

# 5. The Self-Reducibility Method

Objekttyp: **Chapter**

Zeitschrift: **L'Enseignement Mathématique**

Band (Jahr): **28 (1982)**

Heft 1-2: **L'ENSEIGNEMENT MATHÉMATIQUE**

PDF erstellt am: **25.09.2024**

## **Nutzungsbedingungen**

Die ETH-Bibliothek ist Anbieterin der digitalisierten Zeitschriften. Sie besitzt keine Urheberrechte an den Inhalten der Zeitschriften. Die Rechte liegen in der Regel bei den Herausgebern.

Die auf der Plattform e-periodica veröffentlichten Dokumente stehen für nicht-kommerzielle Zwecke in Lehre und Forschung sowie für die private Nutzung frei zur Verfügung. Einzelne Dateien oder Ausdrucke aus diesem Angebot können zusammen mit diesen Nutzungsbedingungen und den korrekten Herkunftsbezeichnungen weitergegeben werden.

Das Veröffentlichen von Bildern in Print- und Online-Publikationen ist nur mit vorheriger Genehmigung der Rechteinhaber erlaubt. Die systematische Speicherung von Teilen des elektronischen Angebots auf anderen Servern bedarf ebenfalls des schriftlichen Einverständnisses der Rechteinhaber.

## **Haftungsausschluss**

Alle Angaben erfolgen ohne Gewähr für Vollständigkeit oder Richtigkeit. Es wird keine Haftung übernommen für Schäden durch die Verwendung von Informationen aus diesem Online-Angebot oder durch das Fehlen von Informationen. Dies gilt auch für Inhalte Dritter, die über dieses Angebot zugänglich sind.

Then  $G = S : h$  where  $S \in DSPACE((\log n)^l)$  and  $|h(x)| \leq k \log_2 |x|$ , for some  $k$ . Then  $x \in G \Leftrightarrow \exists w \forall w' \text{Win}(w, w', x)$ , where  $w$  and  $w'$  range over all strings of length  $\leq k \log_2 |x|$ . Clearly space  $O((\log n)^l)$  suffices to deterministically enumerate all pairs  $(w, w')$  and, for each, to play out  $\text{Strat}(w)$  against  $\text{Strat}(w')$  from position  $x$ , with the help of repeated calls on a deterministic space  $(\log n)^l$  recognizer for  $S$ . It follows that

$$G \in DSPACE((\log n)^l). \quad \blacksquare$$

## 5. THE SELF-REDUCIBILITY METHOD

The "hardest" problems in complexity classes defined by bounds on nondeterministic time or space often possess a structural property called *self-reducibility*. Various formal definitions of self-reducibility can be found in the literature ([12, 18, 20]). Here is one version of the idea. Let  $K$  be a subset of  $\{0, 1\}^*$ . A *self-reducibility structure* for  $K$  is specified by a partial ordering  $<$  of  $\{0, 1\}^*$  such that

- (i)  $A$ , the set of minimal elements in  $<$ , is recursive and
- (ii)  $A \cap K$  is recursive

together with a pair of computable functions  $G_0$  and  $G_1$  mapping  $\{0, 1\}^* - A$  into  $\{0, 1\}^*$ , such that, for all  $x \in \{0, 1\}^* - A$ ,

- (iii)  $G_0(x) < x$ ,  $G_1(x) < x$ ,  $|G_0(x)| = |G_1(x)| = |x|$   
and  $x \in K \Leftrightarrow G_0(x) \in K$  or  $G_1(x) \in K$ .

If  $K$  has a self-reducibility structure, then  $K$  is called *self-reducible*.

