Zeitschrift: L'Enseignement Mathématique

Herausgeber: Commission Internationale de l'Enseignement Mathématique

Band: 28 (1982)

Heft: 1-2: L'ENSEIGNEMENT MATHÉMATIQUE

Artikel: TURING MACHINES THAT TAKE ADVICE

Autor: Karp, Richard M. / Lipton, Richard J.

Kapitel: 3. SUMMARY OF MAIN RESULTS

DOI: https://doi.org/10.5169/seals-52237

Here *UGAP* is the undirected maze problem. As another example, we have Adleman's [1] result that R (the set of languages accepted in polynomial time by randomizing Turing machines) has small circuits, which can be restated as

$$R \text{ is a subset of } P/poly\,.$$

It may be interesting that both these results use the probabilistic method of Erdös to prove the existence of the required advice bits.

3. Summary of Main Results

We will discuss a variety of complexity classes. These include the basic time and space classes $DTIME\,(T\,(\mathrm{n}))$, $DSPACE\,(\mathrm{S}\,(\mathrm{n}))$ and $NSPACE\,(\mathrm{S}\,(\mathrm{n}))$ and the classes:

P = the set of languages accepted in deterministic polynomial time,

R = the set of languages accepted in polynomial time by randomizing Turing machines [1],

NP = the set of languages accepted in nondeterministic polynomial time,

$PSPACE$ = the set of languages accepted in polynomial space,

$$EXPTIME = \bigcup_{i>0} DTIME\,(2^{n^i})\ .$$

Also important is the polynomial-time hierarchy of Meyer and Stockmeyer [19]. For $i \geqslant 1$ we let $\sum_i^p$ (respectively $\prod_i^p$) denote those languages accepted in polynomial time by Turing machines that make i alternations starting from an existential (respectively universal) state. Note that $NP = \sum_1^p$ and co- $NP = \prod_1^p$. Finally, note that P, $PSPACE$ and $EXPTIME$ can be viewed as complexity classes associated with alternating Turing machines; specifically, $P = ASPACE\,(log\, n)$, $PSPACE = AP$ and $EXPTIME = APSPACE$ [3, 10].

Many of the following theorems take the form

$$L \subseteq S/F \Rightarrow L \subseteq S'$$

where L and S' are uniform complexity classes and V/F is a nonuniform complexity class. The proof usually consists of showing that

$$K \in V/F \Rightarrow K \in S'$$

where the set of strings K is complete in L with respect to an appropriate reducibility. The hypothesis tells us that K is of the form S : h where S is a language in V and a bound on $|h(|x|)|$ is known. The proof that $K \in S'$ consists of giving an appropriate uniform algorithm to recognize K. The function $h(|x|)$ is not available to this uniform algorithm, but the algorithm can exploit the fact that $h(|x|)$ is consistent; i.e. for all strings y of the same length as x, $y \in K \Leftrightarrow h(|x|) \cdot y \in S$. The algorithm must somehow filter through all the strings that might be $h(|x|)$, and come up with the right decision about x. The method of doing so depends on the structure of K. The following section treats the case where K is a "game". Section 5 considers the case where K is self-reducible. Finally, Section 6 deals with the case where K has a simple recursive definition.

The main results of this paper are summarized in Figure 1. The rest of the paper is devoted to supplying proofs and additional comments on these main results. As promised in the introduction each result demonstrates that a nonuniform hypothesis can have uniform consequences.

4. The Round-Robin Tournament Method

Insight into the nature of a complexity class can often be gained by identifying "hardest" problems in the class, i.e., problems that are complete in the class with respect to an appropriate definition of reducibility. For complexity classes defined in terms of time and space on alternating Turing machines, these complete problems often take the form of games ([3, 4]). In this section we explain and apply a proof technique called "the round-robin tournament method", which enables us to relate the nonuniform complexity of a game to its uniform complexity. The specific complexity classes we consider are *PSPACE*, *P* and *EXPTIME* (alias *AP*, *ASPACE* (*log n*) and *APSPACE*, respectively ([3, 10])).

A *game G* is specified by

(i) a set $W \subseteq \{0, 1\}^*$ and

(ii) a pair of length-preserving functions F_0 and F_1, each mapping $\{0, 1\}^* - W$ into $\{0, 1\}^*$.