The Complexity of Identifying Characteristic Formulas for μHML

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

[1] School of Computer Science, Reykjavik University, Reykjavik, Iceland
[2] Dept. of Computer Science, ICT, University of Malta, Msida, Malta

Abstract

We examine the complexity of determining whether a formula of the maximal-fixed-point fragment of the Hennessy Milner logic with recursion characterizes a process up to bisimulation equivalence. We discover that the problem is EXP-complete. The decision procedure that establishes this upper bound is based on a two-player game.

1 Introduction

Characteristic formulas are formulas that characterize a process's equivalence class with respect to an equivalence relation, 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 formulas for variants of CCS processes [10] was introduced in [7]. This construction allows one to verify that two CCS processes are equivalent by reducing this problem to model checking, for which efficient software tools exist. Similar constructions were studied later in [11,13] for instance and in a more general manner in [1,2].

We are interested in detecting when a formula is characteristic for a certain process. We call this the characterization problem and we examine its complexity. We focus on the max-fragment of μHML [9], a variant of the μ-calculus [8], which consists of the μHML formulas which only use maximal fixed points, as these formulas are sufficient to provide characteristic formulas for finite-state LTSs. One way to determine whether a formula is characteristic for a process would be to extract a characteristic formula for the model and then check whether it is equivalent to the given formula. One would hope, however that a given characteristic formula may be significantly smaller than the formulas one would get from a construction from [7,11,13], and thus hope for a more efficient method.

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. The complexity of the completeness problem was studied in [3] for modal logics without fixed points. There, it was determined that completeness tends to have the same complexity as validity. We apply similar techniques as in [3] for the case of the max-fragment of μHML to prove that the completeness problem for this fragment is EXP-complete. The EXP-completeness of the characterization problem is an immediate corollary. Although we focus on μHML, it is not hard to see that our methods also apply to the corresponding, maximal-fixed-point, fragment of the μ-calculus.

Fine [5] introduced normal forms for modal logic and they can be used to prove soundness, completeness, and the finite frame property for several modal logics with respect to their classes of frames. Normal forms completely describe the behavior of a Kripke model up to a certain distance from a state. Therefore, normal forms are closely related the constructions from [7] which use formulas similar to HML without recursion.

2 Background

We present the necessary background. We first give the syntax and semantics of μHML.

Definition 1. *The formulae of μHML are constructed using the following grammar:*

$$\begin{array}{lllll} \varphi, \psi \in \mu HML ::= & \mathtt{tt} & \mid \ \mathtt{ff} & \mid \ X \\ & \mid \ \varphi \wedge \psi & \mid \ \varphi \vee \psi & \\ & \mid \ \langle \alpha \rangle \varphi & \mid \ [\alpha] \varphi & \\ & \mid \ \min X.\varphi & \mid \ \max X.\varphi & \end{array}$$

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

We interpret formulas on *processes*, which are states of a labeled transition system (LTS). A labeled transition system is a triple

$$\langle \textsc{Proc}, \textsc{Act}, \rightarrow \rangle$$

where Proc is a set of states or processes, Act is a set of actions, and $\rightarrow \subseteq \textsc{Proc} \times \textsc{Act} \times \textsc{Proc}$ is a transition relation. For our LTS, we assume that for any processes p, q and action α, there are processes which we call $\mathtt{nil}$, $\alpha.p$, and $p + q$, so that for every $\beta \in \textsc{Act}$ and process r, $\mathtt{nil} \stackrel{\beta}{\not\rightarrow} r$; $\alpha.p \stackrel{\beta}{\rightarrow} p$ iff $\alpha = \beta$; and $p + q \stackrel{\alpha}{\rightarrow} r$ iff $p \stackrel{\alpha}{\rightarrow} r$ or $q \stackrel{\alpha}{\rightarrow} r$. In our setting, $|p|$, the size of a process is the number of processes that can be reached from p through any sequence of transitions.

Formulae are evaluated in the context of a labeled transition system and an environment, $\rho : \textsc{LVar} \rightarrow 2^{\textsc{Proc}}$, which gives values to the logical variables in the formula. 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 μHML formulae is given through a function $[\![\cdot]\!]$, which, given an environment ρ, maps each formula to a set of processes — namely the processes which satisfy the formula under the assumption that each $X \in \textsc{LVar}$ is satisfied by the processes in $\rho(X)$. $[\![\cdot]\!]$ is defined as follows:

