Speedup for Natural Problems and $NP = ?coNP$

Hunter Monroe*

June 19, 2009

1 Introduction

Informally, a language L has speedup if, for any Turing machine (TM) for L, there exists one that is better. Blum [2] showed that there are computable languages that have almost-everywhere speedup. These languages were unnatural in that they were constructed for the sole purpose of having such speedup. We identify a condition apparently only slightly stronger than $P \neq NP$ which implies that accepting any $coNP$-complete language has an infinitely-often (i.o.) superpolynomial speedup and $NP \neq coNP$. We also exhibit a natural problem which unconditionally has a weaker type of i.o. speedup based upon whether the full input is read.[1] Neither speedup pertains to the worst case.

* This paper is in honor of the retirement of Benjamin Klein from Davidson College. The views expressed in this column are those of the author and should not be attributed to the International Monetary Fund, its Executive Board, or its management. This paper could not have been prepared without encouragement from Marius Zimand and Bill Gasarch. I would also like to thank participants in a seminar at the University of Maryland Complexity Seminar who provided useful comments. Remaining errors are my own. Email: hkmbh@huntermonroe.com.

[1] For a review of related literature, see Monroe [9].

2 Conditional Speedup for $coNP$-Complete Languages

Def 2.1 Define $BHP = \{\langle N, x, 1^t\rangle |$ there is at least one accepting path of nondeterministic TM N on input x with t or fewer steps$\}$, $DBHP$ is the same but with N deterministic, and $HP = \{\langle N, x\rangle |$ there is at least one accepting path of NTM N on input x (with no bound on the number of steps)$\}$. If M is a deterministic TM then T_M is the function that maps a string x to how many steps $M(x)$ takes.

Note that BHP is NP-complete with the accepting path as a certificate, that $coBHP$ is $coNP$-complete, and $DBHP \in P$.

Suppose $P \neq NP$ and therefore $coBHP \notin P$. The following condition rules out the absurd possibility that some M can nevertheless accept the subset of inputs beginning with any particular machine-input pair within a polynomial bound (for that subset):

> (*) Let M be a deterministic TM accepting $coBHP$. Then there exists $\langle N', x'\rangle \in coHP$ such that the function $f(t) = T_M(N', x', 1^t)$ is not bounded by any polynomial.[2]

An intuition for why this condition might hold could be a belief that there is at least one N', x' for which M must infinitely often use brute force to rule out all possible accepting paths of N' on x' with at most t steps.

Def 2.2 For M and M' accepting a language L, write $M \leq_p M'$ if there exists a polynomial p such that for all inputs $x \in L$:

$$T_M(x) \leq p(|x|, T_{M'}(x)). \quad (1)$$

If L has a least element M under $\leq_p$, say that M is *p-optimal*[3] and otherwise say that L has *i.o. superpolynomial speedup.*

Theorem 2.3 *If L is NP-complete, L does not have superpolynomial speedup.*

[2] The function f may depend on M, N', and x'. For inputs not in $coBHP$, M does not accept, but otherwise its behavior is not constrained.

[3] See Krajíček and Pudlák [6].

Proof: For any $L \in NP$, there is a p-optimal TM for finding witnesses for L, by Levin [7].[4] Levin's universal witness search algorithm works for any NP language by dovetailing every possible TM, running any output produced through a predetermined witness verifier, and then printing out the first witness that is verified. If L is NP-complete, then there is a p-optimal algorithm accepting L using the self-reducibility of NP-complete languages, by Schnorr [11]. ∎

Theorem 2.4 *If (*) holds, then $coBHP$ has superpolynomial speedup, and $NP \neq coNP$.*

Proof: Given M accepting $coBHP$, choose N', x' for M in (*), so $f(t) = T_M(\langle N', x', 1^t \rangle)$ is not polynomially bounded. We create M' as follows:

1. Input $\langle N, x, 1^t \rangle$.
2. If $N, x \neq N', x'$ then run $M(N, x, 1^t)$.
3. If $N, x = N', x'$ then reject immediately.

