Electronic Colloquium on Computational Complexity, Report No. 122 (2006)

All Natural NPC Problems Have Average-Case Complete Versions

Noam Livne [*]

September 20, 2006

Abstract

In 1984 Levin put forward a suggestion for a theory of *average case complexity*. In this theory a problem, called a *distributional problem*, is defined as a pair consisting of a decision problem and a probability distribution over the instances. Introducing adequate notions of simple distributions and average case preserving reductions, Levin developed a theory analogous to the theory of NP-completeness. In particular, he showed that there exists a simple distributional problem that is complete under these reductions. But since then very few distributional problems were shown to be complete in this sense. In this paper we show a very simple sufficient condition for an NP-complete decision problem to have a distributional version that is complete under these reductions. Apparently all known NP-complete decision problems meet this condition.

1 Introduction

The theory of average case complexity, initiated by Levin [9], refers to the complexity of solving problems with respect to certain probability distributions on their instances. Average case complexity, thus, is concerned with *distributional problems*, defined as pairs consisting of some decision problem and a probability distribution over all strings. Solving such a problem, means providing an algorithm that solves all instances and, loosely speaking, runs in expected polynomial time (or, alternatively, that runs in polynomial time and decides the problem with high probability over the related distribution of the inputs). In some sense, one can regard this complexity measure as measuring the complexity of instances that can ”really emerge in real life”.

[*]This research was partially supported by the Israel Science Foundation (grant No. 460/05).

ISSN 1433-8092

Levin [9] set the foundations to an average case complexity theory analogous to the theory of NP-completeness. His first goal was to define what are the "interesting" probability distributions. Letting these probability distributions range over all possible probability distributions would have collapsed the new theory to classic worse-case complexity (since one can always put all the probability mass on the worse case). On the other hand, considering only the uniform distribution seems quiet arbitrary. Levin therefore defined a restricted family of probability distributions, which he called *P-computable*. These are probability distributions over all strings, such that the accumulative probability can be computed in polynomial time (that is, there exists a polynomial time algorithm that given x outputs the probability that a string smaller or equal lexicographically to x is drawn). Focusing on these probability distributions Levin defined:

- The class avgP, which is analogous to $\mathcal{P}$, and consists of the distributional problems that can be solved "efficiently on the average".
- The class distNP, which is analogous to $\mathcal{NP}$, and consists of decision problems in NP paired with P-computable probability distributions.
- A class of reductions, which we call here AP-reductions, analogous to polynomial-time reductions (such as Karp or Cook reductions). Such reductions preserve "easiness on the average", that is, if a distributional problem can be AP-reduced to a problem in avgP, then the reduced problem is also in avgP. Although we did not specify yet what it means to solve a problem on the average, the crucial point is that these AP-reductions preserve "easiness on the average" with respect to various different definitions, including the original ones of Levin. The crucial aspect in these reductions is that instances that occur with some probability are not mapped to instances that occur with much smaller probability.

Next, Levin showed that there exists a distNP-complete distributional problem, that is, a problem in distNP that every problem in distNP can be AP-reduced to it .Thus, this complete problem is in avgP if and only if $\mathrm{avgP} \subseteq \mathrm{distNP}$. However, unlike the case of the (standard) theory of NP-completeness, in this new theory new complete problems were not easily found. In fact, to date, only a few distNP-complete problems were found [9, 4, 10, 5]. This is probably due to the fact that the properties needed from AP-reductions are more complex than the ones needed from the classic reductions.

In this work we show a simple sufficient condition for a $\mathcal{NP}$-complete decision problem to have a distributional version that is distNP-complete. Apparently all known $\mathcal{NP}$-complete decision problems meet this condition. This condition refers to some natural paddability property.

Our technique is based on the identification and construction of a restricted type of Karp-reductions that "preserve order" in some (natural) sense. If such an order preserving reduction exists from some (decisional part of a) distNP-complete problem to some problem in $\mathcal{NP}$, then the later has a probability distribution that when coupled with it, formes a distNP-complete decision problem. The aforementioned order preserving reduction is related to the paddability property mentioned in the previous paragraph.

Let us demonstrate, informally, the high-level ideas of our technique on SAT. Assume some standard encoding for SAT (we will freely identify a formula and its representation). Let (C, μ) be some distNP-complete distributional problem (so, in particular, $C \in \mathcal{NP}$ and μ is P-computable), and let h be a reduction from C to SAT such that $|x| \geq |y|$ if and only if $|h(x)| \geq |h(y)|$ (we will show in Section 3 how to achieve such reductions). We define a new Karp-reduction f such that $f(w)$ "encodes", in some explicit form, w itself into the formula $h(w)$. For example, let $e_0 = (x_0 \vee \neg x_0)$ and $e_1 = (x_1 \vee \neg x_1)$, and assume the encoding of e_1 is lexicographically larger than that of e_0. Now define

$$f(w_1 w_2 \ldots w_{|w|}) = e_{w_1} \wedge e_{w_2} \wedge \ldots \wedge e_{w_{|w|}} \wedge h(w) \tag{1}$$

where w_i is the i-th bit of w. Note that the "encoding" of w in the left part of the formula ensures that f has the following properties:

- Invertibility: given $f(w)$ one can compute w.
- Monotonicity: if w' is lexicographically larger than w'' then $f(w')$ is lexicographically larger than $f(w'')$.
- Preserving satisfiability: $f(w)$ preserves the truth value of $h(w)$ (since e_0 and e_1 are tautologies).