$$\begin{aligned} [\![\mathtt{tt}, \rho]\!] &= \textsc{Proc} \quad \text{and} \quad [\![\mathtt{ff}, \rho]\!] = \emptyset \\ [\![\varphi_1 \wedge \varphi_2, \rho]\!] &= [\![\varphi_1, \rho]\!] \cap [\![\varphi_2, \rho]\!] \\ [\![\varphi_1 \vee \varphi_2, \rho]\!] &= [\![\varphi_1, \rho]\!] \cup [\![\varphi_2, \rho]\!] \\ [\![[\alpha]\varphi, \rho]\!] &= \left\{ p \mid \forall q.\ p \stackrel{\alpha}{\rightarrow} q \text{ implies } q \in [\![\varphi, \rho]\!] \right\} \\ [\![\langle \alpha \rangle \varphi, \rho]\!] &= \left\{ p \mid \exists q.\ p \stackrel{\alpha}{\rightarrow} q \text{ and } q \in [\![\varphi, \rho]\!] \right\} \\ [\![\max X.\varphi, \rho]\!] &= \bigcup \{ S \mid S \subseteq [\![\varphi, \rho[X \mapsto S]]\!] \} \\ [\![\min X.\varphi, \rho]\!] &= \bigcap \{ S \mid S \supseteq [\![\varphi, \rho[X \mapsto S]]\!] \} \\ [\![X, \rho]\!] &= \rho(X). \end{aligned}$$

A formula is closed when every occurrence of a variable X is in the scope of recursive operator max X or min X. Note that the environment, ρ, has no effect on the semantics of a closed formula. For a closed formula φ, we often drop the environment from the notation for $[\![\cdot]\!]$ and write $[\![\varphi]\!]$ instead of $[\![\varphi, \rho]\!]$. Henceforth we work only with closed formulas, unless stated

otherwise. Formulae φ and ψ are (logically) equivalent, written $\varphi \equiv \psi$, if $[\![\varphi, \rho]\!] = [\![\psi, \rho]\!]$ for every environment ρ. For a process p and formula φ, we may also write $p \models \varphi$ instead of $p \in [\![\varphi]\!]$. We can define negation $\neg\varphi$ by recursively using de Morgan laws and implication $\varphi \to \psi$ in the usual way — only $\neg X$ is X, as recursion variables are always positive. The size of a formula is the number of symbols we use to write it. The set $sub(\varphi)$ is the set of all subformulas of φ; we also define $\overline{sub(\varphi)} = sub(\varphi) \cup \{\neg\psi \mid \psi \in sub(\varphi)\}$. In general, $|\overline{sub(\varphi)}| \leq 2|\varphi|$.

The max-fragment of μHML, which we call maxHML, consists of all formulas of μHML that do not use the minimal fixed point operator, $\min X$; similarly, the min-fragment of μHML, which we call minHML, consists of all formulas of μHML that do not use the maximal fixed point operator, $\max X$. It is known that model-checking for these fragments can be done in polynomial time [4].

A relation $\mathcal{R} \subseteq \text{Proc} \times \text{Proc}$ is a *bisimulation* when the following two conditions are satisfied for all $(p, q) \in \mathcal{R}$:

- For every process p and action α, if $p \xrightarrow{\alpha} p'$, then there exists a process q', such that $q \xrightarrow{\alpha} q'$ and $(p', q') \in \mathcal{R}$.
- For every process q and action α, if $q \xrightarrow{\alpha} q'$, then there exists a process p', such that $p \xrightarrow{\alpha} p'$ and $(p', q') \in \mathcal{R}$.

We call processes p and q bisimilar and write $p \sim q$ if there is a bisimulation $\mathcal{R}$ such that $p\mathcal{R}q$.

Completeness and Characterization

We call a formula φ *characteristic* for a process p when $p \models \varphi$ and for every process q, $p \sim q$ if and only if $q \models \varphi$. The characterization problem is the following: *Given a* maxHML*-formula φ and a process p, is φ characteristic for p?* A formula φ is called *complete* when for all processes p and q, if $p \models \varphi$ and $q \models \varphi$, then $p \sim q$. The completeness problem is the following: *Given a* maxHML*-formula φ, is φ complete?*

3 Lower Bounds

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 [8]. Furthermore, this fragment suffices [12] to describe the PDL formula that is constructed by the reduction used in [6] to prove EXP-hardness for PDL, therefore the reduction can be adjusted to prove that min-fragment of the μ-calculus is EXP-complete. We can similarly alter the PDL formula of the reduction in [6] so that it can be described by the min-fragment of the μ-calculus. 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 maxHML, it suffices to use one extra action to represent propositional variables (for example, variable x_i can be replaced by $\langle\alpha\rangle^i\mathtt{tt}$). We now demonstrate that completeness for maxHML is also EXP-hard, with a similar method as in [3]:

