Parameterized Compilability

Master of Logic Thesis Defense

Noel Arteche · Supervised by Ronald de Haan and Hubie Chen

July 11, 2022

Institute for Logic, Language and Computation (ILLC) · University of Amsterdam

Table of contents

Introduction

Classical (polynomial-size) compilation

Assume the instances $x = (y, z)$ of a problem are made of two parts:

- y is the offline part;
- z is the online part.

We can (expensively) compile y into a representation y' that can be polynomially bigger to hopefully solve (y', z) in polynomial time.

Compilation of logic problems

Term Inference

Does $\varphi \models \ell_1 \wedge \cdots \wedge \ell_k$?

Clause Inference

Does $\varphi \models \ell_1 \vee \cdots \vee \ell_k$?

Formula Inference

Does $\varphi \models \psi$?

Compile φ!

Theorem

All three inference problems are **coNP**-complete.

Positive and negative results

Theorem (Selman and Kautz, 1996; Cadoli et al., 2002)

- Term Inference is efficiently compilable.
- Clause Inference is *not* efficiently compilable unless $\mathsf{PH} = \Sigma_2^{\mathsf{p}}$.
- Formula Inference is *not* efficiently compilable unless $\mathsf{P} = \mathsf{NP}$.

The parameter compilation framework

Chen's 2015 *parameter compilation framework* models compilability as a special case of fixed-parameter tractability (FPT).

We study parameterized problems (Q, κ) such that κ points at the compilable part of the input.

Example (TERM INFERENCE)

On input $(\varphi, \ell_1 \wedge \cdots \wedge \ell_k)$ we compile φ. Hence we are interested in the parameterized problem (TERM INFERENCE, φ).

The framework consists of the (parameterized) classes **poly-comp-C**, where **C** is a classical complexity class.

The class **poly-comp-P** models efficient compilation.

A new framework for parameterized compilability

Doubly parameterized problems

We introduce doubly parameterized problems:

$$(Q, \kappa, \lambda)$$

where

- $Q \subseteq \Sigma^*$ is a *decision problem*;
- $\kappa : \Sigma^* \to \Sigma^*$, computable in polynomial time, points at the *compilable part of the input*;
- $\lambda : \Sigma^* \to \Sigma^*$ is a *parameterization* that relaxes the size of the compilation from polynomial-size to λ-fpt-size.

The new fpt-comp-C classes

Let C be a parameterized complexity class like FPT, W[1], para-NP...
We define **fpt-comp-C** as containing all the (Q, κ, λ) such that on input $x \in \Sigma^*$,

- we can compile $\kappa(x)$ into something of λ-fpt-size:

$$|c(\kappa(x), \lambda(x))| \leq h(\lambda(x)) \cdot p(|\kappa(x)|)$$

  for some computable c, computable h and some polynomial p;
- x together with $c(\kappa(x), \lambda(x))$ and parameter $\lambda(x)$ can be solved within the resources of **C**.

Efficient parameterized compilation is captured by

$$\text{fpt-comp-FPT} = \text{fpt-comp-P}.$$

The fpt-comp reductions and methodology theorems

Our **fpt-comp-C** classes are closed under a new notion of reduction: the fpt-comp reductions ($\leq^{\mathsf{fpt}}_{\mathsf{comp}}$).

Theorem (General methodology theorem, Thm. 2.19)

Let C and C' be parameterized complexity classes (like FPT, $\mathsf{W}[1]$, para-NP...). If

- (A, λ) is C-hard,
- $(A, \mathsf{len}, \lambda) \leq^{\mathsf{fpt}}_{\mathsf{comp}} (B, \kappa, \mu)$,
- $(B, \kappa, \mu) \in \mathsf{fpt}$-$\mathsf{comp}$-$\mathsf{C}'$,

then $\mathsf{C} \subseteq \mathsf{C}' / \mathsf{fpt}$.

Example

Take $\mathsf{C} = \mathsf{W}[1]$ and $\mathsf{C}' = \mathsf{FPT}$. If $(\textsc{Clique}, \mathsf{len}, k) \leq^{\mathsf{fpt}}_{\mathsf{comp}} (Q, \kappa, \lambda)$ and $(Q, \kappa, \lambda) \in \mathsf{fpt}$-$\mathsf{comp}$-$\mathsf{FPT}$, then $\mathsf{W}[1] \subseteq \mathsf{FPT} / \mathsf{fpt}$ (cf. $\mathsf{NP} \subseteq \mathsf{P} / \mathsf{poly}$).

Selected compilability results

SAT Completion	
Instance	A Boolean formula φ and a partial assignment α.
Question	Is there an extension of α into a satisfying assignment for φ?

Additional parameterization

The number u of undefined variables in α.

Proposition

(SAT Completion, φ) $\notin$ **poly-comp-P** (unless **PH** collapses) but (SAT Completion, φ, u) $\in$ **fpt-comp-FPT**.

CSP Completion	
Instance	An instance $I = \langle X, D, C \rangle$ of CSP and a partial assignment $\alpha : X \to D$.
Question	Is there an extension of α into a satisfying assignment for I?

Additional parameterization

The number u of undefined variables in α.

Theorem (Corollary 3.7)

(CSP Completion, u) is W[1]-complete.

Theorem (Thm. 3.9)

(CSP Completion, I, u) $\notin$ **fpt-comp-FPT** unless **W[1]** $\subseteq$ **FPT/ fpt**.

