The Complexity of Explicit Constructions

Rahul Santhanam
University of Edinburgh
rsanthan@inf.ed.ac.uk

Abstract

The existence of extremal combinatorial objects, such as Ramsey graphs and expanders, is often shown using the probabilistic method. It is folklore that pseudo-random generators can be used to obtain explicit constructions of these objects, if the test that the object is extremal can be implemented in polynomial time. In this paper, we pose several questions geared towards initiating a structural approach to the relationship between extremal combinatorics and computational complexity. One motivation for such an approach is to understand better why circuit lower bounds are hard. Another is to formalize connections between the two areas, so that progress in one leads automatically to progress in the other.

1 Introduction

The field of extremal combinatorics deals with objects that are as large or small as possible given certain specified constraints. These objects might be graphs (eg., Ramsey graphs, expanders, extractors), sets or families of sets (eg., error-correcting codes, designs), matrices (eg, rigid matrices),or sequences (eg., universal traversal sequences). The study of such extremal objects is motivated by the fact that they are extremely useful in algorithmic contexts, eg., expanders and extractors are useful in designing efficient data structures and small sorting networks, error-correcting codes are critical to transmission of information in the presence of noise, and universal traversal sequences are useful in derandomizing certain randomized space-bounded algorithms.

In most cases, good bounds on the sizes of extremal objects can be obtained by using the probabilistic method initiated by Paul Erdos - to prove that there is an object of size S satisfying certain specified constraints, prove that a *random* object of this size satisfies the constraints with positive probability. The probabilistic method has been spectacularly successful in obtaining good bounds for various extremal objects [AS92].

However, the algorithmic applications of extremal objects often require "explicit" constructions of these objects. The standard meaning of "explicit" is that the object should be produced by a deterministic procedure running in polynomial time in the size of the object. The probabilistic method is inherently non-constructive, and so does not yield explicit constructions in this sense.

Note that there is a naive procedure for deterministically producing an extremal object of size S once we are guaranteed existence. Namely, enumerate all objects of size S, check for each one whether it satisfies the desired contraints, and output the first one which does. However, this procedure takes time exponential in S, even if the check whether the constraints hold can be done in polynomial time.

The question of explicit constructions has been studied intensively for various combinatorial objects. For objects such as expanders, extractors and codes, explicit constructions are known with

near-optimal parameters. For other objects such as Ramsey graphs, rigid matrices and universal traversal sequences, explicit constructions are still unknown despite years of effort. In any case, for a long time the question of explicit constructions was typically studied in isolation for each different combinatorial object. Recently, complexity theory and specifically the theory of pseudo-randomness have motivated progress towards a unified theory. In this paper, we raise several questions and initiate directions that promise progress in both the theory of explicit constructions and in complexity theory.

1.1 The Role of Complexity Theory

The major open problems in complexity theory are about separating complexity classes. Is there a Boolean function in NP which is not in P? Is there a Boolean function in EXP which does not have polynomial-size circuits? A priori, it is not clear what complexity lower bounds have to do with explicit constructions. However, it is a simple observation that virtually all complexity lower bounds we wish to prove hold for a *random Boolean function* - i.e., the probabilistic method applies to these questions. This is a highly "non-explicit" construction; making it explicit corresponds to upper bounding the complexity of a Boolean function for which the lower bound holds, and hence to separating complexity classes. For example, the question of proving that EXP does not have polynomial-size circuits is precisely the question of finding a polynomial-time procedure for outputting the truth table of a function which does not have polynomial-size circuits (here we mean polynomial-time in the size of the truth table, and hence exponential time in the number of input bits). This is an explicit construction question.

