Notions of Average-Case Complexity for Random 3-SAT

Albert Atserias*

Universitat Politècnica de Catalunya, Barcelona, Spain

Abstract. By viewing random 3-SAT as a distributional problem, we go over some of the notions of average-case complexity that were considered in the literature. We note that for dense formulas the problem is polynomial-time on average in the sense of Levin. For sparse formulas the question remains widely open despite several recent attempts.

1 Introduction

The satisfiability problem for propositional logic is central to computational complexity. The work of Cook [2] showed that the problem is NP-complete, even when restricted to 3-CNF formulas, and is thus hard in the worst-case unless P = NP. Later on, the optimization versions of 3-SAT were also considered and showed hard. Namely, Hastad [4] culminated the monumental work of the 1990s on PCPs by showing that the number of clauses that can be satisfied simultaneously in a 3-CNF formula cannot be approximated within a ratio better than $7/8$ in polynomial-time, unless P = NP. The current decade is perhaps time for studying the *average-case complexity* of 3-SAT. Is it hard on average as well, unless P = NP, or is it easier?

The program comes also motivated from the fact that a fairly natural probability distribution on 3-CNF formulas has attracted the attention of many different communities, from AI to statistical physics, through combinatorics and mathematical logic. Our aim here is to review the background for a complexity-theoretic approach to the average-case complexity of random 3-SAT. In this short note we focus on the different definitions of average-case complexity that were introduced in the literature and their relationship. In the talk we will overview some of the partial results towards settling the main open questions.

2 Notions of Average Case Complexity

For every $n \geq 0$, let $f_n : \{0,1\}^n \to \{0,1\}$ be a Boolean function. In order to simplify notation, we will use f instead of f_n, and we will write $f = f_n$ to emphasize the fact that f is actually a sequence of functions parameterized by n. It will be understood from context that f denotes the sequence of functions $\{f_n\}$ in some cases, and the particular function f_n in others. We adopt the framework of *ensemble of distributions* suggested by Impagliazzo [5], where a different probability distribution is considered

* Supported in part by CICYT TIC2001-1577-C03-02 and the Future and Emerging Technologies programme of the EU under contract number IST-99-14186 (ALCOM-FT).

for each input size. So let $\mu = \mu_n$ be a probability distribution on $\{0,1\}^n$. We should think of μ as a sequence of distributions, one for each $n \geq 0$. The pair (f, μ) is called a *distributional problem*. Informally, the problem reads as follows: given a random input $x \in \{0,1\}^n$ drawn according to distribution μ_n, compute $f_n(x)$.

Levin's Average Case. Let (f, μ) be a distributional problem. Consider an algorithm A computing f, and let $T = T_n : \{0,1\}^n \to \mathbb{N}$ be its running time on inputs of length n. What should it mean that the running time T of A be polynomial on average with respect to μ? The obvious immediate candidate definition would be this: there exists a $k \geq 1$ such that $\mathrm{E}_\mu[T] = O(n^k)$ where E_μ denotes expectation with respect to μ. Unfortunately, this simple definition suffers from a serious problem: the class of functions that are polynomial on average under this definition is not closed under polynomial images. Indeed, if we let μ be the uniform distribution on $\{0,1\}^n$, and let T be such that $T(x) = n$ for all but one string x_0 in $\{0,1\}^n$ for which $T(x_0) = 2^n$, then $\mathrm{E}_\mu[T] = O(n)$ while $\mathrm{E}_\mu[T^2] = \Omega(2^n)$. This lack of robustness would spoil any attempt to build a theory of polynomial reducibilities among distributional problems. A satisfactory remedy to this was discovered by Levin and reformulated by Impagliazzo for ensembles of distributions: we say that T *is polynomial on average with respect to* μ if there exists a $k \geq 1$ such that $\mathrm{E}_\mu[T^{1/k}] = O(n)$. It is now immediate from this definition that the class of functions that is polynomial on average is closed under polynomial functions. We say that a distributional problem (f, μ) has a *polynomial-time on average algorithm* if there exists an algorithm A for f whose running time is polynomial on average with respect to μ.