Conclusion

Overview of results in the thesis

- We developed an extension of the parameter compilation framework that models parameterized compilability (Chapter 2).
- We showed parameterized uncompilability results for *completion variants* of problems around the classes W[1] and W[2] (Chapters 3 and 4):

WEIGHTED q-SAT COMPLETION	(**chopped**-W[1]-complete)
HITTING SET COMPLETION	(**chopped**-W[2]-complete)
DOMINATING SET COMPLETION	(**chopped**-W[1]-hard)
...	

- We studied the issue of treewidth in compiling CSP instances, both for CSP COMPLETION and inference problems related to CSP (Chapter 5).

Future work

- Classify more problems under the framework.
- Show tighter classifications, e.g.
  - for Dominating Set Completion, currently **chopped**-W[1]-hard but not **chopped**-W[2]-complete;
  - for (CSP Completion, I, **ptw**), currently **chopped**-W[1]-hard but not known to be complete for any class.
- Focus on positive results.

Thanks for listening!

References

M. Cadoli, F. M. Donini, P. Liberatore, and M. Schaerf.
Preprocessing of intractable problems.
Information and computation, 176(2):89–120, 2002.

H. Chen.
Parameter compilation.
In *10th International Symposium on Parameterized and Exact Computation*, page 127, 2015.

B. Selman and H. Kautz.
Knowledge compilation and theory approximation.
Journal of the ACM (JACM), 43(2):193–224, 1996.

Reductions for poly-comp-C

Definition ($\leq^{\text{poly}}_{\text{comp}}$ reductions)

We say that (A, κ) *poly-comp-reduces* to (B, λ) if there is

- a polynomial-size function $c : \Sigma^* \to \Sigma^*$,
- a polynomial-time function $f : \Sigma^* \times \Sigma^* \to \Sigma^*$

such that for all $x \in \Sigma^*$,

$$x \in A \Leftrightarrow f(x, c(\kappa(x))) \in B$$

and there is a polynomial-size function $s : \Sigma^* \to \mathcal{P}_{\text{fin}}(\Sigma^*)$ such that

$$\lambda(f(x, c(\kappa(x)))) \in s(\kappa(x)).$$

A system for negative results

Hardness of problems is established via reductions from languages where the compilation has access to the length of the input (len).

Theorem (Chen, 2015)

Let A be a C-complete language. If $(A, \mathsf{len}) \leq^{\mathsf{poly}}_{\mathsf{comp}} (B, \kappa)$ and $(B, \kappa) \in \textbf{poly-comp-C}'$, then $\mathsf{C}' \subseteq \mathsf{C}_{/\mathsf{poly}}$.

Example

If $(\mathsf{SAT}, \mathsf{len}) \leq^{\mathsf{poly}}_{\mathsf{comp}} (A, \kappa)$ and $(A, \kappa) \in \textbf{poly-comp-P}$, then $\mathsf{NP} \subseteq \mathsf{P}_{/\mathsf{poly}}$. By the Karp-Lipton theorem, PH collapses.

The case of SAT COMPLETION

SAT COMPLETION

Given a Boolean formula φ and a partial assignment α, decide whether α can be extended into a satisfying assignment for φ.

Theorem

$(\text{3SAT}, \mathsf{len}) \leq^{\mathsf{poly}}_{\mathsf{comp}} (\text{SAT COMPLETION}, \pi_1)$.

Proof. Reduction via the "superinstance technique". Over n variables, there are $\binom{2n}{3} \in O(n^3)$ possible clauses. Enumerate them: $\mathcal{C} = \{C_1, C_2, C_3 \dots\}$ and build the formula $\bigwedge_{c_i \in \mathcal{C}} (c_i \to C_i)$. With a partial assignment to the c_i variables we can "configure" this superinstance to represent our specific formula. □

Corollary

$(\text{SAT COMPLETION}, \pi_1) \notin$ **poly-comp-P** unless **PH** collapses.

Hitting Set and Dominating Set Completion

Hitting Set Completion (HS-C)	
Instance	An instance $\langle U, S, k\rangle$ of Hitting Set together with sets $I, O \subseteq U$ and a set $A \subseteq S \times U$.
Question	Is there a hitting set $H \subseteq U$ of size $k + \lvert I\rvert$ such that $I \subseteq H$, $H \cap O = \emptyset$ and for every $S_i \in S$, $H \cap (S_i \setminus \{u \in U \mid (u, S_i) \in A\}) \neq \emptyset$?

Dominating Set Completion (DS-C)	
Instance	An instance of Dominating Set consisting of a graph $G = (V, E)$ and a number k, together with sets $I, O, S \subseteq V$.
Question	Is there a dominating set D for the induced subgraph $G[V \setminus S]$ such that D is of size $k + \lvert I\rvert$, $I \subseteq D$ and $D \cap O = \emptyset$?

Complexity results

Proposition (Prop. 4.2 and 4.5)

$(\mathsf{HS\text{-}C}, k)$ *and* $(\mathsf{DS\text{-}C}, k)$ *are* W[2]*-complete.*

Theorem (Corollary 4.8)

$(\mathsf{HS\text{-}C}, \langle U, S, k \rangle)$ and $(\mathsf{DS\text{-}C}, \langle G, k \rangle)$ are both **chopped**-**NP**-complete.

Theorem (Corollary 4.9)

$(\mathsf{HS\text{-}C}, \langle U, S, k \rangle, k)$ *is* **chopped**-W[2]*-complete.*

$(\mathsf{DS\text{-}C}, \langle G, k \rangle, k)$ *is* **chopped**-W[1]*-hard and in* **chopped**-W[2].