Then $M' <_p M$, and $coBHP$ therefore has superpolynomial speedup. Since $coBHP$ is $coNP$-complete, and no NP-complete language has superpolynomial speedup, then $NP \neq coNP$. ∎

Theorem 2.4 is a striking result: a condition only slightly stronger than $P \neq NP$, which states that at least one instance of $coBHP$ is hard, implies $NP \neq coNP$.[5]

Theorem 2.5 *If one $coNP$-complete language has superpolynomial speedup, then all of them do.*

Proof: For $coNP$-complete languages L_1 and L_2, suppose L_1 has superpolynomial speedup and L_2 does not. Let f, g be polynomial time reductions from L_1 to L_2 and vice versa, i.e., $x \in L_1$ if and only if $f(x) \in L_2$, and $x \in L_2$ if and only if $g(x) \in L_1$. Suppose M_2 is p-optimal for L_2. Then $M_2' = M_2 \circ f \circ g(x)$ is also p-optimal for L_2. Let $M_1 = M_2 \circ f$. Because L_1

[4] See Gurevich [5], Goldreich [4], Ben-Amram [1], Messner [8], and Sadowski [10].

[5] Hartmanis asked whether is there an optimal search algorithm similar to Levin's that also rejects when there is no witness (Trakhtenbrot [12]); in this case, there is not for NP-complete languages.

has superpolynomial speedup by assumption, there exists $M_1' <_p M_1$. That implies $M_1' \circ g <_p M_2'$ on inputs $x \in L_2$ so in fact M_2 was not p-optimal, a contradiction. ∎

3 Unconditional Speedup for $coBHP$

This section proves unconditionally that $coBHP$ has a different form of speedup which hinges upon whether the full input is read.[6] The intuition is that it is useful for M accepting $coBHP$ to be able to recognize that its input begins with a non-halting N', x', but no M can recognize all non-halting N', x', since $coHP$ is not computably enumerable (c.e.).[7]

Def 3.1 For M and M' accepting a language L, write $M' <_b M$ if (1) there exists an infinite subset of inputs $S \subset L$ on which the runtime of M is not bounded above by a constant but the runtime of M' is bounded above by a constant, and (2) there exists a constant c_S such that the runtime disadvantage of M' on inputs in $L - S$ is less than an additive factor c_S. If for any M there exists M' such that $M' <_b M$, say that L has *i.o. b-speedup*. The speedup is *effective* if M' is computable from M.[8] Otherwise, say that M is *b-optimal*.

Lemma 3.2 *For any M accepting $coBHP$, there is some $N', x' \in coHP$ computable from M for which $\mathrm{T}_M(N', x', 1^t) \geq t$.*

Proof: Assume, by way of contradiction, that for some M and for all $N', x' \in coHP$ there exists a t_0 such that $\mathrm{T}_M(N', x', 1^{t_0}) < t_0$. This computation must have determined that $\langle N', x', 1^{t_0} \rangle \in coBHP$ without reading the entire input. In particular, it only read part of the 1^{t_0}. Hence for all $t > t_0$, $\mathrm{T}_M(N', x', 1^t) < t_0$. Therefore

$$\langle N, x \rangle \in coHP \implies (\exists t_0)[M(N, x, 1^{t_0}) \text{ accepts and } T_M(N, x, 1^{t_0}) < t_0].$$

[6]This consideration is excluded in inequality (1) by the $|x|$ term.

[7]The proof below can be seen as a bounded version of the statement that every non-c.e. language has speedup if M' is "better" than M at accepting a language L if M' correctly accepts a strictly larger subset of L than M. If L is productive, then the speedup is effective.

[8]The trivial linear speedup is not b-speedup. Geffert [3] describes nontrivial linear speedups for nondeterministic machines.

