Alternation Trading Proofs and Their Limitations

Sam Buss*

Department of Mathematics
University of California, San Diego
La Jolla, CA 92130-0112, USA
sbuss@math.ucsd.edu

Abstract. Alternation trading proofs are motivated by the goal of separating NP from complexity classes such as LOGSPACE or NL; they have been used to give super-linear runtime bounds for deterministic and co-nondeterministic sublinear space algorithms which solve the Satisfiability problem. For algorithms which use $n^{o(1)}$ space, alternation trading proofs can show that deterministic algorithms for Satisfiability require time greater than n^{cn} for $c < 2\cos(\pi/7)$ (as shown by Williams [21, 19]), and that co-nondeterministic algorithms require time greater than n^{cn} for $c < \sqrt[3]{4}$ (as shown by Diehl, van Melkebeek and Williams [5]). It is open whether these values of c are optimal, but Buss and Williams [2] have shown that for deterministic algorithms, $c < 2\cos(\pi/7)$ is the best that can obtained using present-day known techniques of alternation trading.
This talk will survey alternation trading proofs, and discuss the optimality of the unlikely value of $2\cos(\pi/7)$.

Keywords: Satisfiability, alternation trading, indirect diagonalization, lower bounds

1 Introduction

A central open problem in computer science is the question of whether nondeterministic polynomial time (NP) is more powerful than ostensibly weaker computational classes such as polynomial time (P) or logarithmic space (LOGSPACE). These are famously important and difficult questions, and unfortunately, in spite of over 40 years of concerted efforts to prove that NP $\neq$ P or NP $\neq$ LOGSPACE, it is generally felt that minimal progress has been made on resolving them.

Alternation trading proofs are a method aimed at separating NP from smaller complexity classes, by using "indirect" diagonalization to prove separations. A typical alternation trading proof begins with a simulation assumption, for instance the assumption that the NP-complete problem of Satisfiability (SAT) can be recognized by an algorithm which uses time n^c and space $n^{o(1)}$. Iterated

* Supported in part by NSF grant DMS-1101228.

application of the simulation assumption allows it to be amplified into an assertion which can be refuted by diagonalization. This yields a proof that the simulation assumption is false.

One of the strongest alternation trading separations known to date is that SAT cannot be recognized by a deterministic algorithm which uses time n^c and space $n^{o(1)}$ for c a constant $< 2\cos(\pi/7) \approx 1.8109$ (see Theorem 8 below). The bound of $2\cos(\pi/7)$ on the runtime exponent might seem unlikely; however, it has recently been shown that this bound on the exponent is *optimal* in the sense that present-day techniques of alternation trading proofs cannot establish any better runtime bound. This is stated as Theorem 10 below, and thus gives an upper bound on the lower bounds that can be achieved with alternation trading proofs — at least using currently known techniques. In short, we provably need better techniques — or better ways to apply known techniques — in order to get improved separation results via alternation trading proofs.

The next section outlines these results in more detail. However, many details of the definitions and proofs are omitted. These details and additional background information can be found in [19, 2]. The earlier survey [12] provides an excellent introduction to alternation trading proofs, but does not include the upper bounds on lower bounds of Theorem 10.

2 Definitions and Preliminaries

We adopt the convention that time- and space-bounded algorithms are run on Turing machines with random access tapes, as this permits robust definitions for subquadratic time and sublinear space computational classes. Specifically, Turing machines are assumed to be multitape machines that have random access (indexed) tapes. This means that the Turing machine's tapes come in pairs. Each pair consists of a sequential access tape and a random access tape. The sequential access tape is accessed as usual in the Turing machine model with a tape head that can move at most one tape cell left or right per step. The random access tape is indexed by the sequential access tape, so that the Turing machine has access to the symbol written in the tape cell whose index is written on the sequential access tape. The input string is stored on a read-only random access tape.

Random access Turing machines form a very robust model of computation; for instance, [9] shows their equivalence to more general random access computers up to logarithmic factors on runtime and space.

The *space* used by the Turing machine is the number of cells which are accessed on either kind of tape, except that the contents of the (read-only) input tape do not count towards the space used by the Turing machine. For t a time-constructible function, the complexity classes $\mathrm{DTIME}(t)$ and $\mathrm{NTIME}(t)$ contain the languages L which can be recognized by deterministic, respectively nondeterministic, algorithms which use time $O(t)$.

