An Isomorphism Between Subexponential and Parameterized Complexity Theory

Yijia Chen
Shanghai Jiaotong University

Martin Grohe
Humboldt-Universität zu Berlin

March 29, 2007

Abstract

We establish a close connection between (sub)exponential time complexity and parameterized complexity by proving that the so-called miniaturization mapping is a reduction preserving isomorphism between the two theories.

1 Introduction

The area of exact algorithms for hard algorithmic problem has received considerable attention in recent years (see, e.g., [19, 20]). The goal is to design exact, as opposed to approximative, algorithms for hard (usually NP-complete) problems that are better than the trivial brute force algorithms, but may still be exponential. For example, the currently best deterministic algorithm for the 3-satisfiability problem (3-SAT) due to Brueggemann and Kern [3] has a running time of $O(1.473^n)$, and the best randomized algorithm due to Rolf [18] has a running time of $O(1.3222^n)$. Here n denotes the number of variables of the input formula.

Exponential complexity theory

A more qualitative question that has been recognized as central for the emerging theory of "exponential time complexity" is whether 3-SAT can be solved in time $2^{o(n)}$. The hypothesis that this is not possible is known as the *exponential time hypothesis (*ETH*)*. It has first been studied systematically by Impagliazzo, Paturi, and Zane [17], who proved that the hypothesis is very robust and equivalent to analogous hypotheses for many other NP-complete problems. For example, ETH is equivalent to the question whether the INDEPENDENT-SET problem can be solved in time $2^{o(n)}$, where n denotes the number of vertices of the input graph. The reader may wonder why the running time is measured in terms of the number of variables and number of vertices, respectively, and not in the actual input size. The main contribution of Impagliazzo et al. is to prove that the ETH is independent of the "size measure" (number of variables or actual size), that is, 3-SAT is solvable in time $2^{o(n)}$ if and only if it is solvable in time $2^{o(m)}$ for the input size m.[1] It is easy to see that this implies the corresponding result for INDEPENDENT-SET and a number other further problems. However, in general subexponential solvability is very sensitive with respect to the size measure, as the trivial example of the CLIQUE problem shows: Clearly, CLIQUE is solvable in time $2^{o(n)}$ if and only if INDEPENDENT-SET is, but CLIQUE is trivially solvable in time $n^{O(\sqrt{m})} = 2^{o(m)}$, where m denotes the size of the input graph.

Authors' addresses: Yijia Chen, BASICS, Department of Computer Science, Shanghai Jiaotong University, Shanghai 200030, China. Email: yijia.chen@cs.sjtu.edu.cn
Martin Grohe, Institut für Informatik, Humboldt-Universität, Unter den Linden 6, 10099 Berlin, Germany.
Email: grohe@informatik.hu-berlin.de

[1] Let us remark that here we assume that 3-CNF formulas have no repeated clauses and hence that $m = O(n^3)$. Otherwise the input size would not be bounded in terms of n and hence the problem would not be solvable in time $f(n)$ for any function f. Later, this problem will disappear because we will always admit a polynomial of the input size as a factor of the running time.

The example of the CLIQUE problem illustrates that it is reasonable for a theory of exponential time complexity to view problems as pairs (P,ν) consisting of a decision problem $P \subseteq \Sigma^*$ over some finite alphabet Σ and a mapping $\nu : \Sigma^* \to \mathbb{N}$, which may be viewed as the *size measure*. For technical reasons, the mapping ν is required to be polynomial time computable. Typical size measures are the number of vertices or the number of edges for graph problems, or the number of variables for satisfiability problems. Of course there is always the trivial size measure $\nu(x) = |x|$. At first sight, it seems reasonable to also require a size measure ν to be polynomially related to the input length, that is, $|x|^c \leq \nu(x) \leq |x|^d$ for some $c,d > 0$ and all $x \in \Sigma^*$. However, we prefer *not* to make this additional requirement, because there are problems where very natural "size measures" do not fulfill it. The most important example is the satisfiability problem SAT for arbitrary CNF formulas. The number of variables still seems one of the most natural complexity parameters, but it may be exponentially smaller than the input length. Hypergraph problems provide further natural examples; here the number of vertices is a natural parameter. Of course for such examples it is no longer appropriate to call ν a "size measure", but other than that the role of ν remains the same.

But what would it mean for SAT to be subexponential with respect to "number of variables?" Clearly, SAT is not solvable in time $2^{o(n)}$, because the size m of the input formula may be $2^{\Omega(n)}$. The natural question is whether SAT is solvable in time $2^{o(n)} \cdot m^{O(1)}$. More generally, we say that a problem (P,ν) is *subexponential* if it is solvable in time

$$2^{o(\nu(x))} \cdot |x|^{O(1)} \tag{1.1}$$

for every instance x. Of course, if ν is polynomially related to the input length, then the term $|x|^{O(1)}$ is dominated by $2^{o(\nu(x))}$ and can hence be omitted. We denote the class of all subexponential problems by SUBEPT. We are interested in the dividing line between exponential and subexponential solvability; the problems we consider are usually (trivially) solvable in time $2^{O(\nu(x))} \cdot |x|^{O(1)}$. Let us denote the class of all these problems by EPT.[2] The main advantage of our "parameterized" approach to exponential complexity is that it makes it easier to compare problems that otherwise would not be by "rescaling" them. For example, PLANAR-INDEPENDENT-SET is solvable in time $2^{O(\sqrt{n})}$, and the question of whether it is solvable in time $2^{o(\sqrt{n})}$ is equivalent to the question of whether INDEPENDENT-SET is solvable in time $2^{o(n)}$. If we rescale PLANAR-INDEPENDENT-SET by letting $\nu(n) = \sqrt{n}$, we actually obtain a problem that is equivalent to INDEPENDENT-SET with the size measure "number of variables" under suitable reductions. Thus by specifying size measures, we get a more robust theory.

To develop a complexity theory, we need suitable reductions. Impagliazzo et al. [17] introduced so called *subexponential reduction families*, which are a form of Turing reductions that preserve subexponential solvability. We essentially work with these reductions, and also with a corresponding notion of many-one reductions.

The goals of *exponential complexity theory* may now be stated as classifying concrete problems within this framework and investigating the structure of the resulting complexity classes.

Parameterized complexity theory

The reader familiar with parameterized complexity theory will have noticed that we are dealing with exactly the type of problems as considered there. A *parameterized problem* is a pair (Q,κ), where $Q \subseteq \Sigma^*$ for some finite alphabet Σ and $\kappa : \Sigma^* \to \mathbb{N}$, the *parameterization*, is polynomial time computable. We usually denote parameterized problems in the "exponential world" by (P,ν) and problems in the "parameterized world" by (Q,κ) because it will be important for us to separate the two "worlds", but formally both are the same. However, the questions asked in parameterized complexity theory are different; parameterized complexity theory's main intention is to address complexity issues in situations where the parameter is expected to be small compared to the input size, whereas in the exponential theory the parameter was introduced as a size measure. A problem (Q,κ) is *fixed-parameter tractable* if it can be solved in time $f(\kappa(x)) \cdot |x|^{O(1)}$, where f is an arbitrary computable function. The class of all fixed-parameter tractable problems is denoted by FPT. There are suitable notions of *fpt (many-one) reduction* and *fpt Turing reduction*.

As opposed to exponential complexity theory, parameterized complexity theory has been developed in depth over the last fifteen years. A rich and maybe a bit unwieldy structure has emerged. The most important parameterized complexity classes are those of the *W-hierarchy*. Most natural parameterized

[2] This terminology has been introduced in [15] in the context of parameterized complexity theory, it will be explained in Section 2.

problems are complete for one of these classes. Parameterized complexity theory almost exclusively studies problems which are in the class XP of all problems that can be solved in polynomial time for every fixed parameter value.

Our results

Our main result is that exponential and parameterized complexity theory are isomorphic. Let us explain what we mean by this: On the exponential side, we consider the partial order induced by subexponential reduction families on the *degrees*, that is, equivalence classes under subexponential reduction families. On the parameterized side, we consider the partial order induced by fpt-reductions on the corresponding degrees. We define a mapping, the so-called *miniaturization mapping*, that associates a problem (Q,κ) with every problem (P,ν) and prove that this mapping is an isomorphism between the partial order of degrees inside EPT under subexponential reduction families and the partial order of degrees inside XP under fpt-reductions. This result holds for both many-one and Turing reductions. In particular, miniaturization maps the class SUBEPT (the lowest "exponential degree") to FPT (the lowest "parameterized degree") and EPT to XP.

If we look at degrees outside of EPT, then the miniaturization mapping is still an embedding, but we prove that it is no longer onto. That is, there are parameterized degrees outside XP that are not in the image of the miniaturization mapping. Technically, this is our most difficult result.

The precise technical statement of our results requires some additional uniformity conditions. Essentially, the "little-oh" in (1.1) has to be interpreted "effectively" (see 2 for a precise statement). Alternatively, the uniformity condition in the definition of fixed-parameter tractability, requiring f to be computable, can be relaxed.

The miniaturization mapping is a fairly natural mapping between parameterized problems that has been studied before [4, 7, 9], albeit not as an abstract mapping between parameterized problems, but just as a transformation between concrete problems such as vertex cover. As a matter of fact, there already is a body of work in parameterized complexity theory, starting with Abrahamson, Downey, and Fellows [1], that studies the relation between fixed-parameter tractability and exponential time complexity [4, 5, 6, 7, 9, 13]. Notably, it follows from this work that the miniaturization mapping maps the degree of 3-SAT with the "number of variables" size measure to a class $\mathrm{M}[1]$ between FPT and the first level $\mathrm{W}[1]$ of the W-hierarchy. The degree of SAT is mapped to a class $\mathrm{M}[2]$ between $\mathrm{W}[1]$ and $\mathrm{W}[2]$, and the degree of the satisfiability problem for Boolean circuits under the "number of input nodes" size measure is mapped to the class W[P]. This shows that the miniaturization isomorphism is not just an abstract mapping between partial orders, but actually is a meaningful mapping between concrete problems and complexity classes. In the last section of this paper, we analyse the correspondence between the W-hierarchy and a natural hierarchy in the exponential theory, and we determine the pre-image of the W-hierarchy under the miniaturization mapping.

2 Preliminaries

In this section we give the necessary background of exponential and parameterized complexity. For more comprehensive details the reader is referred to [12, 14, 17].

We use $\mathbb{N}$ to denote the set of natural numbers (positive integers). For $m \leq n \in \mathbb{N}$ we let $[m,n] := \{m, m+1, \ldots, n\}$ and $[n] := [1,n]$. A *classical decision problem* is a set $P \subseteq \Sigma^*$, where Σ is a finite alphabet. P is *trivial* if $P = \emptyset$ or $P = \Sigma^*$ and *nontrivial* otherwise. A *parameterized problem* is a pair (Q,κ), where $Q \subseteq \Sigma^*$ for some finite alphabet Σ and $\kappa : \Sigma^* \to \mathbb{N}$ is polynomial time computable. The mapping κ is called the *parameterization*. We occasionally write $x \in (Q,\kappa)$ instead of $x \in P$. A parameterized problem (Q,κ) is *trivial* if Q is and nontrivial otherwise.

Exponential complexity

Recall that in the exponential theory we study parameterized problems (P,ν) and view the parameterization ν as a *size measure*. The most obvious size measure for a problem $P \subseteq \Sigma^*$ is the *length of the input* $\nu(x) = |x|$. Unfortunately, for most natural problems, the length of the input is not exactly what we think of as its "size". For example, we are used to thinking of the size of a graph G with n vertices and m edges

as being $(m+n)$ rather than $\Theta(n+m\cdot\log n)$, which would be the length of a reasonable encoding of G over a finite alphabet. To abstract from such a heavy dependence on coding issues, we define the *size* $||G||$ of a graph G with n vertices and m edges to $m+n$. Hence we distinguish between "size" and "length (of an encoding)" of a graph. Similarly, we define the size $||C||$ of a Boolean circuit to be the number of gates plus the number of lines. We view Boolean formulas as special circuits, so they inherit the size measures for circuits. For example, for a CNF-formula $\alpha = \bigwedge_{i=1}^{m}\bigvee_{j=1}^{k_i}\lambda_{ij}$ this means that $||\alpha|| = \Theta(\sum_{i=1}^{m} k_i)$. We denote the parameterization of a graph problem or satisfiability problem P by the input size by *s-P*. For example, we let

s-SAT	
Instance:	A CNF-formula α.
Parameter:	$\|\|\alpha\|\|$.
Problem:	Decide whether α is satisfiable.

Other natural size measures are the "number of vertices" for graph problems and the "number of variables" for satisfiability problems. As examples, consider the following problems:

s-vert-INDEPENDENT-SET	
Instance:	A graph $G=(V,E)$ and a $\ell\in\mathbb{N}$.
Parameter:	$\|V\|$.
Problem:	Decides whether G has an independent set of cardinality ℓ.

s-var-3-SAT	
Instance:	A 3-CNF-formula α.
Parameter:	Number of variables of α.
Problem:	Decide whether α is satisfiable.

Similarly, we can define the graph problems *s-vert*-CLIQUE, *s-vert*-VERTEX-COVER, *s-vert*-DOMINATING-SET and the satisfiability problems *s-var*-SAT (satisfiability of CNF-formulas), *s-var*-CIRCUIT-SAT (satisfiability of Boolean circuits, parameterized by the number of input gates).

For two functions f,g with range $\mathbb{N}$, we say f is *effectively little-oh* of g and write $f\in o^{\text{eff}}(g)$, if there is a computable function ι that is *nondecreasing* and *unbounded* such that

$$f = O\left(\frac{g}{\iota}\right).$$

Definition 1. A parameterized problem (P,ν) is *subexponentially solvable* if there is an algorithm $\mathbb{A}$ that for every instance x decides whether $x\in P$ in time

$$2^{o^{\text{eff}}(\nu(x))}\cdot|x|^{O(1)}.$$

By $2^{o^{\text{eff}}(\nu(x))}$ we mean $2^{f(x)}$ for some function $f\in o^{\text{eff}}(\nu)$. SUBEPT denotes the class of all subexponentially solvable problems.