Thus, f is an "order preserving" reduction of C to SAT.

Let us see how such order preserving reductions (which are defined between *standard* decision problems) are related to AP-reductions (which are defined between *distributional* problems). We couple SAT with the following probability distribution η:

$$\eta(x) = \begin{cases} \mu(f^{-1}(x)) & \text{if } x \in \mathrm{image}(f) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Then the reduction f is a AP-reduction from (C, μ) to (SAT, η), because it maps each instance of C to an instance of SAT that occurs with exactly the same probability. Since (C, μ) is distNP-complete, and AP-reductions are transitive, it follows that (SAT, η) is distNP-hard (under AP-reductions). Furthermore, because of the special properties of f, and since μ is P-computable, then so is η. Loosely speaking, since f is monotonous, in order to compute the accumulative probability of w under η, it suffices to compute the accumulative probability of its inverse under μ; and since f is invertible, we can compute this inverse. It follows that (SAT, η) is in distNP, and therefore (SAT, η) is distNP-complete. For more details and a complete proof see Section 3.

We have just demonstrated that since such an ”order preserving” Karp-reduction exists between (the decisional part of) some distNP-complete problem and SAT, the later has a distributional version that is distNP-complete. Moreover, we note that the construction of the Karp-reduction exploited only the properties of the target problem, SAT. More specifically, the construction used a technique called ”padding”, introduced by Berman and Hartmanis [6], in order to encode w into $h(w)$. This ”paddability” property is a property of decision problems, rather than of reductions. Using this paddability property one can prove similar results for other $\mathcal{NP}$-complete problems.

Hence, essentially we ”reduced” the problem of showing that a $\mathcal{NP}$-complete decision problem has a distNP-complete version to the problem of proving some paddability properties for this decision problem. Although we do not know whether these paddability properties hold for every decision problem in $\mathcal{NP}$ (and showing that they do is at least as hard as proving $\mathcal{P} \neq \mathcal{NP}$), they are very easy to verify for any known problem. In particular, we have verified that these properties hold for the famous twenty-one problems treated in Karp's seminal paper [8]. See further discussion in Section 4.1

Reflection Let us take a second look at the probability distribution of the complete distributional version of SAT defined in Equation 2. We claim that this probability distribution has a ”simple” structure. We elaborate. All complete problems that are known to date have probability distributions that are ”close to uniform” in some sense. For simplicity, let us assume we take a complete problem with uniform probability distribution. Combining Equations 1 and 2, the left side of η is uniform over all encodings of strings (under some standard encoding). Thus, it can be regarded as ”close to uniform”. The right side is determined by the left, and it can be shown that it can also be made close to uniform in some sense. Thus, the structure of the resulted probability distribution is a simple structure. We elaborate more in Section 4.3.

Organization In section 2 we give some definitions that will be used throughout this paper. In Section 3 we provide a rigorous presentation of our results, by first showing a sufficient condition for an NP-complete decision problem to have a distributional version that is distNP-complete, and then, using this sufficient condition to show that some well-known NP-complete decision problems have distributional versions that are distNP-complete. In Section 4 we discuss some related issues: In Section 4.1 we discuss the generality of our results, and why they cannot be generalized to all problems in $\mathcal{NP}$. In Section 4.2 we discuss some alternative definitions of notions in average case complexity and show that our results hold under all of them. Finally, in Section 4.3 we discuss the interpretation of our results, and show that the resulted distributional problems can be regarded as simple in some sense.

2 Preliminaries

2.1 Strings and functions over strings

For a string x, we denote by $|x|$ the length of x. Throughout this paper, the symbol ”$<$”, when applied between strings, will denote the standard lexicographical order over all strings (i.e., $|y| = |y'| \Rightarrow x1y > x0y'$, and $|x| > |x'| \Rightarrow x > x'$). Given a string x, the strings $x - 1$ and $x + 1$ denote, respectively, the strings preceding and succeeding x.

Given an instance of a decision problem, its *characteristic* refers to the value of the characteristic function for this instance (i.e., it equals 1 if the problem contains this instance and 0 otherwise).

Definition 2.1 (P-invertible function) *A function f is* P-invertible *if it is 1-1, and there is a polynomial-time algorithm that given x returns $f^{-1}(x)$ if it is defined, and a failure symbol $\perp$ otherwise.*

Definition 2.2 (length-regular function) *A function f is* length-regular *if for every $x, y \in \{0,1\}^*$, it holds that $|x| \leq |y|$ if and only if $|f(x)| \leq |f(y)|$.*

Note that a function f is length-regular if and only if it satisfies the following two conditions: (1) $|x| = |y|$ if and only if $|f(x)| = |f(y)|$ and (2) $|x| > |y|$ if and only if $|f(x)| > |f(y)|$.

Definition 2.3 (semi-monotonous function) *A function f is* semi-monotonous *if for every $x, y \in \{0,1\}^*$ such that $|x| = |y|$ it holds that $x < y$ if and only if $f(x) < f(y)$.*

While a semi-monotonous function is only monotonous within lengths (that is, the function, when restricted to each length is monotonous), a function that is semi-monotonous and length-regular is monotonous over *all* strings (because in particular, for a length-regular function f, it holds that $|x| > |y|$ implies $|f(x)| > |f(y)|$).