The question of constructing hard Boolean functions is not just one explicit construction question among many - it has relevance to all the others. This connection is through the beautiful theory of pseudo-random generators, initiated by Blum,Micali [BM84] and Yao [Yao82] from cryptographic motivations, and extended by Nisan and Wigderson [NW94] to a complexity-theoretic setting. Given a class C of statistical tests, a pseudo-random generator with seed $s(n)$ against C is a function G from $s(n)$ bits to n bits such that that no test from C can distinguish the uniform distribution on n bits from the uniform distribution on the range of G. If $s(n)$ is much smaller than n (e.g. $s(n) = O(\log(n))$), this means that the uniform distribution on the range of G "looks" random to C, even though it is in fact very far from random. Thanks to work of Nisan-Wigderson, Impagliazzo-Wigderson [IW97] and many others, it's known that the existence of pseudo-random generators with seed length $O(\log(n))$ against C which can be computed "efficiently" (i.e., in time polynomial in n) is equivalent to the existence of functions in linear exponential time which do not have sub-exponential size circuits with oracle access to some test in C.

Now, for a given explicit construction question where existence has been shown using the probabilistic method, let C be any class of tests which includes the test A for whether an object is extremal. A random object passes the test A with probability close to 1 - this is what the probabilistic method gives us. Now, if G is a pseudo-random generator with logarithmic seed length against C, then by pseudo-randomness of G, a random element in the range of G satisfies A with probability close to 1; hence at least one element in the range satisfies A. Thus, the explicit construction of a function in linear exponential time with no sub-exponential size C-oracle circuits implies an explicit construction of a polynomial-size set which contains an extremal object for *any* notion of "extremal" which can be tested within C! In the case that the extremality test is in P, we can identify efficiently an object in the set which is indeed extremal. Thus appropriate circuit lower bounds give explicit constructions for any object whose extremality can be tested in P!

These are folklore observations which have certainly played a role in the development of the field of complexity-theoretic pseudo-randomness. Specifically, as surveyed by Trevisan [Tre06] and Vadhan [Vad07], *conditional* constructions of pseudo-random generators from Boolean function hardness are closely tied to explicit constructions of objects such as expanders, extractors and error-correcting codes. However, there are still many avenues to be explored, and the observations in the previous paragraph leave many questions unanswered. Are there plausible complexity-theoretic hypothesis under which explicit constructions follow even when the test for extremality is not in polynomial time? What is the evidence for and against the extremality test being in polynomial time for questions such as whether a graph is Ramsey, and how does this connect with one-way functions in cryptography and the "natural proofs" framework of Razborov and Rudich [RR97]? Can we define new classes of tests containing natural explicit construction questions for which generic explicit constructions can be shown unconditionally? Is there a reasonable theory of reductions between explicit construction questions, as there is for decision problems, and if so, are there natural "hard" or "complete" questions?

We attempt to formulate these questions in a precise way, and to explore their implications. Underlying our investigation is a fundamental belief - that the connection between complexity lower bounds and explicit constructions , while mysterious, is far from accidental, and that understanding this connection more deeply is key to progress in both areas.

2 Preliminaries

2.1 Complexity-Theoretic Notions

For basic complexity notions, such as definitions of standard complexity classes, we refer the reader to the textbook by Arora and Barak [AB09]. Given a function $s : \mathbb{N} \to \mathbb{N}$, $\mathsf{SIZE}(s)$ is the class of Boolean functions computable by circuits of size $O(s(n))$. Given a complexity class C, $i.o.C$ is the class of languages L' for which there is $L \in C$ such that for infinitely many n, $L' \cap \{0,1\}^n = L \cap \{0,1\}^n$.

We require a standard notion of Kolmogorov time-bounded complexity - for more information on Kolmogorov complexity, see the excellent book by Li and Vitanyi [LV93]. Fix a universal Turing machine U which incurs at most a polynomial overhead in simulating any program p. We think of U as having two inputs - the program p and an auxiliary input x, as well as an output tape. For the purpose of defining time-bounded Kolmogorov complexity R_{Kt}, we will take the auxiliary input to be the empty string. Given a string x, $R_{Kt}(x)$ is the minimum over all programs p such that $U(p) = x$ of $|p| + \log(t_p)$, where t_p is the time taken for the computation of p on the empty string to terminate. Just as with other notions of Kolmogorov complexity, for any string x, $R_{Kt}(x) \leqslant |x| + O(1)$, and by a counting argument, for any positive integer n, there is at least one string x of length n such that $R_{Kt}(x) \geqslant n$.