As mentioned in the introduction, an example of a problem in SUBEPT is *s*-CLIQUE, the clique problem parameterized by the input size, which is solvable in time $2^{O(\sqrt{m}\cdot\log m)}$, where m denotes the size of the input graph. Other, less trivial examples of problems in SUBEPT are the planar restrictions of many standard optimization problems parameterized by the number of vertices (or the size, which is equivalent for planar graphs). For example, the problems *s-vert*-PLANAR-VERTEX-COVER, *s-vert*-PLANAR-INDEPENDENT-SET, and *s-vert*-PLANAR-DOMINATING-SET are all solvable in time $2^{O(\sqrt{n})}$ and hence in SUBEPT [2]. However, most natural NP-complete problems do not seem to be in SUBEPT. To establish a completeness theory giving evidence to claims of problems not being in SUBEPT, we need an appropriate notion of reduction. The following lemma offers an alternative characterization of SUBEPT, which is the basis of the reductions we shall introduce afterwards.

Lemma 2. *Let (P,ν) be a parameterized problem over the alphabet Σ. The following are equivalent:*

(1) $(P,\nu) \in$ SUBEPT.

(2) There is an algorithm $\mathbb{A}$ expecting inputs from $\Sigma^* \times \mathbb{N}$ and a computable function f such that for all $(x,\ell) \in \Sigma^* \times \mathbb{N}$ the algorithm $\mathbb{A}$ decides if $x \in P$ in time $f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}$.

(3) There is an algorithm $\mathbb{A}$ expecting inputs from $\Sigma^* \times \mathbb{N}$, a computable function f, and a constant $c \in \mathbb{N}$, such that for all $(x,\ell) \in \Sigma^* \times \mathbb{N}$ the algorithm $\mathbb{A}$ decides if $x \in P$ in time $f(\ell) \cdot 2^{c \cdot \nu(x)/\ell} \cdot |x|^{O(1)}$.

Proof. $(1) \Rightarrow (2)$: Assume $(P,\nu) \in$ SUBEPT. Let ι be a computable function that is nondecreasing and unbounded, and let $\mathbb{A}$ be an algorithm deciding $x \in P$ in time $2^{c \cdot \nu(x)/\iota(\nu(x))} \cdot |x|^{O(1)}$ for some constant $c \in \mathbb{N}$. For $\ell \in \mathbb{N}$, let $n(\ell) := \max\big(\{n \mid \iota(n) < c \cdot \ell\} \cup \{1\}\big)$ and $f(\ell) := 2^{c \cdot n(\ell)}$. Then for all $(x,\ell) \in \Sigma^* \times \mathbb{N}$ we have $2^{c \cdot \nu(x)/\iota(\nu(x))} \cdot |x|^{O(1)} \leq f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}$. Let $\mathbb{A}'$ be the algorithm that, given $(x,\ell) \in \Sigma^* \times \mathbb{N}$, simply ignores ℓ and simulates $\mathbb{A}$ on input x. Then $\mathbb{A}'$ and f satisfy the conditions of (2).
The direction from (2) to (3) is trivial. We turn to $(3) \Rightarrow (1)$: Let $f : \mathbb{N} \to \mathbb{N}$ be a computable function, $c \in \mathbb{N}$ a constant, and $\mathbb{A}$ an algorithm that, given $(x,\ell) \in \Sigma^* \times \mathbb{N}$, decides if $x \in P$ in time $f(\ell) \cdot 2^{c \cdot \nu(x)/\ell} \cdot |x|^{O(1)}$. Without loss of generality, we may assume f is increasing and time-constructible. Let $\iota : \mathbb{N} \to \mathbb{N}$ be the following computable function: $\iota(n) := \max\big(\{\ell \mid f(\ell) < n\} \cup \{1\}\big)$, which is clearly nondecreasing, unbounded, and computable in time polynomial in n. Let $\mathbb{A}'$ be the following algorithm for deciding P: Given $x \in \Sigma^*$, first compute $n := \nu(x)$ and $\ell := \iota(n)$, and then simulate $\mathbb{A}$ on (x,ℓ). The running time of $\mathbb{A}$ is bounded by

$$\begin{aligned} & |x|^{O(1)} + n^{O(1)} + f(\iota(n)) \cdot 2^{c \cdot n/\iota(n)} \cdot |x|^{O(1)} \\ \leq\ & |x|^{O(1)} + n^{O(1)} + O(n) \cdot 2^{o^{\text{eff}}(n)} \cdot |x|^{O(1)} \\ =\ & 2^{o^{\text{eff}}(n)} \cdot |x|^{O(1)}. \end{aligned}$$

□

Our notion of reduction is essentially that of *subexponential reduction families*, as introduced in [17]. The reduction families in [17] are Turing reductions; we also introduce a many-one version.

Definition 3. Let (P,ν) and (P',ν') be parameterized problems over the alphabets Σ and Σ', respectively.

(1) A *subexponential reduction family*, or simply *serf reduction*, from (P,ν) to (P',ν') is a mapping $R : \Sigma^* \times \mathbb{N} \to (\Sigma')^*$, such that:

  (a) For all $(x,\ell) \in \Sigma^* \times \mathbb{N}$ we have $\big(x \in P \iff R(x,\ell) \in P'\big)$.

  (b) Given a pair $(x,\ell) \in \Sigma^* \times \mathbb{N}$, the image $R(x,\ell)$ is computable in time

$$f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}$$

  for some computable function $f : \mathbb{N} \to \mathbb{N}$.

  (c) There is a computable function $g : \mathbb{N} \to \mathbb{N}$ such that

$$\nu'(R(x,\ell)) \leq g(\ell) \cdot (\nu(x) + \log|x|)$$

  for all $(x,\ell) \in \Sigma^* \times \mathbb{N}$.

(2) A *subexponential Turing reduction family*, or *serf Turing reduction*, from (P,ν) to (P',ν') is an algorithm $\mathbb{A}$ with an oracle to P' such that there are computable functions $f,g : \mathbb{N} \to \mathbb{N}$ with:

  (a) Given a pair $(x,\ell) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathbb{A}$ decides if $x \in P$ in time

$$f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}.$$

(b) For all oracle queries "$y \in P'$?" posed by $\mathbb{A}$ on input $(x,\ell) \in \Sigma^* \times \mathbb{N}$ it holds that

$$\nu'(y) \leq g(\ell) \cdot (\nu(x) + \log|x|).$$

We write $(P,\nu) \leq^{\text{serf}} (P',\nu')$ (or $(P,\nu) \leq^{\text{serf-T}} (P',\nu')$) if there is a serf reduction (serf Turing reduction, respectively) from (P,ν) to (P',ν'). We write $(P,\nu) \equiv^{\text{serf}} (P',\nu')$ if $(P,\nu) \leq^{\text{serf}} (P',\nu')$ and $(P',\nu') \leq^{\text{serf}} (P,\nu)$.

It is clear that $(P,\nu) \leq^{\text{serf}} (P',\nu')$ implies $(P,\nu) \leq^{\text{serf-T}} (P',\nu')$. It is not completely trivial that serf reductions preserve subexponential solvability.

Proposition 4. *Let (P,ν) and (P',ν') be parameterized problems. If $(P,\nu) \leq^{\text{serf-T}} (P',\nu')$ and $(P',\nu') \in$ SUBEPT, then $(P,\nu) \in$ SUBEPT.*

Proof. Let Σ,Σ' be the alphabets of (P,ν), (P',ν'), respectively. Let $\mathbb{A}$ be a serf Turing reduction from (P,ν) to (P',ν'), and let f,g be the functions bounding the running time and the parameter. Let $f' : \mathbb{N} \to \mathbb{N}$ be a computable function and $\mathbb{A}'$ an algorithm that, given $(x',\ell') \in (\Sigma')^* \times \mathbb{N}$, decides if $x' \in P'$ in time $f'(\ell') \cdot 2^{\nu'(x')/\ell'} \cdot |x'|^{O(1)}$. Such $f',\mathbb{A}'$ exist by Lemma 2.

Let $\mathbb{B}$ be the algorithm that, given $(x,\ell) \in \Sigma^* \times \mathbb{N}$, simulates $\mathbb{A}$ and answers the oracle queries with instance x' by simulating $\mathbb{A}'$ on input (x',ℓ'), where $\ell' := g(\ell) \cdot \ell$. Observe that since the running time of $\mathbb{A}$ is bounded by $f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}$, for each oracle query with instance x' we have $|x'| \leq f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}$. Also recall that by the definition of subexponential reduction families, we have $\nu'(x') \leq g(\ell) \cdot (\nu(x) + \log|x|)$. Then $\mathbb{B}$ decides P, and the running time on input (x,ℓ) is bounded by

$$\begin{aligned}
& f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)} \cdot f'(\ell') \cdot 2^{\nu'(x')/\ell'} \cdot |x'|^{O(1)} \\
\leq\ & f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)} \cdot f'(g(\ell) \cdot \ell) \\
& \qquad \cdot 2^{(\nu(x)+\log|x|)/\ell} \cdot f(\ell)^{O(1)} \cdot 2^{O(\nu(x)/\ell)} \cdot |x|^{O(1)} \\
\leq\ & f(\ell)^{O(1)} \cdot f'(g(\ell) \cdot \ell) \cdot 2^{O(\nu(x)/\ell)} \cdot 2^{\log|x|/\ell} \cdot |x|^{O(1)} \\
\leq\ & h(\ell) \cdot 2^{O(\nu(x)/\ell)} \cdot |x|^{O(1)}
\end{aligned}$$

for a suitable computable function h. Thus by Lemma 2, $(P,\nu) \in$ SUBEPT. □

Example 5. For every graph problem P we trivially have *s*-$P \leq^{\text{serf}}$ *s-vert-P*, as a reduction family we can simply use the mapping $R(x,\ell) = x$. Similarly, for every satisfiability problem P we have *s*-$P \leq^{\text{serf}}$ *s-var-P*.

It is a highly nontrivial result due to Impagliazzo et al. [17] that for many natural graph and satisfiability problems, the converse also holds. As a matter of fact, Impagliazzo et al. proved that the following problems are all equivalent with respect to serf Turing reductions: *s*-3-SAT, *s-var*-3-SAT, *s*-INDEPENDENT-SET, *s-vert*-INDEPENDENT-SET, *s*-VERTEX-COVER, *s-vert*-VERTEX-COVER, *s*-DOMINATING-SET, and *s-vert*-CLIQUE.

Note that *s*-CLIQUE does not appear in this list of problems. Indeed, it seems unlikely that *s*-CLIQUE is equivalent to the problems above because it is in SUBEPT, whereas the other problems are not in SUBEPT unless the exponential time hypothesis (mentioned in the introduction) fails.

While unlikely to belong to SUBEPT, all problems considered in the previous example belong to the class EPT:

Definition 6. A parameterized problem (P,ν) is in EPT if there is an algorithm $\mathbb{A}$ that, for every instance x, decides if $x \in P$ in time $2^{O(\nu(x))} \cdot |x|^{O(1)}$.

It follows from the *Time Hierarchy Theorem* that SUBEPT is a *proper* subclass of EPT. The next proposition shows that EPT is closed under serf Turing reductions.

Proposition 7. *Let (P,ν) and (P',ν') be parameterized problems. If $(P,\nu) \leq^{\text{serf-T}} (P',\nu')$ and $(P',\nu') \in$ EPT, then $(P,\nu) \in$ EPT.*

Proof. It is not hard to see that, from the a serf Turing reduction from (P,ν) to (P',ν'), we can construct an algorithm $\mathbb{A}$ with an oracle to P' satisfying:

(R1) $\mathbb{A}$ decides if $x \in P$ in time $O(2^{\nu(x)} \cdot |x|^{O(1)})$ for any instance x.

(R2) For all oracle queries "$y \in P'$?" posed by $\mathbb{A}$ on input x we have $\nu'(y) = O(\nu(x) + \log|x|)$.

Let $\mathbb{A}'$ be an algorithm that, given x', decides $x' \in P'$ in time $2^{O(\nu'(x'))} \cdot |x'|^{O(1)}$.

Now we define the following algorithm $\mathbb{B}$ that, given an instance x, simulates $\mathbb{A}$ and answers the oracle queries with instance x' by simulating $\mathbb{A}'$ on x'. By (R1), for each oracle query with instance x' we have $|x'| = O(2^{\nu(x)} \cdot |x|^{O(1)})$. In addition, (R2) implies that $\nu'(x') = O(\nu(x) + \log|x|)$. It is clear that $\mathbb{B}$ decides P, and its running time on input x is bounded by

$$\begin{aligned} & O(2^{\nu(x)} \cdot |x|^{O(1)}) \cdot 2^{O(\nu'(x'))} \cdot |x'|^{O(1)} \\ = \; & O(2^{\nu(x)} \cdot |x|^{O(1)}) \cdot 2^{O(\nu(x)+\log|x|)} \cdot O(2^{O(\nu(x))} \cdot |x|^{O(1)}) \\ = \; & 2^{O(\nu(x))} \cdot |x|^{O(1)}. \end{aligned}$$

□

The following example introduces another problem that will turn out to be complete for the class EPT under serf-reduction.

Example 8. Consider the halting problem for alternating Turing machines with binary alphabet parameterized by the amount of space a computation uses (halting problems parameterized by space are referred to has "compact" halting problems in the parameterized complexity literature):

p-COMPACT-BIN-ATM-HALT
Instance: An alternating Turing machine M with binary alphabet, $k \in \mathbb{N}$.
Parameter: k.
Problem: Decide whether M accepts the empty input using at most k tape cells.

It is easy to see that *p*-COMPACT-BIN-ATM-HALT $\in$ EPT; just observe that for a given instance (M,k), there are at most $N = 2^{O(k)} \cdot |M|^{O(1)}$ many relevant configurations. So we can first compute the *configuration graph* of M of size N, and then test the accepting condition in time polynomial in N by computing the *alternating reachability problem* on that graph.

Parameterized complexity

As mentioned in the introduction, in parameterized complexity we are dealing with the same type of problems as in exponential complexity, namely parameterized problems (Q,κ), where $Q \subseteq \Sigma^*$ and $\kappa : \Sigma^* \to \mathbb{N}$ is polynomial time computable. However, we study the problems from a different perspective, in parameterized complexity theory we usually assume the parameter to be small, whereas in the exponential theory, the parameter is supposed to be a size measure and hence close to the size of the instance.

