The Complexity of Identifying Characteristic Formulae*†

Luca Aceto[1,2], Antonis Achilleos[1], Adrian Francalanza[3], and Anna Ingólfsdóttir[1]

[1] School of Computer Science, Reykjavik University, Reykjavik, Iceland
[2] Gran Sasso Science Institute, L'Aquila, Italy
[3] Dept. of Computer Science, ICT, University of Malta, Msida, Malta

Abstract

We examine the complexity of determining whether a modal formula (possibly with recursion operators) characterizes a process up to bisimulation equivalence.

1 Introduction

Characteristic formulae are formulae that characterize a process up to some notion of behavioural equivalence or preorder, which in our case is bisimilarity: a formula φ is characteristic for a process p when every process q is bisimilar to p exactly when it satisfies φ. A construction of characteristic formulae for variants of CCS processes [9] was introduced in [6]. This construction allows one to verify that two CCS processes are equivalent by reducing this problem to model checking. Similar constructions were studied later in, for instance, [1, 10, 12].

We are interested in detecting when a formula is characteristic for a certain process. We call this the characterization problem and we determine its complexity. We focus on a representative collection of logics, including a selection of modal logics without recursion and maxHML, the max-fragment of μHML [8], a variant of the μ-calculus [7], which consists of the μHML formulae that only use greatest fixed points. These formulae are sufficient to provide characteristic formulae for any state in a finite labelled transition system.

Similar to the characterization problem is the completeness problem, which asks whether a given formula is complete, meaning that any two processes that satisfy it are bisimilar to each other. Therefore, a complete formula is characteristic if and only if it is satisfiable. As we see in the following sections, the completeness problem tends to have the same complexity as validity. The techniques that we use to determine the complexity of completeness for modal logics without fixed points were presented in [2], but we extend these techniques for the case of maxHML and show that the completeness problem for this fragment is EXP-complete. The EXP-completeness of the characterization problem is an immediate corollary.

2 Background

Definition 1. *The formulae of that we consider are constructed using the following grammar:*

$$\varphi, \psi ::= p \mid \neg p \mid \mathit{tt} \mid \mathit{ff} \mid X \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \langle \alpha \rangle \varphi \mid [\alpha]\varphi \mid \mu X.\varphi \mid \nu X.\varphi$$

where X comes from a countably infinite set of logical variables LVar*, α from a finite set of actions,* Act*, and p from a countable set of propositional variables, P.*

*This extended abstract is partly based on results from [2].

†This research was supported by the project "TheoFoMon: Theoretical Foundations for Monitorability" (grant number: 163406-051) and the project "Epistemic Logic for Distributed Runtime Monitoring" (grant number: 184940-051) of the Icelandic Research Fund.

We interpret formulae on the states of a labelled transition system (LTS). An LTS is a quadruple $\langle \textsc{Proc}, \textsc{Act}, \rightarrow, V \rangle$ where $\textsc{Proc}$ is a set of states or processes, $\textsc{Act}$ is the set of actions, $\rightarrow \subseteq \textsc{Proc} \times \textsc{Act} \times \textsc{Proc}$ is a transition relation, and $V : P \rightarrow 2^{\textsc{Proc}}$ determines on which states a propositional variable is true. We assume that our LTS contains all the possible finite behaviours and only those. State `nil` represents any state that cannot transition anywhere: $\forall \alpha \forall s. \texttt{nil} \stackrel{\alpha}{\not\rightarrow} s$. The size of a state s is $|s|$, the number of states that can be reached from s by any sequence of transitions, and $|\varphi|$ is the length of φ as a string of symbols. All our complexity results are with respect to these measures.

Formulae are evaluated in the context of an LTS and an environment, $\rho : \textsc{LVar} \rightarrow 2^{\textsc{Proc}}$, which gives values to the logical variables. For an environment ρ, variable X, and set $S \subseteq \textsc{Proc}$, $\rho[X \mapsto S]$ is the environment which maps X to S and all $Y \neq X$ to $\rho(Y)$. The semantics for our formulae is given through a function $[\![\cdot]\!]$:

$$[\![\texttt{tt}, \rho]\!] = \textsc{Proc}, \quad [\![\texttt{ff}, \rho]\!] = \emptyset, \quad [\![p, \rho]\!] = V(p), \quad [\![\neg p, \rho]\!] = \textsc{Proc} \setminus V(p), \quad [\![X, \rho]\!] = \rho(X)$$

$$[\![\varphi_1 \wedge \varphi_2, \rho]\!] = [\![\varphi_1, \rho]\!] \cap [\![\varphi_2, \rho]\!] \qquad [\![[\alpha]\varphi, \rho]\!] = \left\{ s \mid \forall t.\ s \stackrel{\alpha}{\rightarrow} t \text{ implies } t \in [\![\varphi, \rho]\!] \right\}$$

$$[\![\varphi_1 \vee \varphi_2, \rho]\!] = [\![\varphi_1, \rho]\!] \cup [\![\varphi_2, \rho]\!] \qquad [\![\langle \alpha \rangle \varphi, \rho]\!] = \left\{ s \mid \exists t.\ s \stackrel{\alpha}{\rightarrow} t \text{ and } t \in [\![\varphi, \rho]\!] \right\}$$

$$[\![\nu X.\varphi, \rho]\!] = \bigcup \{ S \mid S \subseteq [\![\varphi, \rho[X \mapsto S]]\!] \} \quad [\![\mu X.\varphi, \rho]\!] = \bigcap \{ S \mid S \supseteq [\![\varphi, \rho[X \mapsto S]]\!] \}$$