Therefore $coHP$ is c.e., a contradiction. Because $coHP$ is productive, N', x' for which no such t_0 exists is computable from M. ∎

Theorem 3.3 *$coBHP$ and $coDBHP$ each have b-speedup, and the speedup is effective.*[9]

Proof: Suppose M accepts $coBHP$. Compute $N', x' \in coHP$ for M by Lemma 3.2. We create M' as follows:

1. Input $\langle N', x', 1^t \rangle$ but without yet reading any of 1^t.
2. If $N, x \neq N', x'$ then run $M(N, x, 1^t)$.
3. If $N, x = N', x'$ then reject immediately.

Note that there is a constant C such that, for all t, $T_M(N', x', 1^t) \geq t$ and $T_{M'}(N', x', 1^t) \leq C$. Hence, $coBHP$ has b-speedup, with $S = \{\langle N', x', 1^t \rangle\}$. The same proof applies to $coDBHP$. ∎

4 Conclusion

We conjecture that any M which might serve as a counterexample to widely believed complexity hypotheses could, as in Lemma 3.2, be modified to perform tasks known to be noncomputable. In particular:

Conjecture 4.1 *If there exists $M \in P$ accepting a $coNP$-complete language (for instance $coBHP$), then M can be modified to accept a language that is not c.e. (for instance $coHP$).*

Similarly, some suspect that integer multiplication has speedup, and it is generally believed that integer multiplication is a one-way function. These conjectured properties could be related to a known property of integer multiplication that apparently has never been used to prove anything about the complexity of multiplication itself: the Presburger arithmetic without multiplication is a decidable while arithmetic with multiplication is undecidable.

Conjecture 4.2 *Suppose M can factor integers in polynomial time. Then M can be modified to accept true arithmetic statements.*

[9] There are $coNP$-complete languages which do not have b-speedup. For instance, a b-optimal M for $TAUT$ reads clause $i+1$ only if the first i clauses are a tautology.

References

[1] Amir Ben-Amram, *The existence of optimal programs*, Computability and Complexity from a Programming Perspective (Neil D. Jones, ed.), MIT Press, Cambridge, MA, 1997.

[2] Manuel Blum, *A machine-independent theory of the complexity of recursive functions*, J. ACM **14** (1967), 322–36.

[3] Viliam Geffert, *A speed-up theorem without tape compression*, Theor. Comput. Sci. **118** (1993), no. 1, 49–79.

[4] Oded Goldreich, *Foundations of cryptography*, vol. Basic Tools, Cambridge University Press, New York, NY, 2001.

[5] Yuri Gurevich, *Kolmogorov machines and related issues*, Bulletin of the European Association for Theoretical Computer Science **35** (1988), 71–82.

[6] Jan Krajíček and Pavel Pudlák, *Propositional proof systems, the consistency of first order theories and the complexity of computations*, J. Symb. Log. **54** (1989), 1063–79.

[7] Leonid A. Levin, *Universal sequential search problems*, Problems of Information Transmission **9** (1973), 265–66.

[8] Jochen Messner, *On optimal algorithms and optimal proof systems*, Lecture Notes in Computer Science **1563** (1999), 541–50.

[9] Hunter Monroe, *Are there natural problems with speedup?*, Bulletin of the European Association for Theoretical Computer Science **94** (2008), 212–20.

[10] Zenon Sadowski, *On an optimal deterministic algorithm for SAT*, CSL (Georg Gottlob, Etienne Grandjean, and Katrin Seyr, eds.), Lecture Notes in Computer Science, vol. 1584, Springer, 1998, pp. 179–187.

[11] Claus-Peter Schnorr, *Optimal algorithms for self-reducible problems*, ICALP, 1976, pp. 322–37.

[12] Boris A. Trakhtenbrot, *A survey of Russian approaches to perebor (brute-force search) algorithms*, Annals of the History of Computing **6** (1984), 384–400.