Definition 9. A parameterized problem (Q,κ) is *fixed-parameter tractable* if there is an algorithm $\mathbb{A}$ and a computable function f such that for every instance x $\mathbb{A}$ decides if $x \in Q$ in time

$$f(\kappa(x)) \cdot |x|^{O(1)}.$$

FPT denotes the class of all fixed-parameter tractable problems.

Observe that

$$\text{EPT} \subseteq \text{FPT}.$$

Example 10. The following parameterized vertex cover problem is maybe the best studied problem in parameterized complexity theory.

p-VERTEX-COVER
Instance: A graph G and a nonnegative integer k.
Parameter: k.
Problem: Decide if G has a vertex cover of cardinality k.

It is easy to see that p-VERTEX-COVER is fixed-parameter tractable. Actually, the problem is even in EPT.

Example 11. The problem *s-var*-SAT is in EPT and hence fixed-parameter tractable.

Definition 12. Let (Q,κ) and (Q',κ') be two parameterized problems over the alphabets Σ and Σ', respectively.

(1) A *(many-one)* fpt-*reduction* from (Q,κ) to (Q',κ') is a mapping $R:\Sigma^*\to(\Sigma')^*$ such that:

   (a) For all $x\in\Sigma^*$ we have $\big(x\in Q \iff R(x)\in Q'\big)$.

   (b) For all $x\in\Sigma^*$, the image $R(x)$ is computable in time

$$f(\kappa(x))\cdot|x|^{O(1)}$$

   for a computable $f:\mathbb{N}\to\mathbb{N}$.

   (c) There is a computable function $g:\mathbb{N}\to\mathbb{N}$ such that $\kappa'(R(x))\leq g(\kappa(x))$ for all $x\in\Sigma^*$.

(2) An fpt *Turing reduction* from (Q,κ) to (Q',κ') is an algorithm $\mathbb{A}$ with an oracle to Q' such that there are computable functions $f,g:\mathbb{N}\to\mathbb{N}$ with:

   (a) Given an instance $x\in\Sigma^*$, the algorithm $\mathbb{A}$ decides if $x\in Q$ in time

$$f(\kappa(x))\cdot|x|^{O(1)}.$$

   (b) For all oracle queries "$y\in Q'$?" posed by $\mathbb{A}$ on input $x\in\Sigma^*$ it holds that $\kappa'(y)\leq g(\kappa(x))$.

We write $(Q,\kappa)\leq^{\text{fpt}}(Q',\kappa')$ (or $(Q,\kappa)\leq^{\text{fpt-T}}(Q',\kappa')$) if there is an fpt-reduction (fpt Turing reduction, respectively) from (Q,κ) to (Q',κ'), and we write $(Q,\kappa)\equiv^{\text{fpt}}(Q',\kappa')$ (and similarly $(Q,\kappa)\equiv^{\text{fpt-T}}(Q',\kappa')$) if $(Q,\kappa)\leq^{\text{fpt}}(Q',\kappa')$ and $(Q',\kappa')\leq^{\text{fpt}}(Q,\kappa)$. It is easy the see that a many-one fpt-reduction is also an fpt Turing reduction, and FPT is closed under fpt Turing reductions.

Most parameterized problems that are studied in parameterized complexity theory have the property that for every fixed parameter value, the instances with this value are decidable in polynomial time (albeit by an algorithm whose running time is bounded by a polynomial that may depend on the parameter k). Essentially, XP is the class of all such problems, but with a uniformity condition added.

Definition 13. A parameterized problem (Q,κ) is in XP, if there is an algorithm $\mathbb{A}$ and a computable function $f:\mathbb{N}\to\mathbb{N}$ such that $\mathbb{A}$ decides Q, and the running time of $\mathbb{A}$ on input x is bounded by

$$O(|x|^{f(\kappa(x))})$$

Clearly, XP is closed under fpt- and fpt Turing reductions.

Example 14. The *parameterized compact halting problem for alternating Turing machines* is the following parameterized problem.

p-COMPACT-ATM-HALT *Instance:* An *alternating* Turing machine M with *arbitrary* alphabet, $k\in\mathbb{N}$. *Parameter:* k. *Problem:* Decide whether M accepts the empty input using space k.

It has been proved by Demri et al. [8] that p-COMPACT-ATM-HALT is complete for XP under fpt-reductions.

3 The miniaturization mapping

Definition 15. Let (P,ν) be a parameterized problem over the alphabet Σ. We define the *miniaturization* $\mathscr{M}(P,\nu)$ of (P,ν) as the following parameterized problem:

$\mathscr{M}(P,\nu)$
Instance: $x \in \Sigma^*$ and $m \in \mathbb{N}$ in *unary*.
Parameter: $\left\lceil \frac{\nu(x)}{\log m} \right\rceil$ if $m \geq 2$ and $\nu(x)$ otherwise.
Problem: Decide whether $x \in P$.

In other words, $\mathscr{M}(P,\nu)$ is the parameterization of the classical problem

Instance: $x \in \Sigma^*$ and $m \in \mathbb{N}$ in unary.
Problem: Decide whether $x \in P$.

with $\kappa(x,m) = \lceil \nu(x)/\log m \rceil$.

The ideas underlying the following result go back to [4, 9]. The theorem also follows from Theorem 19 below, but nevertheless we find it worthwhile to state and prove it separately first.

Theorem 16. *Let (P,ν) be a parameterized problem. Then $(P,\nu) \in$ SUBEPT $\iff \mathscr{M}(P,\nu) \in$ FPT.*

Proof. Let Σ be the alphabet of P. Assume (P,ν) is in SUBEPT. By Lemma 2, there is an algorithm $\mathbb{A}$ expecting inputs from $\Sigma^* \times \mathbb{N}$ and a computable function f such that for all $(x,\ell) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathbb{A}$ decides if $x \in P$ in time $f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}$.

Let $\mathbb{B}$ be the following algorithm: Given an instance $(x,m) \in \Sigma^* \times \mathbb{N}$, first $\mathbb{B}$ computes the parameter

$$k := \begin{cases} \left\lceil \frac{\nu(x)}{\log m} \right\rceil & \text{if } m \geq 2, \\ \nu(x) & \text{otherwise.} \end{cases}$$

It is easy to verify that

$$2^{\nu(x)/k} \leq 2m. \tag{3.1}$$

Then $\mathbb{B}$ simulates $\mathbb{A}$ on (x,k). The time taken by the simulation is bounded by

$$\begin{aligned} f(k) \cdot 2^{\nu(x)/k} \cdot |x|^{O(1)} &\leq f(k) \cdot 2m \cdot |x|^{O(1)} && \text{(by (3.1))} \\ &\leq 2f(k) \cdot (|x|+m)^{O(1)}. \end{aligned}$$

Therefore $\mathscr{M}(P,\nu)$ is fixed-parameter tractable.

For the converse direction, assume there is an algorithm $\mathbb{A}$ and a computable function $f : \mathbb{N} \to \mathbb{N}$ that for every (x,m) with $m \geq 2$ decides if (x,m) is a 'yes'-instance of $\mathscr{M}(P,\nu)$ in time

$$f\left(\left\lceil \frac{\nu(x)}{\log m} \right\rceil\right) \cdot (|x|+m)^{O(1)}.$$

Without loss of generality we assume that f is nondecreasing. Now let $\mathbb{B}$ be the algorithm that, for any given $(x,\ell) \in \Sigma^* \times \mathbb{N}$, first computes

$$m := \left\lceil 2^{\nu(x)/\ell} \right\rceil,$$

which can be done in time $2^{O(\nu(x)/\ell)} \cdot |x|^{O(1)}$. Note $m \geq 2$. Then $\mathbb{B}$ simulates $\mathbb{A}$ on (x,m), which requires time

$$f\left(\left\lceil \frac{\nu(x)}{\log m} \right\rceil\right) \cdot (|x|+m)^{O(1)} \leq f(\ell) \cdot 2^{O(\nu(x)/\ell)} \cdot |x|^{O(1)}.$$

So again by Lemma 2, (P,ν) is in SUBEPT. □

As the following two examples show, there is a number of problems that have natural parameterized problems as their miniaturizations. The *Hamming weight* of a finite Boolean valued function is the number of arguments that are mapped to TRUE.

Example 17. The miniaturization of *s-var*-CIRCUIT-SAT is fpt-equivalent to the following *parameterized weighted circuit satisfiability problem*:

> *p*-W-CIRCUIT-SAT
> *Instance:* A Boolean circuit C and a $k \in \mathbb{N}$.
> *Parameter:* k.
> *Problem:* Decide whether C has a satisfying assignment of Hamming weight k.

Essentially, this result goes back to Abrahamson, Downey and Fellows [1] (see [14] for a proof). It derives its significance from the fact that *p*-W-CIRCUIT-SAT is complete for the important parameterized complexity class W[P].

Example 18. The miniaturization of *p*-COMPACT-BIN-ATM-HALT (see Example 8) is fpt-equivalent to the parameterized problem *p*-COMPACT-ATM-HALT (see Example 14).

Proof. First it is easy to see the miniaturization of *p*-COMPACT-BIN-ATM-HALT is fpt-equivalent to the problem:

> p-$k \cdot \log n$-COMPACT-BIN-ATM-HALT
> *Instance:* An alternating Turing machine M with binary alphabet, $n \in \mathbb{N}$ in *unary*, $k \in \mathbb{N}$.
> *Parameter:* k.
> *Problem:* Decide whether M accepts the empty input using space $k \cdot \lceil \log n \rceil$.

Therefore it suffices to show *p*-COMPACT-ATM-HALT $\equiv^{\text{fpt}}$ p-$k \cdot \log n$-COMPACT-BIN-ATM-HALT.

p-COMPACT-ATM-HALT $\leq^{\text{fpt}}$ p-$k \cdot \log n$-COMPACT-BIN-ATM-HALT: Given an alternating machine M of arbitrary alphabet, it is not very hard, albeit tedious, to construct another alternating machine M_{bin} with binary alphabet by encoding each symbol in M by a binary string of length $\lceil \log |M| \rceil$. Thus M accepts the empty input using space k if and only if M_{bin} accepts the empty input using space $k \cdot \lceil \log |M| \rceil$. So

$$(M,k) \mapsto (M_{\text{bin}}, |M|, k)$$

is an appropriate fpt-reduction.

p-$k \cdot \log n$-COMPACT-BIN-ATM-HALT $\leq^{\text{fpt}}$ *p*-COMPACT-ATM-HALT: For an alternating machine M with binary alphabet and a natural number n, we can construct an alternating machine $M_{\lceil \log n \rceil}$ whose alphabet is of size n, each corresponds a $\lceil \log n \rceil$-bit binary. Thus $M_{\lceil \log n \rceil}$ can simulate a computation of M which uses space $k \cdot \lceil \log n \rceil$ by a computation using space k. Hence

$$(M,n,k) \mapsto (M_{\lceil \log n \rceil}, k)$$

gives the required fpt-reduction. □

The previous example can be generalized to the halting problems for *deterministic* and *nondeterministic* Turing machines parameterized by space. However for the moment we do not have a similar exact correspondence between "time" halting problems.

As we have seen in Theorem 16, miniaturization maps subexponential solvability of a problem exactly into the fixed-parameter tractability of its miniaturization. Indeed it can be viewed as a consequence of the following theorem.

Theorem 19. *Let* (P,v) *and* (P',v') *be parameterized problems. Then*

(1) $(P,v) \leq^{\text{serf}} (P',v') \iff \mathscr{M}(P,v) \leq^{\text{fpt}} \mathscr{M}(P',v')$.

(2) $(P,\nu) \leq^{\text{serf-T}} (P',\nu') \iff \mathscr{M}(P,\nu) \leq^{\text{fpt-T}} \mathscr{M}(P',\nu')$.

Proof. We show (2), the easier (1) is left to the reader. Let Σ and Σ' be the alphabets of P and P', respectively.

For the forward direction, let $\mathbb{A}$ be a serf Turing reduction from (P,ν) to (P',ν') such that for every input $(x,\ell) \in \Sigma^* \times \mathbb{N}$ the running time of $\mathbb{A}$ is bounded by $f(\ell) \cdot 2^{\nu(x)/\ell} \cdot |x|^{O(1)}$ and for any oracle query "$x' \in P'$?" posed by $\mathbb{A}$ it holds that

$$\nu'(x') \leq g(\ell) \cdot (\nu(x) + \log|x|) \tag{3.2}$$

for some computable function $g : \mathbb{N} \to \mathbb{N}$. Without loss of generality we assume that f is nondecreasing.

Let $\mathbb{B}$ be an algorithm that, given an instance (x,m) of $\mathscr{M}(P,\kappa)$, simulates $\mathbb{A}$ on (x,k), where

$$k := \begin{cases} \left\lceil \frac{\nu(x)}{\log m} \right\rceil & \text{if } m \geq 2, \\ \nu(x) & \text{otherwise,} \end{cases}$$

is the parameter of (x,m), and replaces each oracle query "$x' \in P'$?" posed by $\mathbb{A}$ by "$(x',m') \in \mathscr{M}(P',\nu')$?" where $m' := \max\{|x'|, m, 2\}$.

The overall running time of $\mathbb{B}$ is bounded by

$$f(k) \cdot 2^{\nu(x)/k} \cdot |x|^{O(1)} \leq f(k) \cdot 2m \cdot |x|^{O(1)}.$$

Furthermore, for each oracle query "$(x',m') \in \mathscr{M}(P',\nu')$?" posed by $\mathbb{B}$, we have $\nu'(x') \leq g(k) \cdot (\nu(x) + \log|x|)$ by (3.2). Since $m' \geq 2$, the parameter of (x',m') is $\left\lceil \frac{\nu'(x')}{\log m'} \right\rceil$.