Proposition 1. *The completeness problem for* maxHML *is* EXP*-hard.*

Proof. To prove the theorem we present a reduction from minHML-validity to the completeness problem for maxHML. 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 $\mathtt{nil}$ satisfies it and therefore, φ is valid. Thus, in both cases, φ is valid if and only if φ_c is complete. □

Notice that in the proof of Proposition 1, the reduction could have returned both φ_c and $\mathtt{nil}$, instead of just φ_c. Therefore, the same lower bound holds for the characterization problem:

Corollary 2. *The characterization problem for* maxHML *is* EXP*-hard.*

4 A Game to Determine Completeness for μHML

The game is played on a fixed maxHML-formula φ. A *decomposition* is a maximally consistent subset of $\overline{sub}(\varphi)$. D is called a *decomposition of* $\psi \in \overline{sub}(\varphi)$ when D is a decomposition and $\psi \in D$. The players of the game are called E and B. A play of the game on maxHML-formula φ is a (finite or infinite) sequence $\pi = E_0B_0E_1B_1E_2B_2\cdots$, constructed in the following way:

- At first, E picks a decomposition D of φ and $E_0 = B_0 = (D, \varphi, \emptyset)$.
- For every $i \geq 0$, for $B_i = (D_1, \psi, S)$, E picks a decomposition $D_2 \supseteq S$ of ψ, a $j \in \{1, 2\}$, an $\alpha \in \text{Act}$, and for $b = \{\psi \mid [\alpha]\psi \in D_j\}$, a decomposition $D \supseteq b$; $E_{i+1} = (D_{2-j}, \alpha, D)$ is move $(i + 1)$ of E.
- Then, B or picks a formula $\langle\alpha\rangle\psi' \in D_{2-j}$ and $B_{i+1} = (D, \psi', \{\psi \mid [\alpha]\psi \in D_{2-j}\})$ which is move $(i + 1)$ for B.

Player E wins when at some point B has no move to play, because there is no formula $\langle\alpha\rangle\psi \in D_{2-i}$ to choose from, thus the play is finite and is called a winning play for E. Player B wins in one of two ways: either E does not have a satisfiable decomposition to pick from (so, the game is finite), or the game is infinite; in both cases, the play is called a winning play for B.

A strategy for E on φ is an initial move E_0 and a partial function, which, given a B_i that may occur in a play on φ, returns an appropriate move E_{i+1} for E; similarly, strategy for B on φ is a partial function, which, given an E_i that may occur in a play, returns an appropriate move B_i for B. A strategy f for E is called a winning strategy for E on formula φ when for every play $\pi = E_0B_0E_1B_1\cdots$ on φ, if E_0 is the initial move given by the strategy and for every B_i, E_{i+1} in the play, $E_{i+1} = f(B_i)$, then π is a winning play for E; similarly, we define a winning strategy for B.

We can immediately observe the following:

Lemma 3. *Given a* maxHML*-formula φ, at most one player has a winning strategy on φ.*

Theorem 4. *Given a* maxHML*-formula φ, player B has a winning strategy on φ if and only if φ is complete. Furthermore, player E has a winning strategy on φ if and only if B does not.*

Proof. It suffices to prove that

(a) If φ is incomplete, then E has a winning strategy; and

(b) if φ is complete, then B has a winning strategy.

To prove (a), we first assume that φ is incomplete. Therefore, there are two non-bisimilar processes p_0 and q_0 that satisfy φ. We describe a family of strategies for E, using p_0 and q_0, which ensure that for every $B_i = (D, \psi)$ in the resulting play, there are non-bisimilar processes p_i, q_i, so that $p_i \models \bigwedge D$ and $q_i \models \psi$.

This condition is already true for B_0, so we maintain it recursively for every B_i in the play. Let $B_i = (D, \psi)$ in the play, so that there are non-bisimilar processes p_i, q_i, where $p_i \models \bigwedge D_1$ and $q_i \models \psi$. Then, there is a decomposition D_2 satisfied in q_i; since p_i and q_i are non-bisimilar, there is a process p_i' and action $\alpha \in \text{Act}$, such that $p_i \xrightarrow{\alpha} p_{i+1}$ or $q_i \xrightarrow{\alpha} p_{i+1}$ and for every $q_i \xrightarrow{\alpha} q'$ or, respectively $p_i \xrightarrow{\alpha} q'$, p_{i+1} is non-bisimilar to q'. Therefore, for $j = 1$, p_{i+1} satisfies b (as in the game description), so there is a decomposition D of $\bigwedge b$, that is satisfied by p_{i+1}. Player E's move is then (D_{2-j}, α, D). Now, either B has no move to play, so E wins, or B chooses $\langle\alpha\rangle\psi$; there is some $q_{i+1} \models \psi$, which is non-bisimilar to p_{i+1}.

