On Universally Easy Classes for NP-complete Problems ⋆

Erik D. Demaine [a], Alejandro López-Ortiz [b], J. Ian Munro [b]

[a] *MIT Laboratory for Computer Science, 200 Technology Square, Cambridge, MA 02139, USA*

[b] *Department of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada*

Abstract

We explore the natural question of whether all **NP**-complete problems have a common restriction under which they are polynomially solvable. More precisely, we study what languages are *universally easy* in that their intersection with any **NP**-complete problem is in **P** (*universally polynomial*) or at least no longer **NP**-complete (*universally simplifying*). In particular, we give a polynomial-time algorithm to determine whether a regular language is universally easy. While our approach is language-theoretic, the results bear directly on finding polynomial-time solutions to very broad and useful classes of problems.

Key words: Complexity theory, polynomial time, NP-completeness, classes of instances, universally polynomial, universally simplifying, regular languages

1 Introduction and Overview

It is well-known that many **NP**-complete problems, when restricted to particular classes of instances, yield to polynomial-time algorithms. For example, COLOURING, CLIQUE and INDEPENDENT SET are classic **NP**-complete problems that have polynomial-time solutions when restricted to interval graphs [9]. But this property of interval graphs is not universal: graph list coloring and determining the existence of k vertex-disjoint paths (where k is part of the input) remain **NP**-complete for interval graphs [1,8].

⋆ A preliminary version of this paper appeared at SODA 2001 [3].

Email addresses: `edemaine@mit.edu` (Erik D. Demaine), `alopez-o@uwaterloo.ca` (Alejandro López-Ortiz), `imunro@uwaterloo.ca` (J. Ian Munro).

To better understand this behavior, we introduce the notion of *universally easy* classes of instances for **NP**-complete problems. It turns out that such languages exist, and it seems difficult to give a complete characterization. Thus we focus on two natural classes of languages: regular languages and context-free languages. In particular, we characterize precisely which regular languages are universally easy in the sense defined in Section 2.

Many classes of restrictions have been studied before; see for example Brandstadt, Le, and Spinrad [2] for a detailed survey of graph classes.

2 Definitions

For simplicity of exposition, assume that the alphabet $\Sigma = \{0,1\}$. We use interchangably the notions of a language, a decision problem, and a class of instances.

Definition 2.1 *The* restriction *of a problem P to a class of instances C is the intersection $P \cap C$.*

Definition 2.2 *Given an* **NP***-complete problem P, a language $C \in \mathbf{NP}$ is a* simplifying restriction *if the restriction of P to C is not* **NP***-complete; and a language $C \in \mathbf{P}$ is a* polynomial restriction *if the restriction of P to C is in* **P**.

Of course, this definition is trivial if $\mathbf{P} = \mathbf{NP}$.

Definition 2.3 *A language $C \in \mathbf{NP}$ is* universally simplifying *if it is a simplifying restriction of all* **NP***-complete problems.*

Definition 2.4 *A language $C \in \mathbf{P}$ is* universally polynomial *if it is a polynomial restriction of all* **NP***-complete problems.*

Informally, we use the term *universally easy* to refer to either notion, universally simplifying or universally polynomial.

3 Easy Languages

A natural question is whether there exist universally simplifying languages if $\mathbf{P} \neq \mathbf{NP}$. This can be readily answered in the affirmative by noticing that all finite languages are universally polynomial, which is not very enlightening. A more general class to consider is regular languages, which can be characterized according to their density.

Definition 3.1 *The* growth function *of a language L is the function $\gamma_L(n) = |\{x \in L : |x| \leq n\}|$. A language is* sparse *if its growth function is bounded from above by a polynomial, and is* exponentially dense *if the growth function is bounded from below by $2^{\Omega(n)}$.*

Theorem 3.1 *Any sparse language is either universally simplifying or universally polynomial. If $\mathbf{P} \neq \mathbf{NP}$, it must be universally simplifying.*

Proof: Consider a sparse language L. If it is universally simplifying, there is nothing to show. If it is not universally simplifying, there is a problem $P \subseteq \Sigma^*$ such that the restriction $P \cap L$ is **NP**-complete. Because $P \cap L \subseteq L$, this restriction is also a sparse set, and it is **NP**-complete. Mahaney [7] proved that if a language is sparse and **NP**-complete, then $\mathbf{P} = \mathbf{NP}$. Therefore $\mathbf{P} = \mathbf{NP}$ and consequently $P \cap L \in \mathbf{P}$ for all **NP**-complete languages L. □

Definition 3.2 *A* cycle *in a DFA A is a directed cycle in the state graph of A.*