- If $m \geq 2$, then $k = \left\lceil \frac{\nu(x)}{\log m} \right\rceil$ and we have

$$\begin{aligned} \left\lceil \frac{\nu'(x')}{\log m'} \right\rceil &\leq g(k) \cdot \left\lceil \frac{\nu(x) + \log|x|}{\log m'} \right\rceil \\ &\leq g(k) \cdot \left(\left\lceil \frac{\nu(x)}{\log m} \right\rceil + 1 \right) && \text{(by } m' \geq |x| \text{ and } m' \geq m\text{)} \\ &= g(k) \cdot (k+1). \end{aligned}$$

- Otherwise $m = 1$ and $k = \nu(x)$. It follows that

$$\begin{aligned} \left\lceil \frac{\nu'(x')}{\log m'} \right\rceil &\leq g(k) \cdot \left\lceil \frac{\nu(x) + \log|x|}{\log m'} \right\rceil \\ &\leq g(k) \cdot \left(\left\lceil \frac{\nu(x)}{\log m'} \right\rceil + 1 \right) && \text{(by } m' \geq |x|\text{)} \\ &\leq g(k) \cdot (k+1) && \text{(since } m' \geq 2 \text{ and } k = \nu(x)\text{)}. \end{aligned}$$

Thus $\mathbb{B}$ is an fpt Turing reduction from $\mathscr{M}(P,\nu)$ to $\mathscr{M}(P',\nu')$.

For the backward direction, let $\mathbb{A}$ be an algorithm with an oracle to $\mathscr{M}(P',\nu')$ such that there are nondecreasing computable functions $f,g : \mathbb{N} \to \mathbb{N}$ and a constant $d \in \mathbb{N}$ with:

(F1) Given an instance $(x,m) \in \Sigma^* \times \mathbb{N}$ with $m \geq 2$, the algorithm $\mathbb{A}$ decides if $(x,m) \in \mathscr{M}(P,\nu)$ in time

$$f\left(\left\lceil \frac{\nu(x)}{\log m} \right\rceil \right) \cdot (|x| + m)^d.$$

(F2) For all oracle queries "$(x',m') \in \mathscr{M}(P',v')$?" posed by $\mathbb{A}$ on input (x,m) with $m \geq 2$ we have

$$\left\lceil \frac{v'(x')}{\log m'} \right\rceil \leq g\left(\left\lceil \frac{v(x)}{\log m} \right\rceil \right),$$

if $m' \geq 2$, and

$$v'(x') \leq g\left(\left\lceil \frac{v(x)}{\log m} \right\rceil \right),$$

if $m' = 1$.

To show $(P,v) \leq^{\text{serf-T}} (P',v')$, let $\mathbb{B}$ be an algorithm with an oracle to (P',v') such that, given a pair $(x,\ell) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathbb{B}$ simulates $\mathbb{A}$ on (x,m), where

$$m := \max\left\{ |x|, \left\lceil 2^{v(x)/(d\cdot\ell)} \right\rceil \right\},$$

and replaces oracle queries "$(x',m') \in \mathscr{M}(P',v')$?" posed by $\mathbb{A}$ by "$x' \in P'$?".

Observe that $m \geq 2$ and

$$\frac{v(x)}{\log m} \leq d \cdot \ell. \tag{3.3}$$

So by (F1) the running time of $\mathbb{A}$ on (x,m), and hence $\mathbb{B}$ on (x,ℓ), is bounded by

$$f\left(\left\lceil \frac{v(x)}{\log m} \right\rceil \right) \cdot (|x|+m)^d \leq f(d\cdot\ell) \cdot 2^d \cdot 2^{v(x)/\ell} \cdot |x|^d.$$

Here we assume without loss of generality that $|x| > 0$. Note it follows that for any oracle query "$(x',m') \in \mathscr{M}(P',v')$?" posed by $\mathbb{A}$, we have

$$m' \leq f(d\cdot\ell) \cdot 2^d \cdot 2^{v(x)/\ell} \cdot |x|^d, \tag{3.4}$$

since it is represented in unary.

Now for any oracle query "$x' \in P'$?" posed by $\mathbb{B}$, which comes from the simulation of $\mathbb{A}$ over an oracle query "$(x',m') \in \mathscr{M}(P',v')$?", if $m' \geq 2$, then (F2) implies

$$\left\lceil \frac{v'(x')}{\log m'} \right\rceil \leq g\left(\left\lceil \frac{v(x)}{\log m} \right\rceil \right).$$

Therefore by (3.3) and (3.4)

$$v'(x') \leq g(d\cdot\ell) \cdot \log(f(d\cdot\ell) \cdot 2^d \cdot 2^{v(x)/\ell} \cdot |x|^d) \leq h(\ell) \cdot (v(x) + \log|x|)$$

for some suitable computable function h. Otherwise $m' = 1$. Again by (F2), we have

$$v'(x') \leq g\left(\left\lceil \frac{v(x)}{\log m} \right\rceil \right).$$

It follows that

$$v'(x') \leq g(d\cdot\ell) \leq g(d\cdot\ell) \cdot (v(x) + \log|x|).$$

So $\mathbb{B}$ is a serf Turing reduction from (P,v) to (P',v').

□

The main results of this and the following section can most elegantly be formulated in the language of *degrees* from classical recursion theory. Suppose we have some reducibility relation $\leq$ on parameterized problems, for example, $\leq^{\text{fpt}}$. In general, $\leq$ only needs to be a reflexive and transitive relation. Let us denote the corresponding equivalence relation by $\equiv$. Then the $\leq$-*degree* of a problem (Q,v), denoted by $[\![(Q,v)]\!]^{\leq}$, is the $\equiv$-equivalence class of (Q,v). For example, the $\leq^{\text{fpt}}$-degree of p-COMPACT-ATM-HALT

is the class of all XP-complete problems. The class of all $\leq$-degrees is denoted by $\mathbf{D}_{\leq}$, and for a class C of parameterized problems that is downward closed under $\leq$, the class of all degrees in C is denoted by $\mathbf{C}_{\leq}$. The reduction $\leq$ induces a partial order on $\mathbf{D}_{\leq}$. If $\leq = \leq^{\text{fpt}}$, then to simplify the notation we speak of fpt-degrees instead of $\leq^{\text{fpt}}$-degrees and write $[\![(Q,v)]\!]^{\text{fpt}}$, $\mathbf{D}_{\text{fpt}}$, et cetera. The same notational convention applies to reductions $\leq^{\text{fpt-T}}$, $\leq^{\text{serf}}$, and $\leq^{\text{serf-T}}$.

Note that by Theorem 19 (1), the miniaturization mapping induces a well-defined mapping $\mathscr{M} : \mathbf{D}_{\text{serf}} \to \mathbf{D}_{\text{fpt}}$, defined by $\mathscr{M}\left([\![(P,v)]\!]^{\text{serf}}\right) := \left[\!\left[\mathscr{M}(P,v)\right]\!\right]^{\text{fpt}}$, on the serf-degrees. By Theorem 19(2), it also induces a mapping on the serf Turing degrees. The main results of this section can be summarized in the following theorem:

Theorem 20 (Embedding Theorem). *The miniaturization mapping induces an embedding of the partially ordered set* $(\mathbf{D}_{\text{serf}}, \leq^{\text{serf}})$ *into the partially ordered set* $(\mathbf{D}_{\text{fpt}}, \leq^{\text{fpt}})$ *and also an embedding of the partially ordered set* $(\mathbf{D}_{\text{serf-T}}, \leq^{\text{serf-T}})$ *into the partially ordered set* $(\mathbf{D}_{\text{fpt-T}}, \leq^{\text{fpt-T}})$.

4 An Isomorphism between EPT and XP

Lemma 21. *Let* $(Q,\kappa) \in \text{XP}$. *Then there exists a problem* $(P,v) \in \text{EPT}$ *such that* $(Q,\kappa) \equiv^{\text{fpt}} \mathscr{M}(P,v)$.

Proof. Let Σ be the alphabet of Q. Without loss of generality we may assume that $Q \neq \emptyset$ and $Q \neq \Sigma^*$. In a first step we construct a problem (Q',κ') that is fpt-equivalent to (Q,κ) and decidable in time $O\left(|x|^{\sqrt{\kappa'(x)}}\right)$.

Suppose that $x \in Q$ is decidable in time $O\left(|x|^{f(\kappa(x))}\right)$, where without loss of generality f is increasing and time constructible. Let (Q',κ') be the following parameterized problem:

Input:	$x \in \Sigma^*$, and $\ell \in \mathbb{N}$ in unary such that $\ell \geq f(\kappa(x))^2$.
Parameter:	ℓ.
Problem:	Decide whether $x \in Q$.

It is easy to see that indeed $(Q',\kappa') \equiv^{\text{fpt}} (Q,\kappa)$ and that Q' is decidable in time $O\left(|x|^{\sqrt{\kappa'(x)}}\right)$.

In the second step, we construct the desired problem (P,v). Let Σ' be the alphabet of Q'. We let (P,v) be the following problem:

Input:	$x \in (\Sigma')^*$.
Parameter:	$\kappa'(x) \cdot \lceil \log |x| \rceil$ if $|x| \geq 2$ and $\kappa'(x)$ otherwise.
Problem:	Decide whether $x \in Q'$.

Then $P = Q'$, that is, (P,v) is just a re-parameterization of (Q',κ'). Recall that Q' is decidable in time

$$O\left(|x|^{\sqrt{\kappa'(x)}}\right) = O\left(2^{\sqrt{\kappa'(x)} \cdot \log |x|}\right) \leq O\left(2^{v(x)}\right).$$

It follows that $(P,v) \in \text{EPT}$. Now we claim that $\mathscr{M}(P,v) \equiv^{\text{fpt}} (Q',\kappa')$.

Let $x_+ \in Q'$ and $x_- \in (\Sigma')^* \setminus Q'$. To prove that $\mathscr{M}(P,v) \leq^{\text{fpt}} (Q',\kappa')$, we define a reduction R by letting

$$R(x,m) := \begin{cases} x_+ & \text{if } m \geq |x|^{\sqrt{\kappa'(x)}} \text{ and } x \in Q', \\ x_- & \text{if } m \geq |x|^{\sqrt{\kappa'(x)}} \text{ and } x \notin Q', \\ x & \text{otherwise.} \end{cases}$$

Then clearly for all $(x,m) \in \Sigma^* \times \mathbb{N}$ we have $(x,m) \in \mathscr{M}(P,v) \iff R(x,m) \in Q'$. Moreover, $R(x,m)$ is computable in polynomial time, because $x \in Q'$ is decidable in time $O(|x|^{\sqrt{\kappa'(x)}})$, which is $O(m)$ if $m \geq |x|^{\sqrt{\kappa'(x)}}$.

It remains to prove that the parameter $\kappa'(R(x,m))$ of the image is effectively bounded in terms of the parameter

$$k := \begin{cases} \lceil \nu(x)/\log m \rceil & \text{if } m \geq 2, \\ \nu(x) & \text{otherwise,} \end{cases}$$

of the argument. Let (x,m) be an instance of $\mathscr{M}(P,\nu)$. If $m \geq |x|^{\sqrt{\kappa'(x)}}$, then $\kappa(R(x,m)) \leq \max\{\kappa(x_+), \kappa(x_-)\}$, which is a constant. So let us assume that $m < |x|^{\sqrt{\kappa'(x)}}$. Then $\kappa'(R(x,m)) = \kappa'(x)$.

- If $|x| \geq 2$, then $\log m < \sqrt{\kappa'(x)} \cdot \lceil \log |x| \rceil = \nu(x)/\sqrt{\kappa'(x)}$, because $\nu(x) = \kappa'(x) \cdot \lceil \log |x| \rceil$. Thus in case $m \geq 2$ we have $\kappa'(x) = \kappa'(x) \cdot \log m / \log m < \sqrt{\kappa'(x)} \cdot \nu(x)/\log m$ and therefore $\kappa'(x) \leq (\nu(x)/\log m)^2 \leq k^2$. Otherwise $m = 1$, we have $k = \nu(x) = \kappa'(x) \cdot \lceil \log |x| \rceil \geq \kappa'(x)$.
- If $|x| < 2$, then $\kappa'(x)$ is bounded by a constant, since there are only finitely many such x.

This shows that indeed R is an fpt-reduction and proves $\mathscr{M}(P,\nu) \leq^{\text{fpt}} (Q',\kappa')$.

For the other direction, $(Q',\kappa') \leq^{\text{fpt}} \mathscr{M}(P,\nu)$, we define a reduction $R : (\Sigma')^* \to (\Sigma')^* \times \mathbb{N}$ by $R(x) = (x, 2|x|+2))$. It is easy to see that $\nu(x) \leq \kappa'(x) \cdot \lceil \log(|x|+2) \rceil$ for every $x \in (\Sigma')^*$. Thus

$$\left\lceil \frac{\nu(x)}{\log(2|x|+2))} \right\rceil \leq \left\lceil \frac{\kappa'(x) \cdot \lceil \log(|x|+2) \rceil}{\log(2|x|+2)} \right\rceil \leq \kappa'(x),$$

and R is an fpt-reduction from (Q',κ') to $\mathscr{M}(P,\nu)$. □

Now we can establish a similar correspondence as Theorem 16 with respect to XP and EPT.

Theorem 22. *Let (P,ν) be a parameterized problem. Then $(P,\nu) \in \text{EPT} \iff \mathscr{M}(P,\nu) \in \text{XP}$.*

Proof. Let $(P,\nu) \in \text{EPT}$, in other words, (P,ν) is decidable in time $2^{O(\nu(x))} \cdot |x|^O(1)$. Let (x,m) be an instance of $\mathscr{M}(P,\nu)$. If $m \geq 2$, then we let $k := \lceil \nu(x)/\log m \rceil$. Then the instance is decidable in time $2^{O(\nu(x))} \cdot |x|^{O(1)} \leq m^{O(\nu(x)/\log m)} \cdot |x|^{O(1)} \leq (m+|x|)^{O(k)}$. If $m = 1$, then the parameter k of (x,m) is $\nu(x)$. It follows that we can decide whether $(x,m) \in \mathscr{M}(P,\nu)$ in time $2^{O(\nu(x))} \cdot |x|^{O(1)} = 2^{O(k)} \cdot |x|^{O(1)}$. Hence, $\mathscr{M}(P,\nu)$ is in XP.