2.2 Notions from average case complexity theory

We state here the basic definitions from average case complexity theory that will be used throughout this paper. These are the original definitions used by Levin in [9]. For a comprehensive survey on average case complexity, see Goldreich [3].

Definition 2.4 (probability distribution function) *A function $\mu : \{0,1\}^* \to [0,1]$ is a* probability distribution function *if $\mu(x) \geq 0$ for every x and $\sum_{x \in \{0,1\}^*} = 1$. The* accumulative probability function *associated with μ is denoted $\overline{\mu}$ and defined by $\overline{\mu}(x) = \sum_{x' \leq x} \mu(x')$.*

Definition 2.5 (P-computable probability distribution) *A* probability distribution function *μ is* P-computable *if there exists a polynomial time algorithm that given x outputs the binary expansion of $\overline{\mu}(x) = \sum_{x' \leq x} \mu(x')$.*

Definition 2.6 (distributional problem) *A* distributional problem *is a pair consisting of a decision problem and a probability distribution function. That is, (L, μ) is the distributional problem of deciding membership in the set L with respect to the probability distribution μ.*

Definition 2.7 (distNP) *The class distNP consists of all distributional problems (L, μ) such that $L \in \mathcal{NP}$ and μ is P-computable.*

Definition 2.8 (average-case preserving reduction) *A function f is an* average-case preserving reduction *(abbreviated AP-reduction) of the distributional problem (S, μ_S) to the distributional problem (T, μ_T) if f is a Karp-reduction (i.e. many-to-one polynomial-time reduction) from S to T, and in addition there exists a polynomial q such that for every $y \in \{0,1\}^*$,*

$$\mu_T(y) \geq \frac{1}{q(|y|)} \cdot \sum_{x \in f^{-1}(y)} \mu_S(x).$$

In the special case that f is 1-1, which is the case will be used throughout this paper, the last expression simplifies to the following: For every x it holds that

$$\mu_T(f(x)) \geq \frac{\mu_S(x)}{q(|x|)}.$$

Note that we use the fact that $|f(x)|$ is polynomially related to $|x|$.

AP-reductions preserve ”easiness on on the average” with respect to various definitions. The reason is that the sum of the probabilities of the preimages of every instance (in the range of the reduction), is not much larger than the probability of the instance itself. Thus, an AP-reduction cannot map ”typical” instances of the original problem to ”rare” instances of the target problem, on which an ”average-case algorithm” can perform exceptionally bad.

Definition 2.9 (distNP-complete distributional problem) *A distributional problem is* distNP*-complete if it is in* distNP *and every problem in* distNP *is AP-reducible to it.*

For sake of completeness, we state here the definition of avgP. However, our results only refer to AP-reductions, and not to their particular effect on avgP.

Definition 2.10 (avgP) *The class* avgP *consists of all distributional problems* (L, μ) *such that there exists an algorithm* A *that decides* L *and a constant* $\lambda > 0$ *such that*

$$\sum_{x \in \{0,1\}^*} \mu(x) \cdot \frac{t_A(x)^\lambda}{|x|} < \infty$$

where $t_A(x)$ *denotes the running time of* A *on input* x*.*

For a discussion on the motivation for this somewhat non-intuitive definition see Goldreich [2].

As mentioned above, it can be shown that if (T, μ_T) is AP-reducible to (S, μ_S) and $(S, \mu_S) \in \mathrm{avgP}$ then $(T, \mu_T) \in \mathrm{avgP}$ too. Thus, a distNP-complete problem is in avgP if and only if $\mathrm{distNP} \subseteq \mathrm{avgP}$. But, as mentioned above, AP-reductions preserve other definitions of ”easiness on the average” too.

3 Main Results

We state here a sufficient condition for the existence of a distNP-complete version for an NP-complete decision problem. We then show that some famous NP-complete decision problems meet this condition. By doing so we wish to claim that *all* known NP-complete decision problems meet this condition (or, at least have some reasonable encoding such that they do). For a discussion on the generality of our results, see Section 4.1

Our sufficient condition will enable us to prove completeness results also with respect to slightly different variants, like those of Goldreich [3] (which deal with probability ensembles rather than one probability distribution over all strings). This condition will be very easy to verify for all known $\mathcal{NP}$-complete decision problems. For more details on these alternative definitions, see Section 4.2.

3.1 The general technique

Our first technical tool is the following notion of paddability.

Definition 3.1 (regular-padding) *A decision problem L is* regular-paddable *if there exists some strictly increasing function q and a padding function $S : 1^* \times \Sigma^* \mapsto \Sigma^*$ such that:*

- *S is polynomial-time computable.*
- ***Preserving characteristic:** For every x and every n it holds that $S(1^n, x) \in L$ if and only if $x \in L$.*
- ***Length-regular**[1]**:** For every x and every n such that $n \geq |x|$, it holds that $|S(1^n, x)| = q(n)$.*

We call q the *stretch measure* of S. The first parameter of S determines the length to which the string is to be padded. The following holds:

Lemma 3.2 *If some decision problem is regular-paddable, then every Karp-reduction to it can be made length-regular.*

Proof: We show this by "pumping up" the lengths of all mapped strings. Let L be regular-paddable via S. Given a Karp-reduction f to L, we choose a strictly increasing polynomial r such that $r(|x|) \geq |f(x)|$ for every x, and define $f'(x) = S(1^{r(|x|)}, f(x))$. One can easily verify that f' is length-regular. $\square$

Our main technical tool is the following notion of paddability.