2.2 Combinatorial Properties and Explicit Constructions

Definition 1. *A property is a set $S \subseteq \{0,1\}^*$. For a set $J \subseteq \mathbb{N}$, S is dense on J if $|S \cap \{0,1\}^n| \geqslant (1 - o(1))2^n$ for all $n \in J$. S is said to be dense if S is dense on $\mathbb{N}$.*

Note that from a formal mathematical point of view, a property is exactly the same as a language over the binary strings. But we use a different name to emphasize that the interpretation is different

- we're not interested in classifying instances efficiently but rather in generating instances satisfying the property.

Definition 2. *Let C be a class of functions from $\{0,1\}^*$ to $\{0,1\}^*$. A property S has C-explicit constructions if there is a function $f \in C$ such that $f(1^n) \in S \cap \{0,1\}^n$ whenever $S \cap \{0,1\}^n$ is non-empty. S is said to have C-explicit i.o. constructions if there is $f \in C$ such that $f(1^n) \in S \cap \{0,1\}^n$ for infinitely many n for which S contains a string of length n.*

Throughout this paper, by *explicit constructions*, we mean C-explicit constructions where C is the class of polynomial-time computable functions.

Definition 3. *Let C be a class of functions from Boolean strings to sets of Boolean strings, and let $q : \mathbb{N} \to \mathbb{N}$ be a function. S has a C-explicit list-construction of size $q(n)$ if there is a function $f \in C$ such that $f(1^n) \cap S \cap \{0,1\}^n \neq \varnothing$ whenever $S \cap \{0,1\}^n$ is non-empty, and the cardinality of the set $f(1^n)$ is bounded from above by $q(n)$ for any n.*

Again, when we drop the prefix for "explicit", we take C to be the class of polynomial-time computable functions.

Next, we define some natural properties for which it's an interesting open problem to find explicit constructions. We will assume some natural encoding of graphs as binary strings, say using the adjacency matrix. For technical reasons, we will pad these encodings so that a binary string whose length is not a square represents the graph represented by the largest initial prefix which is a square.

Definition 4. *Given functions $k : \mathbb{N} \to \mathbb{N}$ and $l : \mathbb{N} \to \mathbb{N}$, $Ramsey(k,l)$ is the set of graphs $G = (V,E)$ such that G does not have a clique of size $k(|V|)$ or an independent set of size $l(|V|)$.*

Our default setting for k and l is $k(n) = l(n) = 2\log(n)$, and for this k and l, we will simply call the property the Ramsey property. It was famously shown by Paul Erdos that Ramsey is dense. Density of the Ramsey property implies that a random graph is very likely to be Ramsey - this gives an algorithm for generating Ramsey graphs which runs in probabilistic polynomial time and succeeds with high probability. It's a longstanding open problem to obtain explicit constructions of the Ramsey property. For a long time, the best known explicit construction was due to Frankl and Wilson [FW81], and worked for $Ramsey(k,l)$ where $k(n) = l(n) = O(2^{\sqrt{\log(n)\log(\log(n))}})$; recently this was improved by Barak, Rao, Shaltiel and Wigderson [BRSW06], who found explicit constructions of $Ramsey(k,l)$ for $k(n) = l(n) = 2^{(\log(n))^{\epsilon}}$, where $\epsilon > 0$ is any constant. Note that this is still very far from explicit constructions of Ramsey.

There are other natural graph properties such as the Extractor and Expander properties which are interesting from the point of view of explicit constructions - we will define these properties if and when needed.