For another direction, assume $\mathscr{M}(P,\nu) \in \text{XP}$. By Lemma 21, there is a parameterized problem $(P',\nu') \in \text{EPT}$ such that $\mathscr{M}(P,\nu) \equiv^{\text{fpt}} \mathscr{M}(P',\nu')$. Now Theorem 19 implies $(P,\nu) \equiv^{\text{serf}} (P',\nu')$, and by Proposition 7, $(P,\nu) \in \text{EPT}$. □

Corollary 23. *Let (P,ν) be a parameterized problem. (P,ν) is* EPT*-complete (*EPT*-hard) under* serf*-reductions if and only $\mathscr{M}(P,\nu)$ is* XP*-complete (*XP*-hard, respectively) under* fpt*-reductions.*

Example 24. *p*-COMPACT-BIN-ATM-HALT is complete for EPT under serf-reductions.

Proof. *p*-COMPACT-ATM-HALT is complete for XP under fpt-reductions [8], and the miniaturization of *p*-COMPACT-ATM-HALT is fpt-equivalent to *p*-COMPACT-BIN-ATM-HALT (see Example 18). So Corollary 23 implies *p*-COMPACT-BIN-ATM-HALT is complete for EPT under serf-reductions. □

Rephrasing the results of this section in the language of degrees introduced at the end of the previous section, we obtain the following theorem:

Theorem 25 (Isomorphism Theorem). *The miniaturization mapping induces an isomorphism between $(\mathbf{EPT}_{\text{serf}}, \leq^{\text{serf}})$ and $(\mathbf{XP}_{\text{fpt}}, \leq^{\text{fpt}})$ and also an isomorphism between $(\mathbf{EPT}_{\text{serf-T}}, \leq^{\text{serf-T}})$ and $(\mathbf{XP}_{\text{fpt-T}}, \leq^{\text{fpt-T}})$.*

Outside EPT and XP

The following theorem shows that the Isomorphism Theorem cannot be extended from the degrees in EPT and XP to all degrees, because outside of XP the mapping induced by the miniaturization mapping is not onto.

Theorem 26. *There is a parameterized problem (Q,κ) that is not* fpt-T*-equivalent to $\mathscr{M}(P,\nu)$ for any (P,ν).*

We need some preparation before we prove the theorem.

Definition 27. Let Q and Q' be two classical problems. An algorithm $\mathbb{A}$ with an oracle to Q' is a *2-exptime Turing reduction* from Q to Q', if for any instance x of Q, $\mathbb{A}$ decides if $x \in Q$ in time

$$2^{2^{|x|^{O(1)}}}.$$

2-exptime Turing reductions are slightly at odds with all the usual reductions (including those introduced in this paper so far), namely they are not *transitive*. However they are closed under the composition with polynomial time Turing reductions. More precisely:

Lemma 28. *Let Q, Q', Q'' be classical problems. There is a 2-exptime Turing reduction from Q to Q'',*

- *if there is a 2-exptime Turing reduction from Q to Q', and a polynomial time Turing reduction from Q' to Q'';*
- *or, if there is a polynomial time Turing reduction from Q to Q', and a 2-exptime Turing reduction from Q' to Q''.*

We omit the routine proof.

Lemma 29. *There is a sequence of problems $(Q_i)_{i\in\mathbb{N}}$ such that*

- *$\big\{\{i\}\times Q_i \mid i\in\mathbb{N}\big\}$ is decidable,*
- *for all $i\in\mathbb{N}$, Q_i is not 2-exptime Turing reducible to*

$$L_i := \big\{(j,x) \mid j\neq i \text{ and } x\in Q_j\big\}.$$

Proof. Fix an alphabet Σ. Let $\mathbb{A}_1,\mathbb{A}_2,\ldots$ be an effective enumeration of all the 2-exptime Turing reductions from problems over the alphabet Σ to problems that are subsets of $\mathbb{N}\times\Sigma^*$.

For $S\subseteq\mathbb{N}\times\Sigma^*$, $e\in\mathbb{N}$, and $x\in\Sigma^*$, we define

$$\mathbb{A}_e^S(x) := \begin{cases} 1 & \mathbb{A}_e \text{ accepts } x \text{ with an oracle to } S,\\ 0 & \text{otherwise,}\end{cases}$$

and let

$$u(S,e,x) := \max\big\{|y| \;\big|\; \text{in the computation of } \mathbb{A}_e^S(x) \text{ there is an oracle query ``}y\in S?\text{''}\big\}.$$

Clearly, for a computable S the sets $\mathbb{A}_e^S(x)$ and $u(S,e,x)$ are both computable in e and x.

Note for given $S_1,S_2\subseteq\mathbb{N}\times\Sigma^*$, $e\in\mathbb{N}$, and $x\in\Sigma^*$, if for all $y\in\mathbb{N}\times\Sigma^*$ with $|y|\leq u(S_1,e,x)$, $(y\in S_1 \iff y\in S_2)$, then the computation of $\mathbb{A}_e^{S_1}(x)$ exactly coincides with that of $\mathbb{A}_e^{S_2}(x)$, in particular, $\mathbb{A}_e^{S_1}(x)=\mathbb{A}_e^{S_2}(x)$.

Let $\langle\cdot,\cdot\rangle:\mathbb{N}\times\mathbb{N}\to\mathbb{N}$ be a reasonable bijective encoding function. We shall construct a computable sequence of pairwise distinct elements $(a_j)_{j\in\mathbb{N}}\in\Sigma^*$ such that for $j=\langle i,e\rangle$, a_j witnesses the fact that

$$\mathbb{A}_e \text{ is not a reduction from } Q_i \text{ to } L_i.$$

Simultaneously, we will define a sequence $0 = \ell_0 < \ell_1 < \ell_2 \cdots$ of nonnegative integers and a sequence $\emptyset = P_0 \subseteq P_1 \subseteq P_2 \subseteq \cdots \subseteq \mathbb{N} \times \Sigma^*$ of sets such that for all $j \in \mathbb{N}$ with $j = \langle i, e \rangle$,

$$\ell_{j-1} < |(i, a_j)| \leq \ell_j, \tag{4.1}$$

$$\text{and} \quad |P_j \backslash P_{j-1}| \subseteq \{(i, a_j)\}. \tag{4.2}$$

Then we set, for each $i \in \mathbb{N}$,

$$Q_i := \Big\{ a \in \Sigma^* \mid (i, a) \in \bigcup_{j \in \mathbb{N}} P_j \Big\}. \tag{4.3}$$

Recall $\ell_0 = 0$, $P_0 = \emptyset$. Now for $j \in \mathbb{N}$ with $j = \langle i, e \rangle$, assume ℓ_{j-1} and P_{j-1} are already defined. Let $a_j \in \Sigma^*$ be the minimal element in the lexicographical order such that

$$|(i, a_j)| > \ell_{j-1}$$

and that a_j is distinct from all $a_{j'}$ for $1 \leq j' < j$. Now let

$$T_j := \big\{ (i', a) \in P_{j-1} \mid i' \neq i \big\} = P_{j-1} \backslash \big(\{i\} \times \Sigma^* \big). \tag{4.4}$$

and

$$\ell_j := \max \big\{ u(T_j, e, a_j),\ |(i, a_j)| \big\}. \tag{4.5}$$

(i) If $\mathbb{A}_e^{T_j}(a_j) = 0$, let $P_j := P_{j-1} \cup \{(i, a_j)\}$.

(ii) Otherwise $\mathbb{A}_e^{T_j}(a_j) = 1$, let $P_j := P_{j-1}$.

This finishes the construction.

Now we show that for each $i, e \in \mathbb{N}$, $\mathbb{A}_e$ is not a reduction from Q_i to

$$\begin{aligned} L_i &= \big\{ (i', a) \mid i' \neq i \text{ and } a \in Q_{i'} \big\} \\ &= \big\{ (i', a) \mid i' \neq i, \text{ and there exists a } j \in \mathbb{N} \text{ such that } (i', a) \in P_j \big\} \end{aligned} \tag{4.6}$$

Let $j := \langle i, e \rangle$. Note it suffices to prove that it is *not* the case that

$$a_j \in Q_i \quad \Longleftrightarrow \quad \mathbb{A}_e^{L_i}(a_j) = 1.$$

First observe that (4.4) and (4.6) imply $T_j \subseteq L_i$. And by (4.1), (4.2), and (4.5), for any $(i', a') \in L_i \backslash T_j$, we have $|(i', a')| > u(T_j, e, a_j)$. It follows that

$$\mathbb{A}_e^{L_i}(a_j) = \mathbb{A}_e^{T_j}(a_j). \tag{4.7}$$

Recall during the construction we have two cases for $\mathbb{A}_e^{T_j}(a_j)$:

(i) If $\mathbb{A}_e^{T_j}(a_j) = 0$, then $\mathbb{A}_e^{L_i}(a_j) = 0$ by (4.7). And by our construction $(i, a_j) \in P_j$, therefore $a_j \in Q_i$ by (4.3).

(ii) If $\mathbb{A}_e^{T_j}(a_j) = 1$, then $\mathbb{A}_e^{L_i}(a_j) = 1$. Hence $(i, a_j) \notin P_j \backslash P_{j-1}$. It follows that $(i, a_j) \notin P_{j'}$ for all $j' \in \mathbb{N}$ by (4.2). Consequently $a_j \notin Q_i$.

To see that $\{\{i\} \times Q_i \mid i \in \mathbb{N}\}$ is decidable, let $(i, a) \in \mathbb{N} \times \Sigma^*$ be an instance. Clearly

$$a \in Q_i \quad \Longleftrightarrow \quad (i, a) \in \bigcup_{j \in \mathbb{N}} P_j.$$

We compute the minimal $j \in \mathbb{N}$ such that

$$|(i, a)| \leq \ell_j,$$

and then decide if (i, a) is in the finite set P_j.

□

Proof of Theorem 26. Let $(Q_i)_{i\in\mathbb{N}}$ be as stated in Lemma 29. And for each $e \in \mathbb{N}$, let ϕ_e denote the e-th partially recursive function.

For any $e \in \mathbb{N}$ and $n \in \mathbb{N} \cup \{0\}$, let

$$C(e,n) := \begin{cases} k & k \text{ is maximum such that } \phi_e(1),\ldots,\phi_e(k) \text{ are defined,} \\ & \quad \text{together can be computed in at most } n \text{ steps, and each is smaller than } n, \\ 1 & \text{if no such } k \text{ exists.} \end{cases}$$

Clearly for any fixed e, $C(e,n)$ can be computed in time polynomial in n, and $C(e,n) \leq n$. Moreover if $\phi_e(1)$ is defined, then

$$\phi_e(C(e,n)) \leq \max\{\phi_e(1), n\}. \tag{4.8}$$

Now let

$$Q := \left\{(e,a,k) \mid e \in \mathbb{N},\, a \in Q_e, \text{ and } k = C(e,|a|)\right\}.$$

Define the parameterization κ by $\kappa(e,a,k) := k$.

Suppose for contradiction that

$$(Q,\kappa) \equiv^{\text{fpt-T}} \mathscr{M}(P,\nu)$$

for a parameterized problem (P,ν) over some alphabet Σ'. Let

$$\mu(x,m) = \left\lceil \frac{\nu(x)}{\log m} \right\rceil$$

be the parameterization of $\mathscr{M}(P,\nu)$.

Let $\mathbb{A}$ be an fpt-T-reduction from (Q,κ) to $\mathscr{M}(P,\nu)$, and let $f : \mathbb{N} \to \mathbb{N}$ be a computable function such that, for all instances x, the algorithm $\mathbb{A}$ decides if $x \in Q$ in time

$$f(\kappa(x)) \cdot |x|^{O(1)}.$$

Let $e \in \mathbb{N}$ be such that $\phi_e = f$. We leave e fixed for the rest of the proof.

Claim 1. There is a polynomial time Turing reduction from Q_e to $\mathscr{M}(P,\nu)$.

Proof of the claim: Let $\mathbb{B}$ be the following Turing reduction from Q_e to $\textsc{Mini}(P,\nu)$: Given an instance $a \in \Sigma^*$, the algorithm $\mathbb{B}$ computes in polynomial time $k := C(e,|a|)$. Then it simulates the fpt-T-reduction $\mathbb{A}$ on (e,a,k). Since $a \in Q_e \iff (e,a,k) \in Q$, this algorithm $\mathbb{B}$ correctly decides if $a \in Q_e$.

Moreover (4.8) implies that

$$\phi_e(\kappa(e,a,k)) = \phi_e(k) = O(|a|)$$

for ϕ_e is total. Hence the running time of $\mathbb{A}$ (and hence of $\mathbb{B}$) on (e,a,k) is

$$\begin{aligned} \phi_e(\kappa(e,a,k)) \cdot |(e,a,k)|^{O(1)} &= O(|a|) \cdot |(e,a,k)|^{O(1)} \\ &= O(|a|) \cdot |a|^{O(1)} \\ &= O(|a|^{O(1)}). \end{aligned}$$

The second equality follows from the fact that $k = C(e,|a|) \leq |a|$ and e is a constant. Thus $\mathbb{B}$ is a desired polynomial time Turing reduction, which proves Claim 1.

Since ν is polynomial time computable, without loss of generality we assume

$$\nu(x) \leq 2^{|x|} \tag{4.9}$$

for any $x \in (\Sigma')^*$. Let

$$P' := \left\{(x, 2^{2^{|x|}}) \mid x \in P\right\}.^3$$

It follows that for all $(x,m) \in P'$

$$m = 2^{2^{|x|}}. \tag{4.10}$$

Claim 2. $\mathscr{M}(P,\nu)$ is 2-exptime Turing reducible to P'.
Proof of the claim: Given an instance $(x,m) \in (\Sigma')^* \times \mathbb{N}$, we have

$$(x,m) \in \mathscr{M}(P,\nu) \iff x \in P \iff (x, 2^{2^{|x|}}) \in P'.$$

This can be easily turned into a 2-exptime Turing reduction, thus Claim 2 is proved.