Definition 3.3 (monotonous padding) *A decision problem L is* monotonously-paddable *if there exists a padding function $E : \Sigma^* \times \Sigma^* \mapsto \Sigma^*$ and a decoding function $D : \mathbb{N} \times \Sigma^* \mapsto \Sigma^*$ such that:*

- *E and D are polynomial-time computable.*
- ***Preserving characteristic:** For every $p, x \in \{0,1\}^*$ it holds that $E(p, x) \in L$ if and only if $x \in L$.*
- ***Semi-monotonous:** If $|x_1| = |x_2|$, $|p_1| = |p_2|$, and $p_1 < p_2$ then $E(p_1, x_1) < E(p_2, x_2)$.*
- ***Length-regular:** If $|x_1| = |x_2|$ and $|p_1| = |p_2|$ then $|E(p_1, x_1)| = |E(p_2, x_2)|$, and if $|x_1| < |x_2|$ and $|p_1| < |p_2|$ then $|E(p_1, x_1)| < |E(p_2, x_2)|$.*

[1]Since this condition indeed resembles Definition 2.2, we allowed ourself this abuse of the term here and in the following definition.

- ***Decoding:*** *For every $x, p \in \{0,1\}^*$ it holds that $D(|p|, E(p,x)) = p$ and $D(k,w) = \perp$ if there is no x and p such that $|p| = k$ and $E(p,x) = w$*

Loosely speaking, the first parameter for D defines the part of the string to be regarded as the ”padding”. Note that D is well-defined, that is, if $D(k,w) \neq \perp$ then there exist a unique p such that $D(k,w) = p$ (i.e. a unique $p \in \{0,1\}^k$ such that there exists $x \in \{0,1\}^*$ such that $E(p,x) = w$). Although Definition 3.3 may seem somewhat cumbersome, the following holds:

Fact 3.4 *If the function E is defined such that $E(p,x) = E'(p_1)E'(p_2)\dots E'(p_{|p|})g(x)$, where:*

- *$E' : \{0,1\} \mapsto \{0,1\}^*$ encodes bits such that $|E'(0)| = |E'(1)|$ and $E'(0) < E'(1)$*
- *The function $g(x)$ is length-regular*
- *$E(p,x) \in L$ if and only if $x \in L$*

then E is a monotonous padding function for L.

In the example of SAT (used in the introduction), the function g is the identity function, but generally, the encoding does not necessarily only add some prefix to the string, but can also change the string in some simple way (for example, in the example of SAT the function g could also change the indexes of the variables in the original formula).

It is easy to see that famous $\mathcal{NP}$-complete decision problems are both regular-paddable and monotonously-paddable. For details see Sections 3.2, 3.3 and 3.4.

Theorem 3.5 *If L is $\mathcal{NP}$-complete, regular-paddable and monotonously-paddable then there is a distribution that when coupled with L forms a distNP-complete problem.*

Proof: We use the following result of Levin [9]:

Theorem 3.6 *There exists a distNP-complete distributional problem.*

Let (C, μ) be a distNP-complete distributional problem (where $C \in \mathcal{NP}$ and μ is P-computable), let h be a Karp-reduction from C to L, and let E, D be as in Definitions 3.1 and 3.3. In order for E to ”work properly” (that is, to yield a length-regular, semi-monotonous reduction), it has to be composed (in the appropriate manner), with a length-regular Karp-reduction. Thus, using Lemma 3.2, we transform h to a length-regular Karp-reduction h' of C to L. We then define $f(x) = E(x, h'(x))$. We notice that f enjoys the following properties:

- f is a Karp-reduction from C to L (since E preserves characteristic).
- f is length-regular (since h' and E are both length-regular).
- f is semi-monotonous (since h' is length-regular and E is semi-monotonous).
- f is P-invertible (see next).

P-invertibility is evidenced by the following algorithm: Given y it first tries to find a number k such that $|f(0^k)| = |y|$ (this can be done, e.g., by computing $|f(0)|, |f(0^2)|, \ldots, |f(0^{|y|})|$, capitalizing on $|f(x)| \geq |x|$, which follows from length-regularity). If no such k exists it returns $\perp$. Else, it computes $x = D(k, y)$. If $x = \perp$, the algorithm also returns $\perp$. Else, it computes $f(x) = E(x, h'(x))$. If the result equals y it returns x, else it returns $\perp$.

