22] C. Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):23–33, 1976.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/