Next, we describe the Rigidity property, which is a property of matrices with applications to circuit lower bounds. We assume a standard encoding of square matrices as binary strings. We will assume this representation is padded just as in the case of graphs. We describe the rigidity property only for the 2-element field; an analogous definition can be made for any field.

Definition 5. *Given constants $1 > \epsilon, \delta > 0$, $Rigid(\epsilon,\delta)$ is the set of matrices M of dimension n over the field F_2 such that the rank of M cannot be reduced below ϵn without changing at least $n^{1+\delta}$ entries of M.*

Valiant [Val77] proved that the Rigidity property is dense for any $0 < \epsilon, \delta < 1$; he also showed that explicit constructions of rigid matrices would yield an explicit lower bound against linear-size, logarithmic-depth circuits whose gates are binary linear operations over the field. Despite many efforts, we're not close to obtaining explicit constructions of the Rigidity property - the history of the problem is surveyed by Lokam [Lok09].

The Primality property needs no introduction. We assume the natural encoding of natural numbers as strings.

Definition 6. *Primes is the set of prime numbers.*

Primes is not dense but by the prime number theorem, $|Primes \cap \{0,1\}^n| = \Omega(2^n/n)$. The celebrated result of Agrawal, Kayal and Saxena [AKS02] shows that $Primes \in \mathsf{P}$. No explicit constructions of the Primes property are known. This question was recently the focus of a Polymath project initiated by the mathematician Terence Tao.

The question of whether there are languages in E without small circuits is a major open question. This can be formulated as an explicit construction question about the property $MIN{-}CKT$ studied by Kabanets and Cai [KC00].

Definition 7. *$MIN - CKT$ is the property defined as follows: When the input length $|x| = n$ is of the form 2^m, for some integer m, then $x \in MIN - CKT$ if it is the truth table of a function on m bits which does not have Boolean circuits of size $\sqrt{n}$. When $2^m < n < 2^{m+1}$, $x \in MIN - CKT$ if the first 2^m bits of x constitute the truth table of a function without Boolean circuits of size $2^{m/2}$.*

It's not hard to see that $MIN - CKT$ is dense, using a counting argument a la Shannon. Also $MIN - CKT \in \mathsf{coNP}$. Explicit constructions for $MIN - CKT$ would imply $\mathsf{BPP} = \mathsf{P}$ using an argument via pseudo-random generators described in the next subsection.

Explicit construction problems sometimes have additional parameters than just the size of the object to be generated. For example, one could parameterize the Ramsey property by two graphs H_1 and H_2 such that a graph G satisfies the property if any bi-colouring of G has either an induced copy of H_1 or an induced copy of H_2. Similarly, one could parameterize the Primes property by an integer m and have an integer n satisfy the property if it is prime and lies between m and $2m$. To model such situations, we define the notion of a parameterized property and consider explicit constructions thereof.

Definition 8. *A parameterized property is simply a relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$. R has explicit constructions if there is an algorithm which, given x and 1^m, where x is a binary string and m an integer, generates in polynomial time a string y such that $|y| = m$ and $R(x,y)$ holds, if such a string y exists. R has explicit constructions under the density promise if the algorithm is only required to generate a string satisfying the relation when the fraction of strings y of length m satisfying $R(x,y)$ is $1 - o(1)$.*

To observe that explicit constructions of parameterized properties generalize explicit constructions of properties, for each property S associate a parameterized property R consisting of all pairs $(1^n, y)$ where n is an integer, $y \in S$ and $|y| = n$. This correspondence also implies that explicit constructions of parameterized properties under the density promise generalize explicit constructions of dense properties.

2.3 Pseudorandom Generators

Our discussion of explicit constructions of dense properties requires some knowledge of pseudo-random generator. A pseudo-random generator is an efficiently computable function mapping short random strings to longer "pseudo-random strings" which fool statistical tests, in the sense that the statistical test cannot distinguish pseudo-random strings from purely random strings. A statistical test is modelled here as a Boolean function. The strength of the generator corresponds to the power of statistical tests it can fool.