Note that for all $(x,m) \in P'$ we have

$$\begin{aligned} \mu(x,m) &= \left\lceil \frac{\nu(x)}{\log m} \right\rceil \\ &\leq \left\lceil \frac{2^{|x|}}{\log 2^{2^{|x|}}} \right\rceil \qquad \text{(by (4.9) and (4.10))} \\ &= 1. \end{aligned} \tag{4.11}$$

Claim 3. There is a polynomial time Turing reduction $\mathbb{B}$ from P' to Q. Moreover the set

$$\big\{(e,a,k) \in Q \mid \text{for some } (x,m) \in (\Sigma')^* \times \mathbb{N}, \text{ there is an oracle query ``}(e,a,k) \in Q\text{?'' in the computation of } \mathbb{B} \text{ on } (x,m)\big\}$$

is *finite*.
Proof of the claim: Recall we assume $\mathscr{M}(P,\nu) \equiv^{\text{fpt-T}} (Q,\kappa)$. So there is an algorithm $\mathbb{A}'$ with an oracle to (Q,κ) and a computable function $g : \mathbb{N} \to \mathbb{N}$, such that, given an instance $(x,m) \in (\Sigma')^* \times \mathbb{N}$:

(E1) $\mathbb{A}'$ decides if $(x,m) \in \mathscr{M}(P,\nu)$ in time

$$g(\mu(x,m)) \cdot |(x,m)|^{O(1)}.$$

(E2) For each oracle query "$(e',a',k') \in Q$?" posed by $\mathbb{A}'$,

$$\kappa(e',a',k') = k' \leq g'(\mu(x,m))$$

for a computable function $g' : \mathbb{N} \to \mathbb{N}$. For simplicity, we assume $g' = g$.

Given an instance (x,m) of P', the algorithm $\mathbb{B}$ first checks in polynomial time if $m \neq 2^{2^{|x|}}$ or $\mu(x,m) > 1$. If so, then (x,m) is a 'no'-instance by (4.10) and (4.11). Otherwise $m = 2^{2^{|x|}} \geq |x|$, and $\mu(x,m) = 1$. It follows that

$$(x,m) \in P' \iff x \in P \iff (x,m) \in \mathscr{M}(P,\nu).$$

Therefore $\mathbb{B}$ decides if $(x,m) \in P'$ by simulating $\mathbb{A}'$ on (x,m), which by (E1) requires time

$$g(\mu(x,m)) \cdot |(x,m)|^{O(1)} = g(1) \cdot |(x,m)|^{O(1)},$$

therefore polynomial in $|(x,m)|$.

Now for any oracle query "$(e',a',k') \in Q$?" that occurs in the simulation of $\mathbb{A}'$ on (x,m), by (E2) we have

$$k' = \kappa(e',a',k') \leq g(\mu(x,m)) = g(1).$$

[3] Here $2^{2^{|x|}}$ is represented in unary.

If $(e',a',k') \in Q$ and $e' = e$, then $k' = C(e,|a'|)$ is the maximum such that $\varphi_e(1),\ldots,\varphi_e(k')$ can be computed in at most $|a'|$ steps and all bounded by $|a'|$. As k' is bounded by the fixed constant $g(1)$ independent of x and m, there are only finitely many such a' with $(e,a',k') \in Q$ for all possible instances (x,m), otherwise ϕ_e is not total. Thus Claim 3 is proved.

Claim 4. There is a polynomial time Turing reduction from P' to

$$L_e = \big\{(e',a) \mid e' \neq e \text{ and } a \in Q_{e'}\big\}.$$

Proof of the claim: By Claim 3 it suffices to give a polynomial time Turing reduction from

$$\{(e',a,k) \in Q \mid e' \neq e\}$$

to L_e: For any instance (e',a,k), if $k \neq C(e',|a|)$ or $e' = e$, then it is a 'no'-instance. Otherwise

$$(e',a,k) \in \big\{(e',a,k) \in Q \mid e' \neq e\big\} \iff (e',a) \in L_e.$$

This proves Claim 4.

Combining Claims 1,2, and 4, we have a 2-exptime Turing reduction from Q_e to L_e by Lemma 28, which contradicts our construction. □

5 The S-hierarchy and the W-hierarchy

The purpose of this last section of the paper is to gather further evidence that the miniaturization mapping is not only an abstract isomorphism between exponential and parameterized complexity theory, but actually establishes a relation between interesting and relevant complexity classes on both sides. Specifically, we shall lay out a relation between a natural hierarchy of the exponential theory and the W-hierarchy of parameterized complexity theory. The basic ideas underlying this section go back to Abrahamson, Downey, and Fellows [1] and have been refined in [5, 6, 13].

The W-hierarchy is defined in terms of *weighted satisfiability problems* for classes Γ of Boolean formulas or circuits:

p-WSAT(Γ) *Instance:* $\gamma \in \Gamma$ and $k \in \mathbb{N}$. *Parameter:* k. *Problem:* Decide whether γ has a satisfying assignment of Hamming weight k.

We denote the class of Boolean circuits by CIRC, and the class of formulas by FORM. Recall that FORM is viewed as a subclass of CIRC. Note that p-WSAT(CIRC) is exactly the parameterized problem p-W-CIRCUIT-SAT introduced in Example 17.

For $t \geq 0$ and $d \geq 1$ we inductively define the following classes $\Gamma_{t,d}$ and $\Delta_{t,d}$ of Boolean formulas:

$$\begin{aligned}
\Gamma_{0,d} &:= \{\lambda_1 \wedge \ldots \wedge \lambda_c \mid c \leq d, \lambda_1,\ldots,\lambda_c \text{ literals}\},\\
\Delta_{0,d} &:= \{\lambda_1 \vee \ldots \vee \lambda_c \mid c \leq d, \lambda_1,\ldots,\lambda_c \text{ literals}\},\\
\Gamma_{t+1,d} &:= \Big\{\bigwedge_{i\in I} \delta_i \mid I \text{ finite}, \delta_i \in \Delta_{t,d} \text{ for all } i \in I\Big\},\\
\Delta_{t+1,d} &:= \Big\{\bigvee_{i\in I} \gamma_i \mid I \text{ finite}, \gamma_i \in \Gamma_{t,d} \text{ for all } i \in I\Big\}.
\end{aligned}$$

The *W-hierarchy* of parameterized complexity theory consists of the following classes:

Definition 30. (1) For $t \geq 1$, $\mathrm{W}[t] := \bigcup_{d\geq 1}\big\{(Q,\kappa) \mid (Q,\kappa) \leq^{\mathrm{fpt}} p\text{-WSAT}(\Gamma_{t,d})\big\}$.

(2) $\mathrm{W}[\mathrm{SAT}] := \big\{(Q,\kappa) \mid (Q,\kappa) \leq^{\mathrm{fpt}} p\text{-WSAT}(\mathrm{FORM})\big\}$.

(3) $\mathrm{W[P]} := \big\{(Q,\kappa) \mid (Q,\kappa) \leq^{\mathrm{fpt}} p\text{-}\mathrm{WSAT}(\mathrm{CIRC})\big\}$.

This definition of the W-hierarchy establishes the role of weighted satisfiability problems as the "generic" (hard) problems of parameterized complexity theory. We propose that plain (unweighted) satisfiability problems with the "number of variables" size measure can play a similar role in exponential complexity theory. For every class Γ of Boolean formulas or circuits, we let:

s-var-$\mathrm{SAT}(\Gamma)$
Instance: $\gamma \in \Gamma$.
Parameter: Number of variables of γ.
Problem: Decide whether γ is satisfiable.

The *S-hierarchy* consists of the following classes:

Definition 31. (1) For $t \geq 1$, $\mathrm{S}[t] := \bigcup_{d\geq 1} \big\{(P,\nu) \mid (P,\nu) \leq^{\mathrm{serf}} s\text{-}var\text{-}\mathrm{SAT}(\Gamma_{t,d})\big\}$.

(2) $\mathrm{S[SAT]} := \big\{(P,\nu) \mid (P,\nu) \leq^{\mathrm{serf}} s\text{-}var\text{-}\mathrm{SAT}(\mathrm{FORM})\big\}$.

(3) $\mathrm{S[P]} := \big\{(P,\nu) \mid (P,\nu) \leq^{\mathrm{serf}} s\text{-}var\text{-}\mathrm{SAT}(\mathrm{CIRC})\big\}$.

So far, mainly the first level S[1] of the S-hierarchy has been studied, and some highly nontrivial completeness results are known. Most importantly, Impagliazzo, Paturi, and Zane [17] have proved that $s\text{-}var\text{-}\mathrm{SAT}(\Gamma_{1,3})$ (that is, 3-satisfiability) is complete for S[1] under serf Turing reductions. Thus the exponential hypothesis (discussed in the introduction) is equivalent to $\mathrm{S[1]} \neq \mathrm{SUBEPT}$.

The image of the S-hierarchy

Definition 32. (1) For a class C of parameterized problems that is downward closed under serf-reducibility, the *image of* C *under the miniaturization mapping* is the class

$$\mathscr{M}(\mathrm{C}) := \big\{(Q,\kappa) \mid (Q,\kappa) \leq^{\mathrm{fpt}} \mathscr{M}(P,\nu) \text{ for some } (P,\nu) \in \mathrm{C}\big\}.$$

(2) For a class C of parameterized problems that is downward closed under fpt-reducibility, the *preimage of* C *under the miniaturization mapping* is the class

$$\mathscr{M}^{-1}(\mathrm{C}) := \{(P,\nu) \mid \mathscr{M}(P,\nu) \in \mathrm{C}\}.$$

To relate the S-hierarchy and the W-hierarchy, we consider the image of the S-hierarchy under the miniaturization mapping (the so-called *M-hierarchy*):

Definition 33. For $t \geq 1$, we let $\mathrm{M}[t] := \mathscr{M}(\mathrm{S}[t])$. Furthermore, we let $\mathrm{M[SAT]} := \mathscr{M}(\mathrm{S[SAT]})$ and $\mathrm{M[P]} := \mathscr{M}(\mathrm{S[P]})$.

The M-hierarchy is a natural hierarchy within the realm of parameterized complexity theory. It is populated not only by the miniaturizations of the satisfiability problems, but also by the following re-parameterizations. For every class Γ of Boolean formulas or circuits, we let:

$p\text{-}log^{-1}$-$\mathrm{SAT}(\Gamma)$
Instance: A circuit $\gamma \in \Gamma$ of size m with n inputs (variables).
Parameter: $\lceil n/\log m \rceil$.
Problem: Decide if γ is satisfiable.

The motivation to study these problems can be explained as follows: For simplicity, let us consider $\Gamma = \mathrm{CIRC}$, and let m denote the size and n the number of input gates. If we parameterize $\mathrm{SAT}(\mathrm{CIRC})$ by n then we obtain the familiar problem $s\text{-}var\text{-}\mathrm{SAT}(\mathrm{CIRC})$, which is fixed-parameter tractable; it even belongs to the class EPT. A parameterized problem gets "harder" if we decrease the parameter. For the satisfiability

problem, we may consider the parameterizations $\big(\text{SAT}(\text{CIRC}), \kappa_h\big)$ for functions $h : \mathbb{N} \to \mathbb{N}$, where for every circuit γ of size m with n inputs we let

$$\kappa_h(\gamma) = \left\lceil \frac{n}{h(m)} \right\rceil.$$

For constant $h = 1$, κ_h is just our old parameterization by the number of inputs, and therefore the problem $\big(\text{SAT}(\text{CIRC}), \kappa_h\big) = s\text{-SAT}(\text{CIRC})$ is fixed-parameter tractable. At the other end of the scale, for $h(m) \geq m \geq n$ we have $\kappa_h(\gamma) = 1$, and essentially $\big(\text{SAT}(\text{CIRC}), \kappa_h\big)$ is just the NP-complete unparameterized problem SAT(CIRC). More formally, the problem is complete for the parameterized complexity class para-NP (cf. [14]). Thus by increasing the function h we can shift the fixed-parameter tractable problem s-SAT(CIRC) to a highly intractable problem that is not even contained in the class XP (unless PTIME = NP). We leave it has an easy exercise for the reader to prove that for $h(m) \in o^{\text{eff}}(\log m)$ the problem $\big(\text{SAT}(\text{CIRC}), \kappa_h\big)$ is still fixed-parameter tractable. For $h(m) \in \omega(\log m)$, it seems unlikely that the problem $\big(\text{SAT}(\text{CIRC}), \kappa_h\big)$ is in XP, and therefore it is not so interesting from the point of view of parameterized complexity. However, for $h(m) \in \Theta(\log m)$ the problem $\big(\text{SAT}(\text{CIRC}), \kappa_h\big)$ is right on the "boundary" of XP. Note that $p\text{-}\log^{-1}\text{-SAT}(\text{CIRC})$ is $\big(\text{SAT}(\text{CIRC}), \kappa_h\big)$ for $h(m) = \log m$.

While the problem $p\text{-}\log^{-1}\text{-SAT}(\Gamma)$ bears some similarity with $\mathscr{M}\,(s\text{-}var\text{-SAT}(\Gamma))$, the two problems are not the same. Nevertheless, it can be shown that for "well-behaved" classes Γ of circuits, which include CIRC, FORM, and all classes $\Gamma_{t,d}$, the two problems are fpt-equivalent [13]. Hence the M-hierarchy can be characterized directly in terms of the log-parameterizations:

$$\begin{aligned}
\text{M}[t] &= \bigcup_{d \geq 1} \big\{ (Q, \kappa) \;\big|\; (Q, \kappa) \leq^{\text{fpt}} p\text{-}\log^{-1}\text{-SAT}(\Gamma_{t,d}) \big\} && \text{for } t \geq 1, \\
\text{M}[\text{SAT}] &= \big\{ (Q, \kappa) \;\big|\; (Q, \kappa) \leq^{\text{fpt}} p\text{-}\log^{-1}\text{-SAT}(\text{FORM}) \big\}, \\
\text{M}[\text{P}] &= \big\{ (Q, \kappa) \;\big|\; (Q, \kappa) \leq^{\text{fpt}} p\text{-}\log^{-1}\text{-SAT}(\text{CIRC}) \big\}.
\end{aligned}$$

¿From the fact that s-SAT($\Gamma_{1,3}$) (i.e., 3-SAT) is complete for S[1] under serf Turing reductions, it follows that $p\text{-}\log^{-1}\text{-SAT}(\Gamma_{1,3})$ is complete for M[1] under fpt Turing reductions. Another natural problem complete for M[1] is the following re-parameterized vertex cover problem; this result is due to Cai and Juedes [4]:

> $p\text{-}\log^{-1}$-VERTEX-COVER
> *Instance:* A graph G of size m and a nonnegative integer k.
> *Parameter:* $\lceil k/\log m \rceil$.
> *Problem:* Decide if G has a vertex cover of cardinality k.

