Paradigms for Parameterized Enumeration*

Nadia Creignou[1], Arne Meier[2], Julian-Steffen Müller[2], Johannes Schmidt[3], and Heribert Vollmer[2]

[1] Aix-Marseille Université
nadia.creignou@lif.univ-mrs.fr
[2] Leibniz Universität Hannover
{meier,mueller,vollmer}@thi.uni-hannover.de
[3] Linköping University
johannes.schmidt@liu.se

Abstract. The aim of the paper is to examine the computational complexity and algorithmics of enumeration, the task to output all solutions of a given problem, from the point of view of parameterized complexity. First we define formally different notions of efficient enumeration in the context of parameterized complexity. Second we show how different algorithmic paradigms can be used in order to get parameter-efficient enumeration algorithms in a number of examples. These paradigms use well-known principles from the design of parameterized decision as well as enumeration techniques, like for instance kernelization and self-reducibility. The concept of kernelization, in particular, leads to a characterization of fixed-parameter tractable enumeration problems.

1 Introduction

This paper is concerned with algorithms for and complexity studies of enumeration problems, the task of generating all solutions of a given computational problem. The area of enumeration algorithms has experienced tremendous growth over the last decade. Prime applications are query answering in databases and web search engines, data mining, web mining, bioinformatics and computational linguistics.

Parameterized complexity theory provides a framework for a refined analysis of hard algorithmic problems. It measures complexity not only in terms of the input size, but in addition in terms of a parameter. Problem instances that exhibit structural similarities will have the same or similar parameter(s). Efficiency now means that for fixed parameter, the problem is solvable with reasonable time resources. A parameterized problem is fixed-parameter tractable (in FPT) if it can be solved in polynomial time for each fixed value of the parameter, where the degree of the polynomial does not depend on the parameter. Much like in the classical setting, to give evidence that certain algorithmic problems are not in

* Supported by a Campus France/DAAD Procope grant, Campus France Projet No 28292TE, DAAD Projekt-ID 55892324.

K. Chatterjee and J. Sgall (Eds.): MFCS 2013, LNCS 8087, pp. 290–301, 2013.

FPT one shows that they are complete for superclasses of FPT, like the classes in what is known as the W-hierarchy.

Our main goal is to initiate a study of enumeration from a parameterized complexity point of view and in particular to develop parameter-efficient enumeration algorithms. Preliminary steps in this direction have been undertaken by H. Fernau [5]. He considers algorithms that output *all* solutions of a problem to a given instance in polynomial time for each fixed value of the parameter, where, as above, the degree of the polynomial does not depend on the parameter (let us briefly call this fpt-time). We subsume problems that exhibit such an algorithm in the class Total-FPT. (A similar notion was studied by Damaschke [4]). Algorithms like these can of course only exists for algorithmic problems that possess only relatively few solutions for an input instance. We therefore consider algorithms that exhibit a delay between the output of two different solutions of fpt-time, and we argue that this is the "right way" to define tractable parameterized enumeration. The corresponding complexity class is called Delay-FPT.

We then study the techniques of kernelization (stemming from parameterized complexity) and self-reducibility (well-known in the design of enumeration algorithms) under the question if they can be used to obtain parameter-efficient enumeration algorithms. We study these techniques in the context of different algorithmic problems from the context of propositional satisfiability (and vertex cover, which can, of course, also be seen as a form of weighted 2-CNF satisfiability question). We obtain a number of upper and lower bounds on the enumerability of these problems.

In the next section we introduce parameterized enumeration problems and suggest four hopefully reasonable complexity classes for their study. In the following two sections we study in turn kernelization and self-reducibility, and apply them to the problems Vertex-Cover, MaxOnes-SAT and detection of strong Horn-backdoor sets. We conclude with some open questions about related algorithmic problems.

2 Complexity Classes for Parameterized Enumeration

Because of the amount of solutions that enumeration algorithms possibly produce, the size of their output is often much larger (e.g., exponentially larger) than the size of their input. Therefore, polynomial time complexity is not a suitable yardstick of efficiency when analyzing their performance. As it is now agreed, one is more interested in the regularity of these algorithms rather than in their total running time. For this reason, the efficiency of an enumeration algorithm is better measured by the delay between two successive outputs, see e.g., [7]. The same observation holds within the context of parametrized complexity and we can define parameterized complexity classes for enumeration based on this time elapsed between two successive outputs. Let us start with the formal definition of a parameterized enumeration problem.

Definition 1. *A* parameterized enumeration problem *(over a finite alphabet Σ) is a triple $E = (Q, \kappa, \mathrm{Sol})$ such that*