Impagliazzo's Benign Algorithms. Let $f = f_n : \{0,1\}^n \to \{0,1\}$ be a Boolean function. A *prudent algorithm* for f is one that, for every input $x \in \{0,1\}^n$, outputs either $f(x)$ or ?. We should think of ? as a "don't know answer". Clearly, a prudent algorithm is useful only if it rarely outputs ?. We say that a distributional problem (f, μ) has a *polynomial-time benign algorithm* if there exists a prudent algorithm $A(x, \delta)$ for f that is polynomial-time in $|x|$ and $1/\delta$, and such that $\Pr_\mu[A(x, \delta) = ?] \leq \delta$ where $\Pr_\mu$ denotes probability with respect to μ. The last clause of this definition formalizes the idea that the algorithm "rarely outputs ?".

Impagliazzo [5] showed that the two notions introduced so far coincide, from which we conclude that the concept is fairly robust. We reproduce the proof since it is informative.

Theorem 1 (Impagliazzo). *Let (f, μ) be a distributional problem. Then, the following are equivalent:*

1. *(f, μ) has a polynomial-time on average algorithm.*
2. *(f, μ) has a polynomial-time benign algorithm.*

Proof: We start by 1. implies 2.: Let $\mathrm{E}_\mu[T^{1/k}] \leq cn$. By Markov's inequality, we have $\Pr_\mu[T(x) > (tcn)^k] \leq 1/t$. Thus, for building a benign algorithm, it suffices to run the polynomial-time on average algorithm for $(cn/\delta)^k$ steps, and if it does not terminate, output ?. Next we show that 2. implies 1.: Suppose the benign algorithm runs in $(n/\delta)^k$ steps. Run the benign algorithm with parameter $\delta = 1/2,\ 1/4,\ 1/8,\ \ldots$ until an output

different from ? is returned. Then, the expectation of the $2k$-th root of the running time of this algorithm is bounded by $(2n)^{1/2} + 1/2(4n)^{1/2} + 1/4(8n)^{1/2} + \cdots = O(n^{1/2})$ since at most $1/2$ of the inputs return ? in the first round, at most $1/4$ in the second round, and so on. □

Certification Algorithms. Let (f, μ) be a distributional problem. A *sound algorithm* for f is one that, for every input $x \in \{0,1\}^n$, if it outputs 1, then indeed $f(x) = 1$. Clearly, a sound algorithm A is useful only if it outputs 1 on a large fraction of the "yes" instances, in other words, only if $|\Pr_\mu[f(x) = 1] - \Pr_\mu[A(x) = 1]|$ is small. In such cases we say that it is *almost complete*. We say that a distributional problem (f, μ) has a *polynomial-time certification algorithm* if there exists a sound algorithm $A(x, \delta)$ for f that is polynomial in $|x|$ and $1/\delta$, and such that $|\Pr_\mu[f(x) = 1] - \Pr_\mu[A(x) = 1]| \leq \delta$. The last clause of this definition formalizes the idea of almost completeness. The relationship is now very easy to see:

Lemma 1. *Let (f, μ) be a distributional problem. Then, if (f, μ) has a polynomial-time benign algorithm, then (f, μ) has a polynomial-time certification algorithm.*

Proof: Let $A(x, \delta)$ be the benign algorithm. Consider the following algorithm $B(x, \delta)$: run the benign algorithm $A(x, \delta)$, and if it outputs ?, output 0. Clearly, $B(x, \delta)$ is sound. Moreover, by soundness, we have $|\Pr_\mu[f(x) = 1] - \Pr_\mu[B(x, \delta) = 1]| = \Pr_\mu[f(x) \neq B(x, \delta)]$ which in turn is bounded by $\Pr_\mu[A(x, \delta) = ?] \leq \delta$. □

If we put Theorem 1 and Lemma 1 together we see that if (f, μ) has a polynomial-time on average algorithm, then (f, μ) has a polynomial-time certification algorithm. In the contrapositive form, if (f, μ) is hard to certify, then (f, μ) is hard on average. Although we do not know whether the converse relationship holds in general, we note in the next section that it holds for the particular case of random 3-SAT.