Since there is always a satisfiable decomposition for E to play, if E does not win with this strategy, then the resulting play is infinite. The resulting strategy depends on the choices of p_i and q_i. Let relation $\mathcal{R}$ on processes be such that pRq if and only if for one of the strategies as described above, there is an i so that $p_i = p$ and $q_i = q$. If for every choice for p_i for the strategies described above, there is a choice for q_i so that the game continues, then $\mathcal{R}$ is a bisimulation — a contradiction, because p_0 and q_0 are non-bisimilar.

To prove (b), we now assume that φ is complete. The strategy for B ensures that for every $B_i = \{D', \psi, S\}$ in the game, $\bigwedge D'$ and $\psi \wedge \bigwedge S$ are complete and equivalent to each other. Lets assume that is the case for B_i and $E_{i+1} = (D', \alpha, D)$. Then, there is a decomposition D'' which is equivalent to $\psi \wedge \bigwedge S$ and complete and for which, $D \supseteq \{\psi' \mid [\alpha]\psi' \in D''\}$. Since D'' is complete, $\psi \wedge \bigwedge S \to \langle\alpha\rangle \bigwedge D$ is valid; therefore, so is $\bigwedge D' \to \langle\alpha\rangle \bigwedge D$. Therefore, there is a $\langle\alpha\rangle\psi' \in D'$, such that $\psi' \wedge \bigwedge\{\chi \mid [\alpha]\chi \in D'\}$ is equivalent to D.

We prove that D is complete. To reach a contradiction, assume that D is incomplete. Then it would be satisfiable in two non-bisimilar processes p and q. For every $\langle\alpha\rangle\chi \in D''$ not satisfiable in any of p, q, let $p_\chi \models \chi$ and for process r that satisfies D'', let

$$r_p = \alpha.p + \sum_{p_\chi} p_\chi + \sum_{\substack{\beta \neq \alpha \\ r \xrightarrow{\beta} r'}} r' \text{ and}$$

$$r_q = \alpha.q + \sum_{p_\chi} p_\chi + \sum_{\substack{\beta \neq \alpha \\ r \xrightarrow{\beta} r'}} r'.$$

Then, r_p and r_q satisfy D'' and are non-bisimilar, which is a contradiction. As we see above, as B maintains this condition, then there is always a move to play. Then, every play for which B follows such a strategy is either finite because B wins at some point, or infinite, so B wins as well. □

Corollary 5. *The completeness problem for* maxHML *is* EXP*-complete.*

Proof. The lower bound is provided by Proposition 1. For the upper bound, we use an alternating polynomial-space algorithm with an oracle from EXP. Given a maxHML-formula φ, the algorithm constructs a play of the game on φ move-by-move, keeping in memory only the previous move. For the moves of E it uses a universal choice and for the moves of B an existential choice. The oracle to EXP is used to verify that each decomposition is, indeed, satisfiable. □

Corollary 6. *The characterization problem for* maxHML *is* EXP*-complete.*

Proof. The lower bound is provided bt Corollary 2. For the upper bound, given a formula φ and process p, we can verify that $p \models \varphi$ and that φ is complete; the first can be done in polynomial time and the second in exponential time. □

5 Future Work

There are several relevant equivalence relations and preorders on processes that one may want to characterize, as well as many other corresponding logics — see [1,2], for a general treatment. Therefore, there are many more cases in which our methods might be applied. It is an interesting phenomenon that the complexity of completeness/characterization tends to be identical to the complexity of validity, despite the fact that we have not used any reduction to validity to solve these problems. It would be interesting to see whether there is a general observation to make concerning this phenomenon.

It would perhaps be even more important than our current results to be able to extract characteristic formulas from processes in an efficient manner. So far, this characteristic formula extraction has been used as a link between behavioral equivalence and model checking. Ideally, for certain cases we could thus reduce behavioral equivalence to model checking in an efficient manner, perhaps using the tools of modern algorithmic and complexity theory that tackle intractable problems.

References

[1] L. Aceto, D. Della Monica, I. Fábregas, and A. Ingólfsdóttir. *When Are Prime Formulae Characteristic?*, pages 76–88. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[2] 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.

[3] Antonis Achilleos. The completeness problem for modal logic. *CoRR*, abs/1605.01004, 2016.

[4] E Allen Emerson and Chin-Laung Lei. Efficient model checking in fragments of the propositional mu-calculus. In *IEEE Symposium on Logic in Computer Science*, pages 267–278. IEEE Computer Society Press, 1986.

[5] Kit Fine. Normal forms in modal logic. *Notre Dame journal of formal logic*, 16(2):229–237, 1975.

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

[7] 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.

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

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

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

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

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

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