Definition 3.3 *Let C_1 and C_2 be two cycles in a DFA such that neither is a subgraph of the other. We say that C_1 and C_2* interlace *if there is an accepting computation path in the DFA containing the sequence $C_1 \cdots C_2 \cdots C_1$ or the sequence $C_2 \cdots C_1 \cdots C_2$. See Fig. 1.*

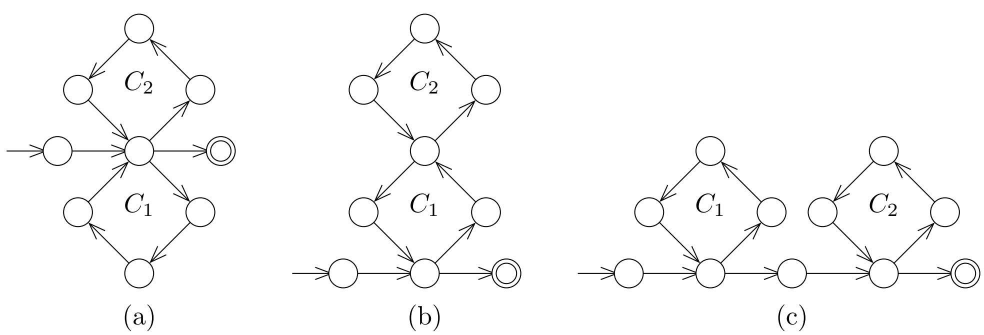

Fig. 1. Examples of DFAs with length-4 cycles C_1 and C_2 that (a–b) interlace and (c) do not interlace. The accepting state is denoted by a double circle.

The following theorem was proved by Flajolet [4]. Our proof uses a constructive argument needed for Theorem 3.3.

Theorem 3.2 *Every regular language is either sparse or exponentially dense.*

Proof: Consider $L \subseteq \Sigma^*$ recognized by a DFA A. If L is finite, then it is trivially sparse; otherwise, L is infinite and contains strings of arbitrary length. The pumping lemma states that any DFA accepting a sufficiently long string has at least one cycle in its state graph, which can be traversed (pumped) zero or more times.

If A has no interlacing cycles, then each accepting computation T_k can be written as

$$T_k = (s_1, t_1, s_2, t_2, \ldots, C_1^*, s_i, t_i, \ldots, C_j^*, \ldots, q_f),$$

where the s_i's are states, t_i's are input symbols causing transitions, C_i's are disjoint cycles, q_f is a final state of A, and $s_i \neq s_j$ for all $i \neq j$. Here s_i, t_i, s_{i+1} denotes the transition from state s_i to s_{i+1} upon reading symbol t_i. Notice that, apart from the actual value represented by the Kleene star, there are only finitely many such orderings of states and cycles, and thus the language L can be written as the finite union of T_k's. Let j_k denote the number of cycles and r_k the number of states in T_k. Then the total number of strings of length n generated by T_k is at most $\binom{n-r_k}{j_k} = O(n^{j_k})$. A union of finitely many such sets, each with a polynomially bounded number of strings of length n, is itself polynomially bounded and therefore sparse.

We now proceed to show that a DFA A with interlacing cycles accepts an exponentially dense language. Consider an accepting computation path T_k of A with interlacing cycles, that is,

$$T_k = (s_1, t_1, \ldots, C_1, \ldots, C_2, \ldots, C_1, \ldots, q_f).$$

Now we pump subsequences $(C_1, \ldots, C_2, \ldots)$, (C_1), and (C_2), and remove the second occurrence of C_1, obtaining

$$T_k' = (s_1, t_1, \ldots, [C_1^*, \ldots, C_2^*, \ldots]^*, \ldots, q_f).$$

We also remove any other cycles occurring in T_k' before or after the square brackets, so that no states are repeated on each side of the square brackets. We introduce the special character w_1 to denote the transitions in C_1 followed by any number of transitions (possibly zero) encompassed by the various "..." in T_k' above (but no C_2). Similarly we define w_2 in terms of C_2. Then T_k' can be rewritten as the regular expression $t_1 \cdots \{w_1, w_2\}^* \cdots t_f$. It follows that there are at least 2^{n-2r_k} strings T_k' of length n in $(\Sigma \cup \{w_1, w_2\})^*$. We are guaranteed that each w_1 expands to a string distinct from each w_2. Also, the lengths of w_1 and w_2 are both bounded above by the length of the original T_k. Thus $\gamma_L(n) \geq 2^{(n-2r_k)/|T_k|}$, which implies $\gamma_L(n) = 2^{\Omega(n)}$ as required. □

Theorem 3.3 *No exponentially dense regular language is universally simplifying.*