3 Random 3-SAT

Let $x_1, \ldots, x_n$ be n propositional variables. A literal is a variable or its negation. A 3-clause is a tuple of three literals. A 3-CNF formula is a set of 3-clauses. Note that the number of 3-clauses is exactly $(2n)^3$. Thus, a 3-CNF formula can be encoded by a binary string of length $(2n)^3$ denoting which clauses are present and which are not.

There are several probability distributions that have been considered in the literature. The one we adopt here is inspired from the theory of random graphs. The distribution $\mu = \mu_n$ is parameterized by a real number $p = p_n$ in $(0, 1)$ and consists in choosing each clause with independent probability p. This probability model is sometimes referred to as the model A. The model B considers the number of clauses m as fixed and chooses the formula uniformly within that set. Both these models have several variants according to whether clauses are ordered tuples or sets, and may, or may not, have repeated and complementary literals. As in the random graph model, which model to use is often a matter of convenience, and rarely an important issue as far as the results are concerned.

We are interested in the distributional problem $(UNSAT, \mu)$, where $UNSAT = UNSAT_n$ is simply the unsatisfiability problem on 3-CNF formulas with n variables, and $\mu = \mu_n$ is the probability distribution that we just described. Notice that here n is not exactly the length of the input, but it is polynomially related. Notice also that μ is parameterized by $p = p_n$, and the complexity of the distributional problem may very well depend on p. As a matter of fact, when p is large, it can be seen that $(UNSAT, \mu)$ has a benign polynomial-time algorithm. Before proving that, we first show that for all values of p that guarantee unsatisfiability of a random formula with overwhelming probability, the three notions of average-case complexity considered so far coincide.

Theorem 2. *Let $p \geq (\ln 2 + \epsilon)/n^2$, with $\epsilon > 0$. Then, the following are equivalent:*

1. *$(UNSAT, \mu)$ has a polynomial-time on average algorithm.*
2. *$(UNSAT, \mu)$ has a polynomial-time benign algorithm.*
3. *$(UNSAT, \mu)$ has a polynomial-time certification algorithm.*

Proof: By Theorem 1 and Lemma 1, it suffices to show that 3. implies 1. Let $A(x, \delta)$ be the certification algorithm. Assume its running time is $(n/\delta)^k$. Consider the following algorithm. Run the certification algorithm with parameter $\delta = 1/2,\ 1/4,\ 1/8, \ldots$ until either "unsatisfiable" is returned, in which case we return "unsatisfiable" as well, or the parameter becomes smaller than 2^{-n}, in which case we run through all 2^n truth assignments to check whether F is satisfiable or not. By soundness of the certification algorithm, it is clear that this algorithm is correct. Let us estimate the expectation of the r-th root of its running time for a constant r to be determined later.

When $p \geq (\ln 2 + \epsilon)/n^2$, the probability that a random formula is satisfiable is $2^{-\gamma n}$ for some constant $\gamma > 0$, as it is easy to see. Let us consider the set of "satisfiable" instances. For those instances, the running time of the algorithm can only be bounded by

$$\sum_{i=1}^{n} (2^i n)^k + 2^{cn}$$

for some constant $c \geq 1$, which is time 2^{dkn} for some other constant $d \geq c$. Hence, the "satisfiable" instances contribute to the expectation of the r-th root of the running time by at most $2^{-\gamma n} 2^{dkn/r}$. Let us now turn to the contribution to the expectation of the "unsatisfiable" instances. The expectation of the r-th root of the running time for those instances is bounded by

$$(2n)^{k/r} + 2^{-1}(4n)^{k/r} + 2^{-2}(8n)^{k/r} + \cdots + 2^{-n+1}(2^n n)^{k/r} + 2^{-n}(2^{cn})^{1/r}$$

since at most $1/2$ of the "unsatisfiable" instances miss the first round, at most $1/4$ of those miss the second round, and so on, until at most $1/2^n$ of the instances miss all rounds in which case the algorithm goes on to cycle through all 2^n truth assignments in time 2^{cn}. It is now straightforward to see that if we take r large enough, say $r > dk/\gamma$, then the total expectation of the r-root of the running time is $O(n)$. □

In general, the proof technique of this result applies to any distributional problem for which the "no" instances represent a fraction that is inversely polynomial with respect

to the worst-case running time that it is required to solve the problem. Let us conclude this paper with the promised benign algorithm when p is large. The reader will notice that the proof below resembles the arguments in [1].