The following theorem shows how the M-hierarchy relates to the W-hierarchy:

Theorem 34 (Abrahamson et al. [1]). *For every $t \geq 1$, $\text{M}[t] \subseteq \text{W}[t] \subseteq \text{M}[t+1]$. Moreover $\text{M}[\text{SAT}] = \text{W}[\text{SAT}]$ and $\text{M}[\text{P}] = \text{W}[\text{P}]$.*

For a proof, we refer the reader to [13] or [14].

A schematic figure illustrating the relations between the complexity classes considered so far can be found in Figure 1.

The preimage of the W-hierarchy

The discussion above shows that the M-hierarchy, that is, the image of the S-hierarchy under the miniaturization mapping, is reasonably well understood and is interesting also for "intrinsically parameterized complexity" reasons. What about the preimage of the W-hierarchy in the world of exponential complexity (shown as dashed ovals in Figure 1)? The remainder of the section is devoted to this question.

When we studied the M-hierarchy and, more generally, the image of problems in EPT under the miniaturization mapping, we re-parameterized the problems by dividing the parameter by the logarithm of the instance size, hence making the parameter smaller and the problems "harder". It turned out that this increased the parameterized complexity by just the right amount, shifting problems from EPT to XP. The

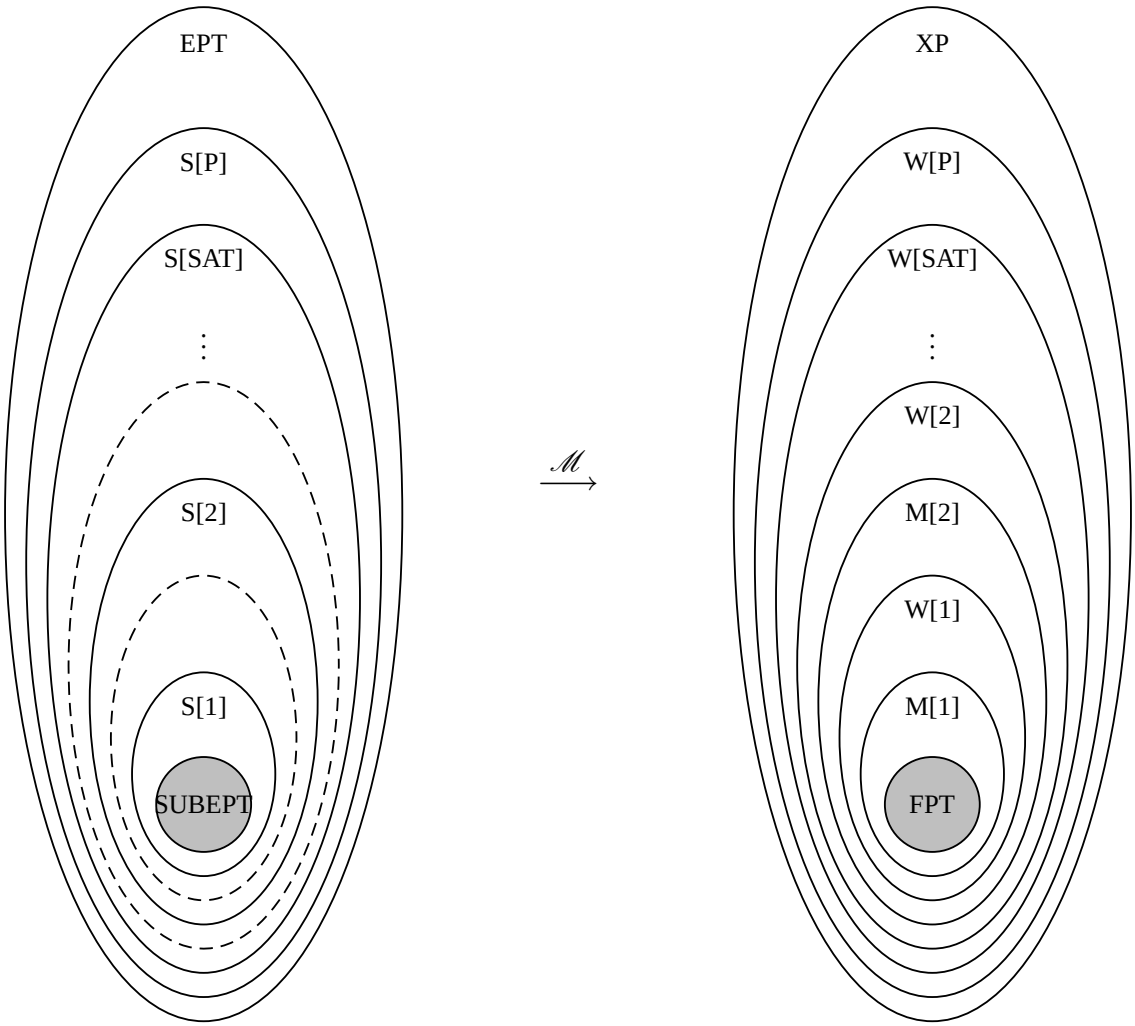

Figure 1: Overview over the complexity classes and hierarchies

natural idea for the converse direction is to multiply the parameter by the logarithm of the instance size, hence making the parameter larger and the problems "easier". And again, we will see that this simple idea works.

It will be convenient to first prove that the reparameterization of problems by multiplying the parameter by the logarithm of the instance size gives an inverse for the miniaturization mapping for all problems satisfying certain technical conditions. After that, we shall prove that the W-hierarchy can be characterized in terms of problems satisfying these technical conditions.

Definition 35. Let (Q,κ) be a parameterized problem over the alphabet Σ.

The *log reparameterization* of (Q,κ) is the mapping $\mathscr{L}$ between parameterized problems defined by $\mathscr{L}(Q,\kappa) := (Q,\lambda)$, where $\lambda : \Sigma^* \to \mathbb{N}$ with

$$\lambda(x) = \begin{cases} \lceil \kappa(x) \cdot \log|x| \rceil & \text{if } |x| \geq 2, \\ 1 & \text{otherwise,} \end{cases}$$

for all $x \in \Sigma^*$.

Then obviously:

Lemma 36. *For all parameterized problems (Q,κ) it holds that*

$$(Q,\kappa) \leq^{\text{fpt}} \mathscr{M}(\mathscr{L}(Q,\kappa)).$$

Proof. The mapping $x \mapsto (x,|x|)$ is an fpt-reduction. □

We can prove a partial converse for problems that have the following property:

Definition 37. A parameterized problem (Q,κ) over the alphabet Σ is *scalable* if there is a mapping $F : \Sigma^* \times \mathbb{N}$ such that for all $x \in \Sigma^*$ and $s \geq \max\{|x|,2\}$ it holds that:

(1) $(x \in Q \iff F(x,s) \in Q)$;

(2) $F(x,s)$ is computable in time polynomial in $(|x|+s)$;

(3) $\kappa(F(x,s)) \leq \left\lceil \kappa(x) \cdot \frac{\log|x|}{\log s} \right\rceil$.

Example 38. The parameterized independent set problem

p-INDEPENDENT-SET *Instance:* A graph G and $k \geq 1$. *Parameter:* k. *Problem:* Decide whether G has an independent set of cardinality k.

is scalable.

To see this, let $G = (V,E)$ be a graph and $k \in \mathbb{N}$. Let m be the size of the instance (G,k) and $s \geq 2$. Clearly $m \geq 2$. Let $\ell := \lceil \log s / \log m \rceil$. Then $m^{\ell-1} < s \leq m^{\ell}$. Without loss of generality we assume that k is a multiple of ℓ (in the proof of Lemma 42 we shall see how to avoid such an assumption). Let G' be the graph whose vertices are all independent sets of G of size ℓ, with an edge between two such independent sets if they are disjoint and their union is an independent set of size 2ℓ. Then G has an independent set of size k if and only if G' has an independent set of size

$$\frac{k}{\ell} \leq k \cdot \frac{\log m}{\log s} \leq \left\lceil k \cdot \frac{\log m}{\log s} \right\rceil.$$

We let $(G', k/\ell)$ be the scaled instance of *p*-INDEPENDENT-SET.

The *slices* of a parameterized problem (Q,κ) are the classical problems $Q_i = \{x \in Q \mid \kappa(x) = i\}$, for $i \geq 1$. Observe that if a problem is scalable and its first slice is polynomial time decidable, then the problem is in XP. To see this, let (Q,κ) be a scalable problem such that Q_1 is polynomial time decidable. Let x be an instance. Scale x with $s = |x|^{\kappa(x)}$. Then the resulting instance x' has parameter value 1. Hence $x' \in Q$ can be decided in polynomial time. Note that this actually gives an algorithm deciding $x \in Q$ in time $|x|^{O(\kappa(x))}$.

Lemma 39. *Let (Q,κ) be a scalable problem. Then*

$$\mathscr{M}(\mathscr{L}(Q,\kappa)) \equiv^{\text{fpt}} (Q,\kappa).$$

Proof. By Lemma 36, we have $(Q,\kappa) \leq^{\text{fpt}} \mathscr{M}(\mathscr{L}(Q,\kappa))$. Thus we only have to prove the converse. Let Σ be the alphabet of Q. Let $F : \Sigma^* \times \mathbb{N} \to \Sigma^*$ be a function witnessing that (Q,κ) is scalable. We define a reduction $R : \Sigma^* \times \mathbb{N} \to \Sigma^*$ by:

$$R(x,m) := \begin{cases} F(x,m) & \text{if } m \geq |x| \geq 2, \\ x & \text{otherwise.} \end{cases}$$

Obviously, $(x,m) \in \mathscr{M}(\mathscr{L}(Q,\kappa))$ if and only if $R(x,m) \in Q$. Furthermore, R is polynomial time computable because F is. To see that the parameter of the R-image of an instance can be bounded in terms of the parameter of the instance, let (x,m) be an instance of $\mathscr{M}(\mathscr{L}(Q,\kappa))$ and $x' := R(x,m)$. If $|x| < 2$, then

$x' = x$, and $\kappa(x') = \kappa(x)$ is bounded by a constant, since there are only finitely many x with $|x| < 2$. So we assume $|x| \geq 2$ and let

$$k := \begin{cases} \left\lceil \frac{\lceil \kappa(x) \cdot \log |x| \rceil}{\log m} \right\rceil & \text{if } m \geq 2, \\ \lceil \kappa(x) \cdot \log |x| \rceil & \text{otherwise,} \end{cases}$$

be the parameter value of (x,m). If $m \geq |x|$, then $x' = F(x,m)$ and

$$\kappa(x') \leq \left\lceil \frac{\kappa(x) \cdot \log |x|}{\log m} \right\rceil \leq k,$$

where the first inequality holds by the definition of scalability. Otherwise $m < |x|$. It follows that $x' = x$ and $\kappa(x') = \kappa(x)$.

- If $m \geq 2$, then
$$\kappa(x) = \frac{\kappa(x) \cdot \log m}{\log m} < \frac{\kappa(x) \cdot \log |x|}{\log m} \leq \left\lceil \frac{\lceil \kappa(x) \cdot \log |x| \rceil}{\log m} \right\rceil = k.$$
- Otherwise $m = 1$ and we have
$$k = \lceil \kappa(x) \cdot \log |x| \rceil \geq \kappa(x).$$

□

The notation in the following example requires explanation: For functions $f, g : \mathbb{N}^2 \to \mathbb{N}$ we say that $g \in o^{\text{eff}}(f)$ if there is a computable function $\iota : \mathbb{N} \to \mathbb{N}$ that is nondecreasing and unbounded such that for all m,k it holds that $g(m,k) \leq f(m,k)/\iota(m+k)$. We say that $g \in 2^{o^{\text{eff}}(f(m,k))}$ if there is a function $g' \in o^{\text{eff}}(f)$ such that $g(m,k) \leq 2^{g'(m,k)}$ for all m,k. Some care needs to be taken with this notation. For example,

$$2^{o^{\text{eff}}(k \cdot \log n)} \subset o^{\text{eff}}(n^k).$$

An example of a function in $o^{\text{eff}}(n^k) \setminus 2^{o^{\text{eff}}(k \cdot \log n)}$ is $n^{k/2}$.

Example 40. The log reparameterization of *p*-INDEPENDENT-SET is the problem:

s-log-INDEPENDENT-SET
Instance: A graph G of size m and $k \geq 1$.
Parameter: $\lceil k \cdot \log m \rceil$.
Problem: Decide if G has an independent set of cardinality k.

Hence the miniaturization mapping maps *s-log*-INDEPENDENT-SET to a parameterized problem fpt-equivalent to *p*-INDEPENDENT-SET, or equivalently, maps the serf-degree $[\![$*s-log*-INDEPENDENT-SET$]\!]^{\text{serf}}$ to the fpt-degree $[\![p$-INDEPENDENT-SET$]\!]^{\text{fpt}}$. As *p*-INDEPENDENT-SET is complete for $\mathrm{W}[1]$ under fpt-reductions [11], it follows that *s-log*-INDEPENDENT-SET is complete for $\mathscr{M}^{-1}(\mathrm{W}[1])$ under serf-reductions.

An interesting consequence of this result is that $\mathrm{FPT} = \mathrm{W}[1]$ if and only if there is an algorithm deciding whether a graph of size m has an independent set of size k in time $2^{o^{\text{eff}}(k \cdot \log m)} \cdot m^{O(1)}$, or equivalently, an algorithm deciding whether an n-vertex graph has an independent set of size k in time $2^{o^{\text{eff}}(k \cdot \log n)} \cdot n^{O(1)}$.

We cannot apply the same technique as in the previous example to the the defining problems p-WSAT$(\Gamma_{t,d})$ of the W-hierarchy directly, because they are not (obviously) scalable. We take a detour through the monotone and antimonotone versions of these problems. Let us call a Boolean formula *monotone*, if it contains no negations, and *antimonotone*, if all variables are negated and no other negations occur. For a class Φ of formulas, we use Φ^+ to denote the class of monotone formulas in Φ and similarly Φ^- for the antimonotone formulas.