Recall that since f is length-regular and semi-regular, f is monotonous over all strings. Next, we couple the decision problem L with the following probability distribution η:

$$\eta(y) = \begin{cases} \mu(f^{-1}(y)) & \text{if } y \in \mathrm{image}(f) \\ 0 & \text{otherwise} \end{cases}$$

We claim that the reduction f is a AP-reduction from (C, μ) to the distributional problem (L, η), and that η is P-computable. Since AP-reductions are transitive, the theorem follows. The first claim is straightforward, since every instance of C is mapped to an instance of L of exactly the same probability (i.e., $\mu(x) = \eta(f(x))$). To see the second claim, recall that μ is P-computable, and note that the accumulative probability function induced by η, denoted $\overline{\eta}$, satisfies:

$$\overline{\eta}(y) = \overline{\mu}(x) \text{ where } x \text{ is the largest string such that } f(x) \leq y. \tag{3}$$

where $\overline{\mu}$ is the accumulative probability function induced by μ. We elaborate. Suppose, as an intermediate step, that we wish to compute $\eta(x)$ rather then $\overline{\eta}(x)$. Then we can simply compute $y = f^{-1}(x)$ (which can be done since f is P-invertible), then if $y = \perp$ we output 0, otherwise we output $\mu(y)$. Hence, the mere fact that f is P-invertible is sufficient to compute $\eta(x)$. Turning to the task of computing $\overline{\eta}$, we notice that since f is also monotonous (over all strings), for any string x in $\mathrm{image}(f)$, it holds that $\overline{\eta}(x) = \overline{\mu}(f^{-1}(x))$. For any other string, its accumulative probability is equal to that of the largest string in $\mathrm{image}(f)$ that is smaller than it (since all strings between them occur with probability 0). Equation 3 follows.

The string $\max(\{x | f(x) \leq y\})$ can be computed in polynomial time, since the reduction f is monotonous. An algorithm for computing this string can first compute $x = f^{-1}(y)$. If $x \neq \perp$ it outputs x, else it performs a binary search to find the string x' such that $f(x') < y$ and $f(x' + 1) > y$, and outputs x'. $\square$

Using Theorem 3.5 we now turn to prove that some $\mathcal{NP}$-complete decision problems have distNP-complete distributional versions. We have verified that all twenty-one $\mathcal{NP}$-complete decision problems that are treated in Karp's paper [8] do meet the sufficient condition of Theorem 3.5. In the rest of this section we describe three of them. The first one is SAT, which we chose since it is the most canonical $\mathcal{NP}$-complete decision problem. We then show that CLIQUE meets this condition, as an example of a typical graph problem. Finally we provide a proof that the same is true for HAM, the problem of Hamiltonian cycle, since this proof is a little less straightforward than the other problems in Karp's paper. We believe that these three examples in particular, and the fact that same results hold for all NP-complete decision problems in Karp's paper, give strong evidence that the results hold for all known $\mathcal{NP}$-complete decision problems.

3.2 SAT, revisited

The following theorem can be proved using Theorem 3.5.

Theorem 3.7 SAT *has a distributional version that is* distNP*-complete.*

To show that SAT meets the hypotheses of Theorem 3.5 one can use similar ideas to those presented in the introduction. We just have to assume some assumptions on the standard encoding of SAT (e.g. that the encoding acts on each clause, and each variable in the clause, in a context-free manner).

We choose two strings e_0, e_1 such that both are encodings of CNF clauses such that:

1. Both clauses are satisfiable.
2. $e_0 < e_1$
3. $|e_0| = |e_1|$

We first sketch the ideas used to show SAT is regular-paddable. In order to "stretch" some formula ϕ we "pad-up" ϕ by prefixing it with a series of e_0's. We then "shift" the variables in the original ϕ by raising their index, such that the variables in ϕ are disjoint to the ones in the added prefix. Since the added clauses are satisfiable, and consists of disjoint variables to the initial ϕ, the padding function does not affect the characteristic of ϕ.[2]

[2]There are various small technicalities to be concerned, like assuring that $|e_0|$ divides the difference between the desired length and the length of the initial formula. However, there are various ways of coping with such difficulties, e.g., by using various e_0's with different lengths, and by "normalizing" the lengths of the variables in ϕ (see next).

Following the ideas in the introduction, by using e_0 and e_1 to encode 0's and 1's, one can show that SAT is also monotonously-paddable. If it is required that instances of SAT do not have multiple occurrences of the same clause, then this requirement can be met by allocating sufficient amount of variables for the padding (i.e., using for the padding variables that are disjoint to the ones used in the original formula), and using different variables for each clause in the padding.

We did not define here rigorously the encoding of SAT (e.g., how is a variable encoded, how is a clause encoded, etc). Different encodings will yield different padding functions. However, Theorem 3.7 can be proved under any reasonable encoding of SAT.

In the following, we demonstrate our technique on two graph problems, given in matrix representation. For these problems the encoding will be defined rigorously, thus we will give a rigorous proof of our results.

3.3 Clique

We consider the CLIQUE decision problem, consisting of all pairs of an undirected graph G and a natural number k such that there exists a complete induced subgraph of G of size k. We assume the graph is given as an incidence matrix (which can either be symmetric, or upper-triangular), that the first row of the matrix is encoded by the leftmost bits, and that k is represented as a $\lceil \log(n) \rceil$-bit number to the right of the matrix.

Theorem 3.8 CLIQUE *has a distributional version that is* distNP*-complete.*

Proof: It is straightforward to see that CLIQUE is regular-paddable. We simply add "dummy" nodes with degree 0 and leave k as is. Thus we can transform any input of size $n^2 + \lceil \log(n) \rceil$ to an input of size $m^2 + \lceil \log(m) \rceil$ for any $m \geq n$, and thus we can achieve a regular-padding function with stretch measure $q(n) = n^2 + \lceil \log n \rceil$.