A formula is closed when every occurrence of a variable X is in the scope of recursive operator νX or μX. Henceforth we consider only closed formulae. As the environment has no effect on the semantics of a closed formula φ, we write $s \models \varphi$ for $s \in [\![\varphi, \rho]\!]$. Depending on how we further restrict our syntax, and the LTS, we can describe several logics. Without further restrictions, the resulting logic is the μ-calculus. If we do not allow any propositional variables, the resulting logic is μHML, and if we further disallow the recursive operators, the resulting logic is HML. If we allow propositional variables, but only one action and no recursive operators (or recursion variables), then we have the basic modal logic **K**, and further restrictions on the LTS can result in a wide variety of modal logics. For example, logic **D** has all the restrictions of **K**, but furthermore, `nil` is not allowed as a state, while for logic **S4**, the transition relation must be reflexive and transitive — for more on Modal Logic, see [3, 4]. For convenience and brevity, we examine the logics HML, $\mathbf{D}^-$ that has all the restrictions of both HML and **D**, and maxHML that allows only the formulae of μHML that do not use the operator μX. For more details on our techniques — mostly on non-recursive logics — the reader can see [2].

In the context of these logics, we call a formula φ *characteristic* for state s when $s \models \varphi$ and for every state t, $s \sim t$ if and only if $t \models \varphi$, where $\sim$ stands for bisimilarity without accounting for variables. The characterization problem is the following: *Given a formula φ and a state s, is φ characteristic for s?* A formula φ is called *complete* when for all states s and t, if $s \models \varphi$ and $t \models \varphi$, then $s \sim t$. The completeness problem is: *Given a formula φ, is φ complete?*

3 The Complexity of Completenes and Characterization

Proposition 1. *The completeness and characterization problems for $\boldsymbol{D}^-$ are in P.*

Proof. For $\mathbf{D}^-$, all states are bisimilar, so all formulae are complete; for more, see [2]. □

It is known that satisfiability for the min-fragment of the μ-calculus (on one action) is EXP-complete. It is in EXP, as so is the satisfiability problem of the μ-calculus [7]. Furthermore, this fragment suffices [11] to describe the PDL formula that is constructed by the reduction

used in [5] to prove EXP-hardness for PDL, therefore the reduction can be adjusted to prove that the min-fragment of the μ-calculus is EXP-complete. Therefore, validity for the min- and max-fragments of the μ-calculus (on one action) is EXP-complete. To see that this lower bound transfers to minHML and maxHML, it suffices to use one extra action to represent propositional variables (for example, variable x_i can be replaced by $\langle\alpha\rangle^i\mathtt{tt}$).

Proposition 2. *The completeness problems for HML and* maxHML *are* PSPACE*-hard and* EXP*-hard, respectively.*

Proof. We prove the theorem for the case of μHML by a reduction from minHML-validity. First, notice that $\bigwedge_{\alpha\in\text{Act}}[\alpha]\mathtt{ff}$ is complete and it is satisfiable by process `nil`. Given a minHML-formula φ, there are two cases. If $\mathtt{nil} \models \neg\varphi$, then φ is not valid and we set $\varphi_c = \mathtt{tt}$. Otherwise, let $\varphi_c = \neg\varphi \vee \bigwedge_{\alpha\in\text{Act}}[\alpha]\mathtt{ff}$. For the second case, if φ is valid, then φ_c is equivalent to $\bigwedge_{\alpha\in\text{Act}}[\alpha]\mathtt{ff}$, which is complete; if φ_c is complete, then only `nil` satisfies it and therefore, φ is valid. Thus, in both cases, φ is valid if and only if φ_c is complete. □

Theorem 3. *The completeness problems for HML and* maxHML *are* PSPACE*-complete and* EXP*-complete, respectively.*

Notice that in the proof of Proposition 2, the reduction could have returned both φ_c and `nil`, instead of just φ_c. Furthermore, as model-checking has a lower complexity than completeness, checking if φ is characteristic of s is at most as hard as checking if φ is complete. Therefore, the same complexity bounds hold for the characterization problem of HML and maxHML.

References

[1] Luca Aceto, Anna Ingolfsdottir, Paul Blain Levy, and Joshua Sack. Characteristic formulae for fixed-point semantics: a general framework. *Mathematical Structures in Computer Science*, 22(02):125–173, 2012.

[2] Antonis Achilleos. The completeness problem for modal logic. In *Logical Foundations of Computer Science*, Lecture Notes in Computer Science, pages 1–21. Springer, 2018.

[3] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[4] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge*. The MIT Press, 1995.

[5] Michael J Fischer and Richard E Ladner. Propositional dynamic logic of regular programs. *Journal of computer and system sciences*, 18(2):194–211, 1979.

[6] S. Graf and J. Sifakis. A modal characterization of observational congruence on finite terms of CCS. *Information and Control*, 68(1-3):125–145, January 1986.

[7] Dexter Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27(3):333–354, 1983.

[8] Kim Guldstrand Larsen. Proof systems for satisfiability in Hennessy-Milner logic with recursion. *Theoretical Computer Science*, 72(2&3):265–288, 1990.

[9] R. Milner. *Communication and concurrency*. Prentice-Hall, Inc., 1989.

[10] Markus Mller-Olm. Derivation of characteristic formulae. *Electronic Notes in Theoretical Computer Science*, 18:159–170, 1998.

[11] V. R. Pratt. A decidable mu-calculus: Preliminary report. In *22nd Annual Symposium on Foundations of Computer Science (sfcs 1981)*. IEEE, oct 1981.

[12] Bernhard Steffen and Anna Ingólfsdóttir. Characteristic formulas for processes with divergence. *Information and Computation*, 110(1):149–163, 1994.