Lemma 41 (Downey and Fellows [10, 11]). *Let $t,d \in \mathbb{N}$ such that $t + d \geq 3$.*

(1) If t is even, then p-WSAT$(\Gamma^+_{t,d})$ is complete for $\mathrm{W}[t]$ under fpt-reductions.

(2) If t is odd, then p-WSAT$(\Gamma^-_{t,d})$ is complete for $\mathrm{W}[t]$ under fpt-reductions.

Lemma 42. *Let $t,d \in \mathbb{N}$ such that $t+d \geq 3$.*

(1) If t is even, then p-WSAT$(\Gamma^+_{t,d})$ is scalable.

(2) If t is odd, then p-WSAT$(\Gamma^-_{t,d})$ is scalable.

Furthermore, the problems p-WSAT(FORM) and p-WSAT(CIRC) are scalable.

Proof. Assume first that t is odd. Let $\gamma \in \Gamma^-_{t,d}$ be a formula of size m with n variables. Without loss of generality we assume $m \geq n \geq 2$ and let $k,s \in \mathbb{N}$ such that $n \geq k \geq 1, s \geq m$, and $\ell := \lceil \log s / \log n \rceil$. Let us assume first that k is a multiple of ℓ.

Let V denote the set of variables of γ. For each ℓ-element subset $S \subseteq V$ we introduce a new variable $Y_S \notin V$. Now we replace each negative literal $\neg X$ in γ by the conjunction

$$\bigwedge_{\substack{S \subseteq V \text{ with} \\ |X| = \ell \text{ and } X \in S}} \neg Y_S.$$

The resulting formula can be transformed into an equivalent $\Gamma^-_{t,d}$-formula γ' in time $O(m^{d \cdot \ell})$. We let

$$\gamma'' := \gamma' \wedge \bigwedge_{\substack{S,S' \subseteq V \text{ with} \\ |S| = |S'| = \ell \text{ and } S \cap S' \neq \emptyset}} (\neg Y_S \vee \neg Y_{S'}).$$

As $t+d \geq 3$, the resulting formula is still in $\Gamma^-_{t,d}$. It is easy to see that γ'' has a satisfying assignment of Hamming weight k/ℓ if and only if γ has a satisfying assignment of Hamming weight k. Furthermore,

$$\frac{k}{\ell} \leq k \cdot \frac{\log n}{\log s} \leq k \cdot \frac{\log m}{\log s} \leq \left\lceil k \cdot \frac{\log m}{\log s} \right\rceil.$$

It remains to explain what we do if k is a not multiple of ℓ. We let $\tilde{k}$ be the least multiple of ℓ greater than k. Then we choose $\tilde{s} \geq s$ such that $\lceil \log \tilde{s} / \log(n+1+\tilde{k}-k) \rceil = \ell$. We let $\tilde{\gamma}$ be a the $\Gamma^+_{t,d}$ formula

$$\gamma \wedge (\neg Y_1 \vee \neg Z) \wedge \ldots \wedge (\neg Y_{\tilde{k}-k} \vee \neg Z),$$

where the $Y_1, \ldots, Y_{\tilde{k}-k}, Z$ are new variables not in γ. Then $\tilde{\gamma}$ has a satisfying assignment of Hamming weight $\tilde{k}$ if and only if γ has a satisfying assignment of Hamming weight k. We apply the construction above to the instance $(\tilde{\gamma}, \tilde{k})$ and $\tilde{s}$ obtain a $\Gamma^+_{t,d}$-formula $\tilde{\gamma}''$ that has a satisfying assigment of Hamming weight $\tilde{k}/\ell$ if and only if γ has a satisfying assignment of weight k. As $\lceil \tilde{k}/\ell \rceil = \lceil k/\ell \rceil$ by the choice of $\tilde{k}$, this is good enough.

Assume now that t is even. Let $\gamma \in \Gamma^+_{t,d}$ be a formula of size m with n variables, and let $k,s \in \mathbb{N}$ such that $n \geq k \geq 1, s \geq m$. Let $\ell := \lceil \log s / \log n \rceil$. Without loss of generality, we may assume that k is a multiple of ℓ. Let V be the set of variables of γ. For each ℓ-element subset $S \subseteq V$ we introduce a new variable Y_S. Now we replace each variable X in γ by the disjunction $\bigvee_{S \subseteq V \text{ with } X \in S} Y_S$. The resulting formula can be transformed into an equivalent $\Gamma^+_{t,d}$-formula γ' in time $O(m^{d \cdot \ell})$. It is easy to see that γ' has a satisfying assignment of Hamming weight k/ℓ if γ has a satisfying assignment of Hamming weight k. Conversely, if γ' has a satisfying assignment of Hamming weight k/ℓ then γ has a satisfying assignment of Hamming weight at most k, and hence by monotonicity, a satisfying assignment of weight exactly k. Moreover, we have $k/\ell \leq \lceil k \cdot \log m / \log s \rceil$ (as above).

Using similar ideas, it is easy to prove that p-WSAT(FORM) and p-WSAT(CIRC) are scalable. □

For every class Γ of Boolean formulas or circuits, we let:

s-log-WSAT(Γ)
Instance: $\gamma \in \Gamma$ of size m and $k \in \mathbb{N}$.
Parameter: $\lceil k \cdot \log m \rceil$.
Problem: Decide if γ has a satisfying assignment of Hamming weight k.

Corollary 43. *Let $t, d \in \mathbb{N}$ such that $t + d \geq 3$.*

(1) If t is even, then s-log-$\mathrm{WSAT}(\Gamma^+_{t,d})$ *is complete for* $\mathscr{M}^{-1}(\mathrm{W}[t])$ *under* serf*-reductions.*

(2) If t is odd, then s-log-$\mathrm{WSAT}(\Gamma^-_{t,d})$ *is complete for* $\mathscr{M}^{-1}(\mathrm{W}[t])$ *under* serf*-reductions.*

As $\mathscr{M}^{-1}(\mathrm{W}[\mathrm{SAT}]) = \mathrm{S}[\mathrm{SAT}]$ and $\mathscr{M}^{-1}(\mathrm{W}[\mathrm{P}]) = \mathrm{S}[\mathrm{P}]$, we also get:

Corollary 44. *(1) s-log-*$\mathrm{WSAT}(\mathrm{FORM})$ *is complete for* $\mathrm{S}[\mathrm{SAT}]$ *under* serf*-reductions.*

(2) s-log-$\mathrm{WSAT}(\mathrm{CIRC})$ *is complete for* $\mathrm{S}[\mathrm{P}]$ *under* serf*-reductions.*

Note that, combined with our earlier results, (2) implies the following corollary. (Statement (1) has a similar consequence for formulas, which we do not state explicitly.)

Corollary 45. *The following statements are equivalent:*

(1) There is an algorithm deciding if a circuit of size m with n inputs is satisfiable in time $2^{o^{\mathrm{eff}}(n)} \cdot m^{O(1)}$.

(2) There is an algorithm deciding if a circuit of size m with n inputs has a satisfying assignment of Hamming weight k in time $2^{o^{\mathrm{eff}}(k \cdot \log m)} \cdot m^{O(1)}$.

Corollary 43 already yields a characterization of the preimage of the W-hierarchy under the miniaturization mapping, but we find it not yet completely satisfactory because it involves a restriction of the satisfiability problems not used so far. Fortunately, we can easily get rid of this restriction.

Lemma 46. *Let $t, d \in \mathbb{N}$ such that $t + d \geq 3$.*

(1) If t is even, then s-log-$\mathrm{WSAT}(\Gamma_{t,d}) \equiv^{\mathrm{serf}}$ *s-log-*$\mathrm{WSAT}(\Gamma^-_{t,d})$.

(2) If t is odd, then s-log-$\mathrm{WSAT}(\Gamma_{t,d}) \equiv^{\mathrm{serf}}$ *s-log-*$\mathrm{WSAT}(\Gamma^+_{t,d})$.

Proof. In [16], the authors give a polynomial time reductions from $\mathrm{WSAT}(\Gamma_{t,d})$ to $\mathrm{WSAT}(\Gamma^+_{t,d})$ for even t and from $\mathrm{WSAT}(\Gamma_{t,d})$ to $\mathrm{WSAT}(\Gamma^+_{t,d})$ for odd t that are linear in the parameter. These reductions can easily be turned into the desired serf-reductions. □

Finally, we are ready to state and prove the main result of this section:

Theorem 47. *For every $t \geq 1$,*

$$\mathscr{M}^{-1}\big(\mathrm{W}[t]\big) = \bigcup_{d \geq 1} \big\{ (P, v) \bigm| (P, v) \leq^{\mathrm{serf}} \textit{s-log-}\mathrm{WSAT}(\Gamma_{t,d}) \big\}.$$

6 Concluding remarks

This paper is a contribution to a (not yet clearly established) exponential complexity theory. A long term goal of such a theory might be to prove that the exponential time hypothesis is equivalent to $\mathrm{P} \neq \mathrm{NP}$ and thus obtain a solid basis for exponential lower bounds such as the one for 3-SAT stated by the exponential time hypothesis. However, with current methods this goal seems out of reach, and maybe such an equivalence cannot be established without actually proving that the exponential time hypothesis and hence $\mathrm{P} \neq \mathrm{NP}$ holds.[4]

What we can establish now is a close connection between exponential and parameterized complexity theory. The obvious open question is whether the exponential time hypothesis is equivalent to $\mathrm{FPT} \neq \mathrm{W}[1]$, or more or less equivalently, whether $\mathrm{M}[1] = \mathrm{W}[1]$. Despite serious efforts, so far researchers in parameterized complexity have not been able to prove that $\mathrm{M}[1] = \mathrm{W}[1]$, even though this still seems quite plausible. However, it would also be compatible with our current knowledge that $\mathrm{M}[2] = \mathrm{W}[1]$.

The higher levels of the S-hierarchy (or equivalently, the M-hierarchy), have not yet received much attention. Specifically, no completeness results for $\mathrm{S}[2]$ are known, even though the class contains natural problems such as *s-var-*SAT that are not believed to be in $\mathrm{S}[1]$. It may be worthwhile to study the class $\mathrm{S}[2]$ and develop a completeness theory for this class similar to the $\mathrm{S}[1]$-completeness theory, which is based on Impagliazzo, Paturi, and Zane's Sparsification Lemma [17].

[4] However, 20 years ago an equivalence between the inapproximability of, say, MAX-SAT and $\mathrm{P} \neq \mathrm{NP}$ also must have seemed far out of reach.

References

[1] K.A. Abrahamson, R.G. Downey, and M.R. Fellows. Fixed-parameter tractability and completeness IV: On completeness for W[P] and PSPACE analogs. *Annals of Pure and Applied Logic*, 73:235–276, 1995.

[2] J. Alber, H. Fernau, and R. Niedermeier. Parameterized complexity: Exponential speed-up for planar graph problems. In F. Orejas, P.G. Spirakis, and J. van Leeuwen, editors, *28th International Colloquium on Automata, Languages and Programming*, volume 2076 of *Lecture Notes in Computer Science*, pages 261–272. Springer-Verlag, 2001.

[3] T. Brueggemann and W. Kern. An improved deterministic local search algorithm for 3-SAT. *Theoretical Computer Science*, 329:303–313, 2004.

[4] L. Cai and D. Juedes. On the existence of subexponential parameterized algorithms. *Journal of Computer and System Sciences*, 67(4):789–807, 2003.

[5] J. Chen, B. Chor, M. Fellows, X. Huang, D. Juedes, I. Kanj, and G. Xia. Tight lower bounds for certain parameterized NP-hard problems. In *Proceedings of the 19th IEEE Conference on Computational Complexity*, pages 150–160, 2004.

[6] J. Chen, X. Huang, I. Kanj, and G. Xia. Linear fpt reductions and computational lower bounds. In *Proceedings of the 36th ACM Symposium on Theory of Computing*, pages 212–221, 2004.

[7] Y. Chen and J. Flum. On miniaturized problems in parameterized complexity. *Theoretical Computer Science*, 351:314–336, 2006.

[8] S. Demri, F. Laroussinie, and Ph. Schnoebelen. A parametric analysis of the state explosion problem in model checking. In H. Alt and A. Ferreira, editors, *Proceedings of the 19th Annual Symposium on Theoretical Aspects of Computer Science*, volume 2285 of *Lecture Notes in Computer Science*, pages 620–631. Springer-Verlag, 2002.

[9] R.G. Downey, V. Estivill-Castro, M. Fellows, E. Prieto-Rodriguez, and F. Rosamond. Cutting up is hard to do: the parameterized complexity of *k*-cut and related problems. In J. Harland, editor, *Proceedings of the Australian Theory Symposium*, volume 78 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science Publishers, 2003.

[10] R.G. Downey and M.R. Fellows. Fixed-parameter tractability and completeness I: Basic results. *SIAM Journal on Computing*, 24:873–921, 1995.

[11] R.G. Downey and M.R. Fellows. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theoretical Computer Science*, 141:109–131, 1995.

[12] R.G. Downey and M.R. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.

[13] J. Flum and M. Grohe. Parameterized complexity and subexponential time. *Bulletin of the EATCS*, 84:71–100, October 2004.

[14] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer-Verlag, 2006.

[15] J. Flum, M. Grohe, and M. Weyer. Bounded fixed-parameter tractability and $\log^2 n$ nondeterministic bits. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 555–567. Springer-Verlag, 2004.

[16] J. Flum, M. Grohe, and M. Weyer. Bounded fixed-parameter tractability and $\log^2 n$ nondeterministic bits. *Journal of Computer and System Sciences*, 72:34–71, 2006.

[17] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[18] D. Rolf. Improved bound for the PPSZ/Schöning-algorithm for 3-SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 1:111–122, 2006.

[19] U. Schöning. Algorithmics in exponential time. In V. Diekert and B. Durand, editors, *Proceedings of 22nd Annual Symposium on Theoretical Aspects of Computer Science*, volume 3404, pages 36–43. Springer-Verlag, 2005.

[20] G. Woeginger. Space and time complexity of exact algorithms: Some open problems. In R.G. Downey, M. Fellows, and F. Dehne, editors, *Proceedings of the 1st International Workshop on Parameterized and Exact Computation*, volume 3162 of *Lecture Notes in Computer Science*, pages 36–43. Springer-Verlag, 2004.