We will work primarily with algorithms for Satisfiability that use sublinear space of only $n^{o(1)}$ or $n^{e+o(1)}$ for some constant $e < 1$. Note these sublinear space

algorithms do not even have sufficient space to store a single truth assignment for an instance of Satisfiability.

Definition 1. *Let $c, e \geq 0$. The complexity class* $\mathrm{DTISP}(n^c, n^e)$ *is the set of decision problems L such that L can be recognized by a deterministic algorithm which uses time $n^{c+o(1)}$ and space $n^{e+o(1)}$. The complexity class* $\mathrm{NTISP}(n^c, n^e)$ *is defined similarly but allowing nondeterministic algorithms instead of deterministic algorithms.*

$\mathrm{DTS}(n^c)$ *is equal to* $\mathrm{DTISP}(n^c, n^0)$. *And* $\mathrm{NTS}(n^c)$ *is* $\mathrm{NTISP}(n^c, n^0)$.

It is a little unusual for the definitions of DTISP and NTISP to include the "$o(1)$" terms in the exponents, but the advantage is that it gives extra $n^{o(1)}$ factors which can absorb polylogarithmic factors in time or space bounds.

The Cook-Levin theorem states that SAT is NP-complete. In fact, SAT is NP-complete in a very strong way. An algorithm is called "quasilinear time" provided it has runtime $n(\log n)^{O(1)}$, and "polylogarithmic time" provided it has runtime $(\log n)^{O(1)}$.

Theorem 2. *Let $L \in \mathrm{NTIME}(n)$. Then there is a quasilinear time many-one reduction f from L to* SAT *such that there is a polylogarithmic time algorithm, which given x and j, produces the j-th symbol of $f(x)$.*

The point of Theorem 2 is that the computational complexity of SAT is as strong as any language in $\mathrm{NTIME}(n)$. In particular:

Corollary 3. *Fix $c \geq 0$.* $\mathrm{NTIME}(t) \subseteq \mathrm{DTS}(n^c)$ *if and only if* $\mathrm{SAT} \in \mathrm{DTS}(n^c)$.

Proofs of Theorem 2 and its precursors were given by [14, 17, 15, 3, 16, 18, 7, 12]. For the most direct proof of Theorem 2 as stated see [12], which uses much the same methods as [17, 16].

Corollary 3 provides the justification for "slowdown" steps in alternation trading proofs. Alternation trading proofs also contain "speedup" steps which allow sublinear space computations to be speeded up, at the cost of introducing alternations. Speedup steps are based on the following theorem which states that runtime can be speeded up by alternation. The theorem is based on techniques independently developed by Bennett [1], Nepomnjaščiĭ [13], and Kannan [10]. We state it only for the special case where the space is $n^{o(1)}$, but it can be generalized to space n^e for constants $e < 1$.

Theorem 4. *Suppose $a > b > 0$ and that $L \in \mathrm{DTS}(n^a)$. Then membership in L can be expressed as*

$$x \in L \;\Leftrightarrow\; (\exists y,\, |y| \leq |x|^{b+o(1)})(\forall z,\, |z| \leq d \log |x|)(\langle x, g(y,z)\rangle \in L')$$

for some constant $d > 0$, some $L' \in \mathrm{DTS}(n^{a-b})$, and some function $g \in \mathrm{DTS}(n^0)$ such that $|(g(y,z)| = |x|^{o(1)}$.

3 Separation Results with Alternation Trading

The first separation results using alternation trading were established by Kannan [10] and Fortnow [6], who were motivated by problems such as proving that NP is not equal to NL. Theorem 5 states a simplified version of Fortnow's results.

Theorem 5. *Let* $\epsilon > 0$. *Then* SAT $\notin$ DTISP$(n^1, n^{1-\epsilon})$. *In fact, we have* $\overline{\text{SAT}} \notin$ NTISP$(n^1, n^{1-\epsilon})$. *Consequently,* NTIME$(n) \not\subseteq$ coNTISP$(n^1, n^{1-\epsilon})$.

Fortnow's theorem was quickly extended to better runtime lower bounds. Lipton and Viglas [11] improved the n^1 time bound to n^c for all $c < \sqrt{2}$, but with polylogarithmic space instead of $n^{1-\epsilon}$. Their methods give the following theorem:

Theorem 6. *Let* $c < \sqrt{2} \approx 1.414$. *Then* SAT $\notin$ DTS(n^c).

This bound was improved by Fortnow and van Melkebeek [8, 7] to use $c < \phi$ where $\phi = (1 + \sqrt{5})/2 \approx 1.618$ is the golden ratio.