To illustrate the concept, we give self-reducibility structures for two important examples. The first example is the satisfiability problem for propositional formulas, encoded so that the following property holds: Let  $F(t_1, t_2, \dots, t_n)$  be a formula in which the variables  $t_1, t_1, \dots, t_n$  appear, and let  $F(a, t_2, \dots, t_n)$  be the same formula with the Boolean constant  $a$  substituted for  $t_1$ . Let  $\langle F(t_1, t_2, \dots, t_n) \rangle$  and  $\langle F(a, t_2, \dots, t_n) \rangle$  denote the encodings of these two formulas as strings. Then

$$|\langle F(t_1, t_2, \dots, t_n) \rangle| = |\langle F(a, t_2, \dots, t_n) \rangle|.$$

Let SAT denote this version of the satisfiability problem. The set SAT has a self-reducibility structure in which  $A$  is the set of propositional formulas containing no variables,

$$G_0(\langle F(t_1, t_2, \dots, t_n) \rangle) = \langle F(0, t_2, \dots, t_n) \rangle \text{ and}$$

$$G_1(\langle F(t_1, t_2, \dots, t_n) \rangle) = \langle F(1, t_2, \dots, t_n) \rangle .$$

As a second example, let DAG denote the set of encodings of triples  $(\Psi, s, t)$  such that

- (i)  $\Psi$  is a directed acyclic graph in which the out-degree of each vertex is either 0 or 2; if  $v$  has out-degree 2 then its successor vertices are denoted  $\sigma_0(v)$  and  $\sigma_1(v)$ ;
- (ii)  $s$  is a vertex and  $t$  is a vertex of out-degree 0;
- (iii) there exists a directed path from  $s$  to  $t$ .

Assume that, for any directed acyclic graph  $G$ , any vertex  $t$  of out-degree 0, and any two vertices  $v$  and  $w$ , the encodings of  $(\Psi, v, t)$  and  $(\Psi, w, t)$  are of the same length. Then DAG is clearly self-reducible. Let  $A$  be the set of triples  $(\Psi, s, t)$  such that  $s$  is of out-degree 0, and let  $G_0((\Psi, s, t)) = (\Psi, \sigma_0(s), t)$  and  $G_1((\Psi, s, t)) = (\Psi, \sigma_1(s), t)$ .

It is possible to relate the uniform complexity of a self-reducible set  $K$  to its nonuniform complexity. Suppose  $K$  has a self-reducibility structure  $(\langle, A, G_0, G_1)$  and  $K = S : h$ . For each  $w \in \{0, 1\}^*$  define  $reduct_w$ , a total function over  $\{0, 1\}^*$ , by the following recursive definition:

$$reduct_w(x) = \text{if } x \in A \text{ then } x \text{ else}$$

$$\text{if } w \cdot G_0(x) \in S \text{ then } reduct_w(G_0(x)) \text{ else}$$

$$reduct_w(G_1(x)).$$

Then, for all  $w$ ,  $reduct_w(x) \in A$ . Also,  $reduct_w(x) \in K \Rightarrow x \in K$  and  $x \in K \Leftrightarrow reduct_{h(|x|)}(x) \in K$ . These observations imply the following lemma.

LEMMA 5.1. Let  $w$  range over some set which includes  $h(|x|)$ . Then

$$x \in K \Leftrightarrow \exists w [reduct_w(x) \in K] .$$

Lemma 5.1 suggests a uniform way of testing membership in  $K$ : for each  $w$  in a suitable set, compute  $reduct_w(x)$  and test whether

$$reduct_w(x) \in A \cap K .$$

The complexity of this algorithm will depend on the time and space needed to test membership in  $A$ , and in  $A \cap K$ , on the lengths of chains in the

partial ordering  $<$ , and on the number of strings  $w$  that need to be considered.

Now we are ready to give some applications of self-reducibility.

**THEOREM 5.2.**  $P = NP \Leftrightarrow NP \subseteq P/\log$ .

*Proof.* The implication  $P = NP \Rightarrow NP \subseteq P/\log$  is immediate. Since SAT is  $NP$ -complete, the reverse implication will follow once we prove that

$$\text{SAT} \in P/\log \Rightarrow \text{SAT} \in P .$$

Assume that  $\text{SAT} \in P/\log$ . Then  $\text{SAT} = S : h$ , where  $S \in P$  and, for some  $k$ ,  $|h(n)| \leq k \log_2 n$ .