Theorem 3. *Let $p = \omega(1/n)$. Then $(UNSAT, \mu)$ has a polynomial-time benign algorithm.*

Proof sketch: Consider the following algorithm. Let F be the input 3-CNF formula and let δ be the error parameter. Let $\gamma > 0$ be a small constant to be determined later. If $\delta < 2^{-\gamma n}$, simply run through all 2^n truth assignments to check whether F is satisfiable or not. If $\delta \geq 2^{-\gamma n}$, find the most popular variable x in F. Consider the set of 2-clauses in $F|_{x=0}$ and $F|_{x=1}$, and run a polynomial-time 2-SAT algorithm on the resulting 2-CNF formulas. If both are unsatisfiable, report "unsatisfiable". Otherwise, output ?.

It should be clear from its definition that the algorithm is prudent. It is also clear that the running time of the algorithm is polynomial in n and $1/\delta$. Indeed, when $\delta < 2^{-\gamma n}$, the running time is $2^{O(n)}$ which is polynomial in $1/\delta$, and in the other case the running time is polynomial in n. Let us argue that the probability that it outputs ? is smaller than δ. When $\delta < 2^{-\gamma n}$, the algorithm never outputs ?. So let us assume that $\delta \geq 2^{-\gamma n}$. Each variable appears in $\Theta(n^2)$ clauses. Hence, the expected number of occurrences of each variable is $\Theta(n^2 p)$, which is $\omega(n)$ since $p = \omega(1/n)$. It follows from concentration bounds that the probability that a particular variable appears less than half this number of times is $e^{-\omega(n)}$. Thus, by Markov's inequality, the probability that some variable appears $\omega(n)$ times is at least $1 - ne^{-\omega(n)}$. The number of 2-clauses in $F|_{x=0}$ and $F|_{x=1}$ is thus $\omega(n)$ with at least that much probability. Moreover, the resulting 2-CNF formulas are random, so the probability that one of them is satisfiable is bounded by $2^{-\Omega(n)}$, as is well-known. All in all, the probability that the algorithm does not report "unsatisfiable" is bounded by $2^{-\Omega(n)}$. Thus, the probability that the algorithm outputs ? is bounded by δ since $\delta \geq 2^{-\gamma n}$. Here γ is chosen to be the hidden constant in the $2^{-\Omega(n)}$ bound. □

It follows from this result and Theorem 1 that when $p = \omega(1/n)$, the distributional problem $(UNSAT, \mu)$ is solvable in polynomial-time on average in the sense of Levin. For $p = O(1/n^{3/2-\epsilon})$, recent work has focused on certification algorithms [3]. For $p = O(1/n^2)$, the problem is widely open.

References

1. P. Beame, R. Karp, T. Pitassi, and M. Saks. The efficiency of resolution and Davis-Putnam procedures. *SIAM Journal of Computing*, pages 1048–1075, 2002.
2. S. Cook. The complexity of theorem proving procedures. In *3rd Annual ACM Symposium on the Theory of Computing*, pages 151–158, 1971.
3. J. Friedman and A. Goerdt. Recognizing more unsatisfiable random 3-SAT instances efficiently. In *28th International Colloquium on Automata, Languages and Programming*, volume 2076 of *Lecture Notes in Computer Science*, pages 310–321. Springer-Verlag, 2001.
4. J. Hastad. Some optimal inapproximability results. *Journal of the ACM*, 48:798–859, 2001.
5. R. Impagliazzo. A personal view of average-case complexity. In *10th IEEE Structure in Complexity Theory*, pages 134–147, 1995.