Definition 9. *Let $T : \{0,1\}^* \to \{0,1\}$ be a statistical test, and let $T_n = T \cap \{0,1\}^n$ for any n. Given a function $s : \mathbb{N} \to \mathbb{N}$ and an error function $\delta : \mathbb{N} \to [0,1]$, a PRG G fooling T with seed length $s(n)$ and error δ is a sequences of functions $\{G_n\}$ such that G_n maps $s(n)$ bits to n bits, is computable in time $2^{O(s(n))}$, and $|\Pr_{x\in\{0,1\}^n} T(x) - \Pr_{y\in\{0,1\}^{s(n)}} T(G_n(y))| \leqslant \delta(n)$. For a class C of statistical tests, G fools C if it fools T for each $T \in C$.*

There is a fundamental connection between complexity lower bounds and pseudo-random generators, first made by Nisan and Wigerson [NW94] and built upon by several others [IW97, KvM02]. We state one version of the connection, which is useful for us.

Theorem 10. *[NW94, IW97, KvM02] Let S be any property. If there is an $\epsilon > 0$ such that E does not have S-oracle Boolean circuits of size $2^{\epsilon n}$ infinitely often, then there is a PRG G with seed length $O(\log(n))$ and error $1/poly(n)$ fooling S.*

Corollary 11. *[NW94, IW97] If there is an $\epsilon > 0$ such that E does not have Boolean circuits of size $2^{\epsilon n}$ infinitely often, then $\mathsf{BPP} = \mathsf{P}$.*

3 The Relevance of Complexity Theory to Explicit Constructions

It has been observed by many researchers [Tre06] that the theory of pseudo-random generators can be used to provide explicit constructions of properties that are easy to test, under standard complexity lower bound assumptions.

Proposition 12. *Let S be any dense property. If $S \in \mathsf{P}$, then S has explicit constructions under the assumption there is an $\epsilon > 0$ such that $\mathsf{E} \not\subseteq i.o.\mathsf{SIZE}(2^{\epsilon n})$.*

Proof. Let M be a polynomial-time Turing machine deciding S. Using Theorem 10, under the assumption that $\mathsf{E} \not\subseteq i.o.\mathsf{SIZE}(2^{\epsilon n})$, there is a polynomial-time Turing machine M' which given n in unary, outputs a polynomial-sized set of strings of length n such that most strings in the set belong to S.

We define explicit constructions as follows: run M' on 1^n to obtain a polynomial-sized set of strings, then run M on each of these strings in succession until a string y is found for which M accepts on y. Halt and output y.

This procedure runs in polynomial time since M' and M run in polynomial time. □

Indeed, for the conclusion of Proposition 12 to hold, it is sufficient for the property to be testable in BPP, since the hardness assumption made also implies $\mathsf{BPP} = \mathsf{P}$ [IW97]. This observation was made to me by Lance Fortnow.

Proposition 12 cannot be used directly to show that *Primes* has explicit constructions if E does not have sub-exponential size circuits infinitely often. But by using a δ-PRG, where $\delta = 1/n^2$ together with the fact that a fraction $\Omega(1/n)$ of numbers of length n are prime for any n, the same proof goes through to show explicit constructions for the primes.

Corollary 13. *If there is an $\epsilon > 0$ such that $\mathsf{E} \not\subseteq i.o.\mathsf{SIZE}(2^{\epsilon n})$, then Primes has explicit constructions.*

The polynomial-time procedure witnessing the (conditional) explicit constructions of Proposition 12 has time complexity greater than the complexity of verifying the property. It turns out that this is not merely an artifact of the proof technique - there are dense easy properties for which explicit constructions require arbitrarily high polynomial-time complexity.