We now show that CLIQUE is monotonously-paddable. The idea is as follows. Given a graph $G = (V, E)$ where $V = \left\{v_1, v_2, \ldots, v_{|V|}\right\}$, we first "shift" all vertices by raising their index by the number of bits we wish to encode (and of course change the edges accordingly). This "frees" the vertices indexed lower or equal to the length of the string we wish to encode. We then encode the bits by edges connected to v_1, such that each bit is encoded by the edge indexed as the bit's position, and such that the edge will appear if and only if the bit value is 1. Thus, these edges will result in 0's and 1's in the first row of the incidence matrix of the graph. This will add a star-shaped subgraph (rooted at v_1) to the original graph. We will ensure that this will not change the characteristic of the instance.

Formally, for M, an 0-1-matrix of size $n \times n$ we define $E(p,(M,k)) = (M',k)$ where M' is an 0-1-matrix of size $(n+|p|)\times(n+|p|)$, such that $M'_{i+|p|,j+|p|} = M_{i,j}$ for $1 \leq i,j \leq n$, and $M'_{1,j} = p_j$ for $1 \leq j \leq |p|$ (where p_j is the j-th bit of p). This, of course, adds to the graph cliques of size 2 (but does not add larger cliques). If $k = 2$ and the graph did not have a clique of size 2 (i.e., the graph was edgeless), this could be a problem. To fix this, the padding function can check (in polynomial time) if indeed $k = 2$ and the graph is edgeless. If this is the case, it simply changes k to 3 and we are done.

This transformation preserves the characteristic of the instance. Moreover, we have p encoded in the most trivial manner, i.e. bit-by-bit, as a prefix of the string $E(p,x)$. It is straightforward to see that E meets all the conditions of a monotonous padding function.[3] $\square$

3.4 Hamiltonian Cycle

We consider the Hamiltonian Cycle decision problem, denoted HAM. The Hamiltonian Cycle decision problem consists of all undirected graphs that have a simple cycle that contains all nodes of the graph. We assume the graph is given as an incidence matrix (which can either be symmetric, or upper-triangular), and that the first row of the matrix is encoded at the beginning of the string.

Theorem 3.9 HAM *has a distributional version that is* distNP*-complete.*

Proof: We first show that HAM is regular-paddable. We do this by showing that any graph over n nodes can be transformed in polynomial time into a graph over $n+k$ nodes for any $k \geq 2$ such that preserves Hamiltonianicity. Given a graph $G = (V,E)$ where $|V| = n$, and $k \geq 2$ we define $G' = (V',E')$ where $|V'| = n+k$. Intuitively, what we are going to do is to "split" v_n (an arbitrary choice) into $k+1$ nodes, denoted $v_n, v_{n+1}, \ldots, v_{n+k}$, and "force" any Hamiltonian cycle to regard them as one node, that is, any Hamiltonian cycle in the new graph will have to contain the sub-path $v_n, v_{n+1}, \ldots, v_{n+k}$ or its reverse. The idea is as follows: after adding the mentioned nodes to the graph, we connect all of them to

[3] We note that any other reasonable encoding can be shown do yield the same result. For example, if k was encoded to the left of the matrix, the constructed reduction here would not be monotonous, since the "encoding row" would not be added at the beginning of the string. To fix this, the function E could fix k for every length of x, thus disabling its effect on the lexicographical order: Given an input (M,k) where M is an $n \times n$ matrix, the reduction would transform it to a matrix M' of size $2n \times 2n$, then add a clique of size $n-k$ to the original graph, using the added nodes, and connect all nodes of the added clique to all nodes of the original graph. The reduction would then generate the instance (M',n), which has the same characteristic as (M,k).

form the path mentioned above. We then connect the last node, v_{n+k}, to all the nodes connected to v_n. Formally:

$$E' = E \cup \{(v_{n+i}, v_{n+i+1}) | 0 \le i \le k-1\} \cup \{(u, v_{n+k}) | (u, v_n) \in E\}\,.$$

We show that this transformation preserves Hamiltonianicity. For every Hamiltonian cycle x, v_n, y in G (where x and y are sub-paths), the path $x, v_n, v_{n+1}, \ldots, v_{n+k}, y$ is a Hamiltonian cycle in G'. On the other hand, for any Hamiltonian cycle in G', in order to reach the nodes $v_{n+1}, v_{n+2}, \ldots, v_{n+k-1}$, it has to be of the form $x, v_n, v_{n+1}, \ldots, v_{n+k}, y$ or of the form $x, v_{n+k}, v_{n+k-1}, \ldots, v_n, y$, and in both cases this yields that x, v_n, y is a Hamiltonian cycle in G.

We now show that HAM is monotonously-paddable. The idea is similar to the regular-padding described above. In addition to the added path, we encode bits by adding edges within the path. We do it such that the path will still have to be taken in the natural order of the nodes, and such that the added edges will encode the desired padding in the prefix, i.e. first row, of the incidence matrix. In order to achieve the later goal, we "shift" the nodes of the graph by raising their index, thus the added nodes posses the smallest indexes, and then replace v_1 by the path $v_1, v_2, \ldots, v_{|p|+3}$, similarly to the construction of the regular-padding. We then encode the bits of the padding such that the edge (v_1, v_3) encodes the first bit, the edge (v_1, v_4) encodes the second bit and so on. We skip (v_1, v_2) since this edge must anyway exist in order for the mentioned path to exist. This construction ensures that the added path will still have to be taken in its natural order in any Hamiltonian cycle. We describe the construction formally.