Proof: Let L be an exponentially dense regular language. From the proof of Theorem 3.2, we know that a DFA accepting L necessarily contains interlacing cycles. Furthermore, there is a computation path T_k with interlacing cycles of the form $T_k = (t_1 \cdots t_i \{w_1, w_2\}^* t_j \cdots t_f)$ where w_1 and w_2 are distinct. We define an injective polynomial-time transformation $F : \Sigma^* \to L$ as follows. Now we map 0 to w_1, and 1 to w_2. So a string $x_1 x_2 \cdots x_j \in \Sigma^*$ is mapped to $t_1 \cdots t_i w_{x_1+1} w_{x_2+1} \cdots w_{x_j+1} t_j \cdots t_f$. This transformation F and its inverse can be computed in polynomial time. (To compute the inverse of F, drop the leading i characters and the trailing $f - j + 1$ characters, and repeatedly extract a leading w_1 and w_2, preferring longer matches over shorter ones, and output the corresponding 0 or 1.)

Given any **NP**-complete language P, we define

$$\hat{P} = \{x \in L : x = F(y) \text{ for some } y \in P\}.$$

It follows that $\hat{P}$ is **NP**-complete, because the y's together with polynomial-length certificates from P serve as certificates for $\hat{P}$, and F is a reduction from P to $\hat{P}$. Because $\hat{P} \subseteq L$, we have $\hat{P} \cap L = \hat{P}$, which is **NP**-complete. Thus L is not universally simplifying. □

Corollary 3.1 *If an exponentially dense regular language is universally polynomial, then* $\mathbf{P} = \mathbf{NP}$.

Note that the property of interlacing cycles for regular languages, and hence "easiness", can be tested in polynomial time.

4 Extensions

Recently, the sparse/exponential-density property in Theorem 3.2 has been generalized to context-free languages [5,6]. In the original version of this paper [3], we conjectured that our results also generalize to context-free languages, the main obstruction being to find a polynomially constructive proof. Recently, Tran [10] extended our work to prove this conjecture, i.e., every universally simplifying context-free language is sparse. In addition, he establishes that, if **DEXT** = **NEXT**,[1] all sparse context-free (or regular) languages are universally polynomial; and if **DEXT** $\neq$ **NEXT**, only finite languages are universally polynomial. In the latter case of **DEXT** $\neq$ **NEXT**, we also have $\mathbf{P} \neq \mathbf{NP}$ [11, Cor. 24.3, p. 425], so every sparse language is universally simplifying.

Acknowledgments

We thank the anonymous referees for their helpful comments.

[1] **DEXT** is the class problems solvable in $2^{O(n)}$ deterministic time, and **NEXT** is the analogous class for nondeterministic time.

References

[1] Esther M. Arkin and Ellen B. Silverberg. Scheduling jobs with fixed start and end times. *Discrete Appl. Math*, 18(1):1–8, 1987.

[2] Andreas Brandstadt, Van Bang Le, and Jeremy P. Spinrad. *Graph Classes: A Survey.* SIAM Monographs on Discrete Mathematics and Applications, 1999.

[3] Erik D. Demaine, Alejandro López-Ortiz, and J. Ian Munro. On universally easy classes for NP-complete problems. *Proc. 12th Annual ACM-SIAM Sympos. Discrete Algorithms*, Washington, DC, January 2001, pp. 910-911.

[4] Philippe Flajolet. Analytic models and ambiguity of context-free languages. *Theoret. Comput. Sci.*, 49:283–309, 1987.

[5] Lucian Ilie, Grzegorz Rozenberg, and Arto Salomaa. A characterization of poly-slender context-free languages. *Theoret. Informatics Appl.*, 34(1):77–86, 2000.

[6] Roberto Incitti. The growth function of context-free languages. *Theoretic. Comput. Sci.*, 255:601–605, 2000.

[7] Stephen R. Mahaney. Sparse complete sets for NP: Solution of a conjecture of Berman and Hartmanis. *J. Comput. System Sci.*, 25(2):130–143, 1982.

[8] S. Natarajan and A. P. Sprague. Disjoint Paths in Circular Arc Graphs. *Nordic J. Comput.*, 3(3):256–270, Fall 1996.

[9] Christos H. Papadimitrou. *Computational Complexity.* Addison-Wesley, 1994.

[10] Nicholas Tran. On universally polynomial context-free languages. *Proc. 7th Annual Internat. Computing and Combinatorics Conf.*, vol. 2108 of Lecture Notes in Computer Science, Guilin, China, August 2001, pp. 21–27.

[11] K. Wagner and G. Wechsung. *Computational Complexity.* D. Reidel and Kluwer, 1986.