Theorem 14. *For any k, there is a dense property $S_k \in \mathsf{P}$ such that there are no explicit constructions for S_k in time $O(n^k)$.*

Proof. Define S_k to be the set of strings such that $R_{Kt}(x) \geqslant (k+2)\log(|x|)$. S_k is clearly dense. $S_k \in P$ because in polynomial time, we can enumerate all strings with R_{Kt}-complexity at most $(k+2)\log(n)$, and check whether the input string is in this list.

The proof that there are no explicit constructions for S_k within time $O(n^k)$ is by contradiciton. Assume that there is a Turing machine M running in time $O(n^k)$ which yields explicit constructions of S_k. The string output by M on input 1^n can be described by $\log(n) + O(1)$ bits (the description of M together with the binary representation of n) and can be recovered from this description in time $O(n^k \mathsf{polylog}(n))$ (by assumption on running time of M). Thus, the R_{Kt} complexity of this string is $(k+1)\log(n) + o(\log(n))$, which means that it cannot be in S_k for n sufficiently large. □

How about if a property is not easy, but is dense? The proof idea of Proposition 12 can be used to show that even if the property is not known to be easy to test, under a strong enough hardness assumption, we can get explicit list-constructions of size $\mathsf{poly}(n)$.

Proposition 15. *Let S be a dense property. If E does not have S-oracle circuits of size $2^{\epsilon n}$ infinitely often for some $\epsilon > 0$, there are explicit list-constructions of size $\mathsf{poly}(n)$ for S.*

Proof. Using Theorem 10, under the assumption that E does not have S-oracle circuits of size $2^{\epsilon n}$ infinitely often for some $\epsilon > 0$, there is a PRG G against S mapping seeds of length $O(\log(n))$ to strings of length n. The range of this PRG is computable in polynomial time and is a list of strings such that at least one belongs to S, using the pseudo-randomness property and the fact that S is dense. □

Since most combinatorial properties of interest are in NP or coNP, lower bounds for E against circuits with SAT-oracle gates generally suffice to apply Proposition 15. But then a natural question arises: is there a fundamental reason why the conclusion weakens to explicit list-constructions rather than explicit constructions? This question was studied by Fortnow and Santhanam, who gave complexity-theoretic evidence that the weakening cannot be avoided.

In order to give such complexity-theoretic evidence, they consider parameterized properties. Note that the results of Proposition 12 and 15 apply also to explicit constructions of parameterized properties under the density promise. The result below, proved jointly with Lance Fortnow, states that it's unlikely we can find a common generalization of the two results giving explicit constructions of properties that are hard to test, under reasonable complexity-theoretic hypotheses.

Theorem 16. *If* $\mathsf{NP} \neq \mathsf{P}$*, then there is a parameterized property* $S \in \mathsf{NP}$ *and a parameterized property* $S' \in \mathsf{coNP}$ *such that* S *and* S' *do not have explicit constructions under the density promise.*

Proof Sketch.

We will first sketch how to use explicit constructions for parameterized properties in NP or coNP to decide satisfiability of formulae with either zero or one satisfying assignments. We will then use the classical result of Valiant and Vazirani [VV85] to strengthen this and get $\mathsf{NP} = \mathsf{P}$ from the explicit construction assumption.

The basic idea is that explicit constructions can be used to *prune* the space of candidate solutions. Let ϕ be a formula which has either one or zero satisfying assignments. We show how to prune the space of candidate solutions by a factor $1 - 1/n$; by applying this idea iteratively n^2 times, we get a polynomial-size space which can be searched exhaustively. Consider $\log(n)$-size prefixes of possible assignments. Define a prefix to be in the property if it is *not* the prefix of a satisfying assignment. Suppose there is a satisfying assignment and an explicit construction for this property. Then, we can get a string y of length $\log(n)$ in polynomial time which is not a prefix of the assigment - this allows us to prune the space of solutions by the stated factor. To apply the idea iteratively, we parameterize by y and re-encode the remaining strings as binary strings of the appropriate length. To define a property in NP, just check that there exists a satisfying assignment without the input prefix. To define it in coNP, check that *no* satisfying assignment has the input prefix. Since we assumed that there are either zero or one satisfying assignments, either of these properties work in the argument above.