For the $n \times n$ incidence matrix M of the graph $G = (V, E)$, we define $E(p, M) = M'$ where M' is the incidence matrix of size $(n+|p|+2) \times (n+|p|+2)$ of the graph $G' = (V', E')$ where $|V'| = |V| + |p| + 2$ and

$$E' = \big\{(v_{i+|p|+2}, v_{j+|p|+2}) | (v_i, v_j) \in E\big\} \cup \big\{(v_1, v_{i+|p|+2}) | (v_1, v_i) \in E\big\} \cup$$
$$\{(v_i, v_{i+1}) | 1 \le i \le |p|+2\} \cup \{(v_1, v_{i+2}) | p_i = 1\}$$

(where p_i is the i-th bit of p). The first set in the union above is the original graph, with its nodes "shifted" by raising their index by $|p|+2$. The second set connects v_1 to the nodes $v_{|p|+3}$ is connected to (which are the nodes v_1 in the original graph was connected to, with their index "shifted"), the third set forms the path $v_1, v_2, \ldots, v_{|p|+3}$. Finally, the last set encodes p by edges connected to v_1 (thus the bits representing them will be encoded in the first row of the matrix).

We have that every node that was connected in the original graph to v_1 is now connected both to v_1 and to $v_{|p|+3}$, and that $v_1, v_2, \ldots, v_{|p|+3}$ is a rout in the new graph. Assuming the diagonal is all zeros[4], the resulted encoding of M' starts

[4]The bits in the diagonal are meaningless.

with '01', followed by the bits of p. Using similar argument to the one used to prove that the regular-padding preserves Hamiltonianicity, one can verify that indeed such transformation preserves Hamiltonianicity too (in particular, note that any Hamiltonian cycle in G', in order to meet $v_{|p|+2}$, has to contain the sequence $v_1, v_2, \ldots, v_{|p|+3}$ or its reverse). Again, it is straightforward to see that E meets all the conditions of a monotonous padding function. $\Box$

We note that the similar decision problem of Parameterized Hamiltonian Cycle, which consists of all couples of a graph and a natural number k such that there is a simple cycle over k nodes in the graph, is easier to be shown to have a distNP-complete version. The same is true for the similar decision problem over *directed* graphs.

4 Conclusions

We note that by the result of Impagliazzo and Levin [7], every distNP-complete problem is also complete[5] for the wider class of $\mathcal{NP}$ problems coupled with P-sampleable distributions (introduced in [1]).

4.1 On the generality of our results

A natural question arises regarding our results. Since apparently, for all known $\mathcal{NP}$-complete decision problems we can provide a proof that our result hold, can we expect to prove, using our techniques, that our result holds for *all* $\mathcal{NP}$-complete decision problems? Apparently the answer is negative. Recall that in order to prove that some $\mathcal{NP}$-complete decision problem has a distributional version which is distNP-complete, our technique involves proving that this problem is paddable in some particular manner. However, proving that all $\mathcal{NP}$-complete problems are paddabale, in particular, involves proving that all $\mathcal{NP}$-complete problems are infinite. But such a proof would imply $\mathcal{P} \neq \mathcal{NP}$ (because if $\mathcal{P} = \mathcal{NP}$ then any non-empty finite set is $\mathcal{NP}$-complete). For this reason we cannot hope to do better than prove these results for all *known* NP-complete decision problems using our techniques.

The same phenomena occurs with respect to the *isomorphism conjecture* of Berman and Hartmanis [6]. This conjecture states that every two NP-complete decision problems are related via a 1-1, onto, polynomial-time, and polynomial-time invertible Karp-reduction. Berman and Hartmanis showed that every two decision problems that are paddable in some simple manner, are related via such a reduction.

[5] In some relaxed notion. See proof of theorem 10.24 in [3].

They observed that the paddability condition holds for numerous $\mathcal{NP}$-complete decision problems and concluded that these problems are pairwise isomorphic. They conjectured that the same is true for *all* $\mathcal{NP}$-complete decision problems. Thus, both their result and ours build on some paddability properties that are very easy to verify for given $\mathcal{NP}$-complete decision problems, but that are very hard to generalize for *all* $\mathcal{NP}$-complete decision problems.

The paddability condition required for Berman and Hartmanis's result is slightly weaker than ours. Loosely speaking, they do not require monotonicity. However, typically their padding functions, as ours, encode bit by bit. Ensuring that the padding is added at the beginning of the string, and that the encoding of 1 is larger than the encoding of 0, results in a monotonous padding function. We mention that to date, no counter-example to Berman and Hartmanis's isomorphism conjecture was found. Furthermore, one can show, that their paddability condition is not only sufficient, but is also necessary[6]. That is: If a decision problem is isomorphic to SAT, then it is paddable in the sense they define. Thus, the fact that no counter-example to Berman and Hartmanis's isomorphism conjecture was found, implies that no non-paddable decision problem was found. We believe, although our condition is slightly stronger than Berman and Hartmanis's notion of paddability, that this gives a strong evidence that all known $\mathcal{NPC}$ problems meet our condition too.