Theorem 7. *Let* $c < \phi$. *Then* SAT $\notin$ DTS(n^c).

The bound $c < \phi$ was improved to $c < \sqrt{3} \approx 1.732$ by Williams [20] and to $c < 1.759$ by Diehl and van Melkebeek [4] (the latter result was a more general result about randomized computation). Finally, these bounds were improved by Williams [21, 19] to $c < 2\cos(\pi/7) \approx 1.8109$. His theorem applied to a more general setting of modular counting, but for SAT and NTIME(n) his results were:

Theorem 8. *Let* $c < 2\cos(\pi/7)$. *Then* SAT $\notin$ DTS(n^c).

Corollary 9. *Let* $c < 2\cos(\pi/7)$. *Then* NTIME$(n) \not\subseteq$ DTS(n^c).

Subsequently to proving Theorem 8, Williams used a computer-based search (coded in Maple) to search for better alternation trading proofs. For this, Williams formulated a precise set of inference rules that allow the derivation of assertions about inclusions between complexity classes. We do not describe the inference rules here, but they can be found in [19, 2]. The essential idea is that the inference rules formalize the "slowdown" and "speedup" principles of Corollary 3 and Theorem 4. This computerized search did not lead to any improved alternation trading proofs beyond those already found for Theorem 8.

The somewhat mysterious value $2\cos(\pi/7)$ arises from its being one of the roots of $x^3 - x^2 - 2x + 1 = 0$.

4 Limits on Alternation Trading Proofs

It had long been informally conjectured that alternation trading proofs should be able to establish Theorems 6-8 for all values of $c < 2$. However, as a result of the computerized search, Williams conjectured that the (admittedly unlikely sounding) value $2\cos(\pi/7)$ is the best that can be achieved with his formalized inference rules. This conjecture was recently proved by Buss and Williams [2]:

e	c
0.001	1.80083
0.01	1.79092
0.1	1.69618
0.25	1.55242
0.5	1.34070
0.75	1.15765
0.9	1.06011
0.99	1.00583
0.999	1.00058

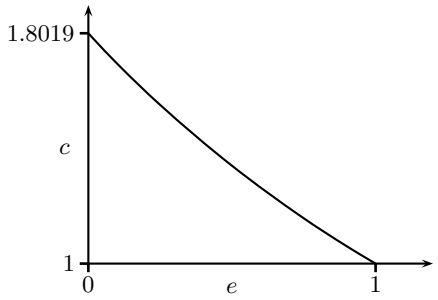

Fig. 1. Showing the maximum value of c, as a function of e, for which alternation trading proofs suffice to show that SAT is not in $\mathrm{DTISP}(n^c, n^e)$. The values are accurate to within 10^{-5}. This figure is from [2].

Theorem 10. *The alternation trading proof inference system, as described in [19, 2], can prove that* $\mathrm{SAT} \notin \mathrm{DTS}(n^c)$ *if and only if* $c < 2\cos(\pi/7)$.

This inference system for alternation trading proofs includes all alternation trading proofs which have been developed so far, and seems to fully capture the power of the Bennett-Nepomnjaščiĭ-Kannan technique of Theorem 4. Thus, Theorem 10 appears to put a meaningful bound on what can be achieved by alternation trading proofs.

Fortnow and van Melkebeek [8] and Williams [19] also used alternation trading proofs to prove results about $\mathrm{NTIME}(n) \not\subseteq \mathrm{DTISP}(n^c, n^e)$ for values of $c > 1$ and $e < 1$. Already [8] showed that, for any value of $e < 1$, this holds for c sufficiently close to 1; and improved values were given by [19]. The possible values for c and e were further improved, and shown to be optimal by Buss and Williams [2]:

Theorem 11. *The alternation trading proof inference systems described in [19, 2] can prove* $\mathrm{SAT} \notin \mathrm{DTISP}(n^c, n^e)$ *for precisely the values of* c *and* e *graphed in Fig. 1.*

Unfortunately, the values shown in Fig. 1 are numerically computed; there is no known formula for describing the values of c and e for which alternation trading proofs exist.