To remove the assumption that there are zero or one satisfying assignments, we use the Valiant-Vazirani result to probabilistically reduce the satisfiability question to the unique satisfiability question. Thus, under the assumption about explicit constructions, we get $\mathsf{NP} \subseteq \mathsf{BPP}$ and hence $\mathsf{NP} = \mathsf{RP}$. But note that the assumption also clearly implies $\mathsf{RP} = \mathsf{P}$. Given any language $L \in \mathsf{RP}$, we can define a corresponding parameterized property where the parameter is the input and the property consists of those random strings which cause the input to accept. Using explicit constructions for parameterized properties under the density promise, we can find an accepting random string in polynomial time for inputs in L, and hence decide if an input is in L or not. □

To re-iterate, most properties, as well as parameterized properties, of interest are either in NP or in coNP.

Theorem 16 indicates that the complexity of testing a property is a key consideration when determining whether the property is likely to have explicit constructions or not. While, by a generalization of Proposition 12, parameterized properties that are easy to test have explicit constructions under the density promise if a widely-believed complexity hypothesis holds, Theorem 16 shows that if a different widely-believed complexity hypothesis holds, parameterized properties that are hard to test do not in general have explicit constructions under the density promise..

Our focus on dense properties is because the use of the probabilistic method to prove that a property is non-trivial also typically establishes that the property is dense. But how about properties that are not dense, but are nevertheless easy to test - could one expect explicit constructions in general for such properties?

It's not to hard to show that a similar argument as the one in Theorem 16 holds in this case as well.

Theorem 17. *There is a parameterized property* $S \in P$ *such that if* S *has explicit constructions, then* $\mathsf{NP} = \mathsf{P}$*.*

Proof. We simply use a parameterized property S that encodes satisfiability. S consists of all pairs $< \phi, w >$ where w is a satisfying assignment to the formula ϕ. If this property has explicit constructions, we can solve SAT in polynomial time as follows: Given an input formula ϕ, compute the number of variables m in ϕ and then run the explicit construction procedure for S on ϕ and 1^m. If the procedure returns a string w of length m, check that w is a satisfying assignment to ϕ, outputting YES if it is and NO if it is not. By the definition of explicit constructions for parameterized properties, if ϕ is indeed satisfiable, w will be a satisfying assignment to ϕ, hence this algorithm for SAT is correct. Since the explicit construction procedure runs in polynomial time, so does this algorithm. Clearly the parameterized property S is in P, as verifying whether an assignment satisfies a formula can be done in polynomial time. □

In fact, Theorem 17 can be strengthened to state that there is a property in AC^0 for which explicit constructions would imply a collapse of NEXP to EXP. This is because verification of whether an assignment satisfies a 3-CNF formula can be done in AC^0.

Theorems 16 and 17 have relevance to the natural proofs framework of Razborov and Rudich. The natural proofs framework focuses on studying properties of functions (represented as their truth tables) which imply circuit lower bounds. The main result of Razborov and Rudich [RR97] states that under a standard assumption about the existence of one-way functions, there cannot be lower bound properties which are both dense and easy. However, to establish a lower bound, it's not sufficient to define an appropriate lower bound property - one must also exhibit an explicit function satisfying the property. The results here mean that in general this is hard to do for (parameterized) properties which are not both dense and easy, i.e., properties which could conceivably be lower bound properties under the assumption that one-way functions exist.