4.2 The extension of our results to different definitions

An alternative to the definition of P-computable probability distribution is the *simple probability ensemble* defined by Goldreich [3]. A probability ensemble $\{X_n\}_{n\in\mathbb{N}}$ is a sequence of random variables such that X_n ranges over $\{0,1\}^n$. Such an ensemble is *simple* if there exists a polynomial-time algorithm such that for every n and every $x \in \{0,1\}^n$ it outputs the value $\sum_{x'\in\{0,1\}^n, x'\leq x} \Pr(X_n = x')$. That is, the condition refers to a sequence of finite probability distributions, rather than on one infinite probability distribution. The class avgP can then be defined analogously using such probability ensembles, and the class distNP can be defined analogously using simple probability ensembles rather then P-computable probability distributions. Under these definitions our results hold too. The reason is that any reduction f that is constructed using our technique is length-regular, and

[6]Let f be a 1-1, onto, polynomial-time, and polynomial-time invertible Karp-reduction of $L \in \mathcal{NPC}$ to SAT. Then in order to pad the instance x of the problem L with p, we first pad $f(x)$ with p using the padding function of SAT. Denote the result by w. We then compute $f^{-1}(w)$ to obtain the padded instance. In order to retrieve the padding from some string y, the decoding function first computes $f(y)$, and then uses the decoding function of the padding function of SAT to retrieve the padding from $f(y)$. To conclude, one has to show that this padding function is also length-increasing. This can be achieved, e.g., by modifying the padding function of SAT to pad, instead of p, a longer string, such as p concatenated to itself a polynomial number of times.

therefore meets the condition $|x| = |y|$ if and only if $|f(x)| = |f(y)|$, and is also semi-monotonous.

A different notion of solvability can also be considered. While in our definition of avgP we require that the algorithm solves the decision problem for all instances, one can consider a relaxation such that the algorithm runs in polynomial time, but solves ”almost all instances”, that is, for every polynomial p and for every n the probability that given an input of length n the algorithm errs on it is upper bounded by $\frac{1}{p(n)}$. (Note that in the former definition, the algorithm, although is permitted to run in super-polynomial time for a negligible fraction of the instances, has to be correct on all instances.) Since AP-reductions preserve ”easiness on the average” also under this definition of solvability, our results, which refer to AP-reductions, are relevant also to this definition.

4.3 The interpretation of our results

One can argue that our results, although addressing the challenge of providing a variety of distNP-complete distributional problems, raise some questions about the adequateness of the definitions, specifically about the definition of P-computable probability distributions, since the probability distributions of our complete problems are rather ”unnatural”.

We argue, however, that the resulted probability distributions of the complete problems constructed by our technique can be viewed, in some sense, as ”quasi-uniform”, and thus are not that easy to discard from a theory that aims at average-case with respect to various simple distributions. We elaborate. Typically our distribution can be partitioned into two parts: the right side is determined by the initial reduction we take, and the left side is the padding (of both types). This holds, for example, for the three problems mentioned in this paper.

Since the distribution of the distNP-complete problem provided by Theorem 3.6 is close to uniform, the left side can be regarded too as close to uniform, because it is a 1-1 simple encoding of quasi-uniform distributed instances. The right side is determined by the left (and can be computed in polynomial time from it), and can be regarded as a ”parity check” of the left side. We mention, that using Berman and Hartmanis's isomorphism conjecture (see Section 4.1), and using the fact that this conjecture also holds for all known NP-complete decision problems, we can make the right side also close to uniform in some sense, by taking the initial reduction to be as guaranteed by their results.

Thus, typically our probability distributions have a simple structure and are ”close to uniform” in some sense. Moreover, we argue that the mere existence of *any* kind of distribution that makes some problem distNP-complete, may imply that simpler, more ”natural” distributional versions exist as well.

Acknowledgements

Deep thanks to my advisor, Oded Goldreich, for many helpful discussions, for his advice in presenting the ideas in this paper, and for his clear surveys and lectures that presented me the topics treated in this paper.

References

[1] S. Ben-David, B. Chor, and O. Goldreich. On the theory of average case complexity. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 204–216, New York, NY, USA, 1989. ACM Press.

[2] O. Goldreich. Notes on levin's theory of average-case complexity. Technical Report TR97-058, ECCC, 1997.

[3] O. Goldreich. Computational complexity: A conceptual perspective, draft of a book, 2006. Unpublished manuscript, available from `http://www.wisdom.weizmann.ac.il/~oded/cc-book.html`.

[4] Y. Gurevich. Complete and incomplete randomized NP problems. In *Proceedings of IEEE FOCS'87*, pages 111–117, 1987.

[5] Y. Gurevich. Matrix decomposition problem is complete for the average case. In *In Proc. of the 31st IEEE Annual Syrup. on Foundation of Computer Science*, pages 802–811, 1990.

[6] J. Hartmanis and L. Berman. On isomorphisms and density of NP and other complete sets. In *STOC '76: Proceedings of the eighth annual ACM symposium on Theory of computing*, pages 30–40, New York, NY, USA, 1976. ACM Press.

[7] R. Impagliazzo and L.A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Proc. of the 31st IEEE Symp. on Foundation of Computer Science*, pages 812–821, 1990.

[8] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[9] Leonid A Levin. Average case complete problems. *SIAM J. Comput.*, 15(1):285–286, 1986.

[10] Ramarathnam Venkatesan and Leonid Levin. Random instances of a graph coloring problem are hard. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 217–222, New York, NY, USA, 1988. ACM Press.