Using the self-reducibility structure for SAT given above, coupled with the method of lemma 5.1, we can test whether string  $x$  is in SAT. It is necessary to compute  $\text{reduct}_w(x)$  for each of the polynomially-many strings  $w$  of length  $\leq k \log_2 n$  and, for each, to test whether

$$\text{reduct}_w(x) \in A \cap K .$$

Each such computation can be done in polynomial time. Hence we conclude that  $\text{SAT} \in P$ . ■

By similar methods we can relate the nonuniform and uniform complexities of other self-reducible problems. For example, we can state the following result.

**THEOREM 5.3.** Let *Factor* denote the set of triples of integers  $\langle x, y, z \rangle$  such that  $x$  has a factor between  $y$  and  $z$ . Then

$$\text{Factor} \in P/\log \Leftrightarrow \text{Factor} \in P .$$

As another application of the self-reducibility method, we give the following theorem.

**THEOREM 5.4.**

$$\begin{aligned} \text{NSPACE}(\log n)/\log &\subseteq \text{DSPACE}(\log n)/\log \\ &\Leftrightarrow \text{NSPACE}(\log n) = \text{DSPACE}(\log n) . \end{aligned}$$

*Proof.* It is sufficient to prove

$$\begin{aligned} \text{NSPACE}(\log n)/\log &\subseteq \text{DSPACE}(\log n)/\log \\ &\Rightarrow \text{NSPACE}(\log n) = \text{DSPACE}(\log n) . \end{aligned}$$

Since DAG is logspace complete in  $NSPACE(\log n)$ , it suffices to show that

$$DAG \in DSPACE(\log n)/\log \Rightarrow DAG \in DSPACE(\log n).$$

Suppose that  $DAG = S : h$ , where  $S \in DSPACE(\log n)$  and

$$|h(n)| \leq k \log_2 n.$$

Then, guided by the self-reducibility of DAG, we can test whether  $(\Psi, s, t) \in DAG$  by performing the following computation for each string  $w$  of length  $\leq k \log_2 n$ :

$v := s$ ;

while  $v$  has out-degree 2 do

$v :=$  if  $w \cdot (\Psi, v_0, t) \in S$  then  $v_0$  else  $v_1$ .

If  $v$  is ever set equal to  $t$  then accept  $(\Psi, s, t)$ ; otherwise, reject it. It is clear that this method recognizes DAG deterministically within space  $O(\log n)$ . ■

## 6. THE METHOD OF RECURSIVE DEFINITION

Let  $K$  be a subset of  $\{0, 1\}^*$ , and let  $C_K: \{0, 1\}^* \rightarrow \{0, 1\}$  be the characteristic function of  $K$ . By a recursive definition of  $C_K$  we mean a rule that specifies  $C_K$  on a "basis set"  $A \subseteq \{0, 1\}^*$ , and uniquely determines  $C_K$  on the rest of  $\{0, 1\}^*$  by a recurrence formula of the form

$$C_K(x) = F(x, C_K(f_1(x)), C_K(f_2(x)), \dots, C_K(f_t(x))), \\ x \in \{0, 1\}^* - A.$$

*Example 1.* Let  $G$  be a game, as defined in Section 4, and let  $G$  be the set of positions from which the player to move can force a win. Then  $G$  is uniquely determined by

- (i) if  $x \in W$  then  $x \in G$
- (ii) if  $x \in \{0, 1\}^* - W$  then  $x \in G \Leftrightarrow F_0(x) \notin G$  or  $F_1(x) \notin G$ .

*Example 2.* Let  $(<, A, G_0, G_1)$  be a self-reducibility structure for the set  $K \subseteq \{0, 1\}^*$ . Then  $K$  is determined uniquely by its intersection with  $A$ , together with the recurrence

$$\text{for } x \notin A, x \in K \Leftrightarrow G_0(x) \in K \cup G_1(x) \in K.$$