5 Other Directions

So far, we have discussed the question of whether SAT lies in $\mathrm{DTS}(n^c)$ or $\mathrm{DTISP}(n^c, n^e)$ for constant values of c and e. The alert reader will have noticed that Theorem 5 also discussed whether $\overline{\mathrm{SAT}}$ lies in the nondeterministic class $\mathrm{NTISP}(n^1, n^{1-\epsilon})$. A number of further such results have been obtained, in particular by [8, 7, 21, 19], culminating in the following theorem proved by Diehl, van Melkebeek, and Williams [5]:

Theorem 12. *Let* $c < \sqrt[3]{4}$. *Then* $\overline{\text{SAT}} \notin \text{NTS}(n^c)$. *Consequently,* $\text{NTIME}(n) \not\subseteq \text{coNTS}(n^c)$.

It is tempting to conjecture that the methods of [2] can be extended to prove that the constant $\sqrt[3]{4}$ is optimal for what can be proved with alternation trading proofs. However, to the best of our knowledge, this has not been attempted yet and so it remains an open problem.

References

1. Bennett, J.: On Spectra. Ph.D. thesis, Princeton University (1962)
2. Buss, S., Williams, R.: Limits on alternation-trading proofs for time-space lower bounds, manuscript, submitted for publication. Shorter version appeared in IEEE Conf. on Computational Complexity (CCC), pp. 181-191, 2012
3. Cook, S.A.: Short propositional formulas represent nondeterministic computations. Information Processing Letters 26, 269–270 (1988)
4. Diehl, S., van Melkebeek, D.: Time-space lower bounds for the polynomial-time hierarchy on randomized machines. SIAM Journal on Computing 36, 563–594 (2006)
5. Diehl, S., van Melkebeek, D., Williams, R.: An improved time-space lower bound for tautologies. Journal of Combinatorial Optimization 22(3), 325–338 (2011), an earlier version appeared in COCOON'09, pp. 429-439
6. Fortnow, L.: Nondeterministic polynomial time versus nondeterministic logarithmic space: Time-space tradeoffs for satisfiability. In: Proc. IEEE Conference on Computational Complexity (CCC). pp. 52–60 (1997)
7. Fortnow, L., Lipton, R., van Melkebeek, D., Viglas, A.: Time-space lower bounds for satisfiability. Journal of the ACM 52(6), 835–865 (2005)
8. Fortnow, L., van Melkebeek, D.: Time-space tradeoffs for nondeterministic computation. In: Proc. IEEE Conference on Computational Complexity (CCC). pp. 2–13 (2000)
9. Gurevich, Y., Shelah, S.: Nearly linear time. In: Logic at Botik'89. pp. 108–118. Lecture Notes in Computer Science #363, Springer Verlag (1989)
10. Kannan, R.: Towards separating nondeterminism from determinism. Mathematical Systems Theory 17, 29–45 (1984)
11. Lipton, R., Viglas, A.: On the complexity of SAT. In: Proc. 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS). pp. 459–464 (1999)
12. van Melkebeek, D.: Time-space lower bounds for NP-complete problems. In: Current Trends in Theoretical Computer Science, pp. 265–291. World Scientific (2004)
13. Nepomnjaščiĭ, V.A.: Rudimentary predicates and Turing computations. Dokl. Akad. Nauk SSSR 195, 282–284 (1970), English translation in *Soviet Math. Dokl.* 11 (1970) 1462–1465
14. Pippenger, N., Fisher, M.J.: Relations among complexity measures. Journal of the ACM 26, 361–381 (1979)
15. Robson, J.M.: A new proof of the NP completeness of satisfiability. In: Proc. 2nd Australian Computer Science Conference. pp. 62–69 (1979)
16. Robson, J.M.: An $O(T \log T)$ reduction from RAM computations to satisfiability. Theoretical Computer Science 81, 141–149 (1991)
17. Schnorr, C.P.: Satisfiability is quasilinear complete in NQL. Journal of the ACM 25, 136–145 (1978)

18. Tourlakis, I.: Time-space tradeoffs for SAT and related problems. Journal of Computer and System Sciences 63(2), 268–287 (2001)
19. Williams, R.: Alternation-trading proofs, linear programming, and lower bounds, to appear. A shorter extended abstract appeared in *Proc. 27th Intl. Symp. on Theory of Computings (STACS 2010)*, DOI: 10.4230/LIPIcs.STACS.2010.2494, available from http://stacs-conf.org
20. Williams, R.: Algorithms and Resource Requirements for Fundamental Problems. Ph.D. thesis, Carnegie Mellon University (August 2007)
21. Williams, R.: Time-space tradeoffs for counting NP solutions modulo integers. Computational Complexity 17(2), 179–219 (2008)