The proofs of Theorem 16 and 17 apply to artificially defined properties rather than to natural combinatorial properties. Therefore we ask:

Question 18. *Is there a "natural" combinatorial parameterized property in* NP *or* coNP *such that explicit constructions for that property would refute a strongly-believed complexity conjecture?*

Given the preceding observations, it's clearly of interest to study the complexity of testing various natural properties. Consider the Ramsey property, for instance: it is known to be in coNP, but whether it is in P is unknown. Perhaps this is because of limitations in our algorithmic knowhow? After all, the *Primes* property was known for a long time to be in $\mathsf{NP} \cap \mathsf{coNP}$ but only recently was it shown to be in P.

We can give some evidence that the Ramsey property is unlikely to be easy. This is based on a minor variant of a hardness assumption first suggested independently by Jerrum [Jer92] and Kucera [Kuc95], which has since been used in a variety of contexts.

Definition 19. *Planted Clique Assumption: There is no polynomial-time algorithm which distinguishes the two distributions D_0 and D_1 on n-vertex graphs defined as follows. D_0 is the distribution induced by picking an Erdos-Renyi random graph on n vertices with edge probability $1/2$. D_1 is the distribution induced by picking an Erdos-Renyi random graph on n vertices with edge probability $1/2$ in which a random clique of size $3\log(n)$ has been planted.*

Theorem 20. *Under the Planted Clique Assumption, Ramsey $\notin \mathsf{P}$.*

Proof. The proof follows from the observation that an Erdos-Renyi random graph with edge probability $1/2$ satisfies the Ramsey property with probability close to 1, while a random graph with a planted clique of size $3\log(n)$ is not Ramsey. □

It would be very interesting to provide stronger evidence that a natural combinatorial property is not easily verifiable.

Question 21. *Is there a natural combinatorial property S such that if one-way functions exist, then $S \notin \mathsf{P}$?*

The Ramsey property is testable in quasi-polynomial time just by making an exhaustive search for cliques and independent sets of size $2\log(n)$. However, this does not seem to be the case for other properties such as the Rigidity property. Could one actually prove that this property is hard for the smallest complexity class known to contain it - coNP in this case? Such a hardness result might hold, but an idea of Kabanets and Cai [KC00] can be used to give evidence that this would require new techniques, since it would imply explicit constructions.

Theorem 22. *Let S be any property which is NP-complete or coNP-complete under polynomial-time honest m-reductions. Then S has explicit i.o.constructions.*

Proof. We establish the result for the case that S is NP-complete; the case that S is coNP-complete is similar. Let f be a polynomial-time honest m-reduction from SAT to S. Let $\{\phi_n\}, |\phi_n| = n$ be a polynomial-time constructible sequence of satisfiable formula - such a sequence is easily defined. Now consider the sequence $\{f(\phi_n)\}$. Since f is an honest m-reduction from SAT to S, we have that for infinitely many m, there is a ϕ_i such that $f(\phi_i) \in S \cap \{0,1\}^m$. Furthermore for infinitely many m, the first such string $f(\phi_i)$ can be found in polynomial time, simply by enumerating the first polynomially (in m) many i's and verifying if the length of $f(\phi_i)$ is m for each such i. Thus S has explicit i.o.constructions. □

Note that virtually all known examples of NP-completeness or coNP-completeness proofs are via polynomial-time honest m-reductions. The proof idea of Theorem 22 illustrates how hardness results can be used *positively* (to explicitly construct a member of a set) as well as *negatively* (to show that a set is hard).

Thus, before aiming to show that a property is hard for a complexity class, we should be able to produce explicit constructions for the property. But is producing explicit constructions the main obstacle, in some sense, to showing hardness results for certain natural properties? In the case of the superconcentrator property, Blum, Karp, Vornberger, Papadimitriou and Yannakakis [BKV$^+$81] showed that it was coNP-hard by using known explicit constructions of super-concentrators in their reduction.

Question 23. *Are there natural properties (apart from the Super-concentrator property) for which explicit constructions would imply hardness for some complexity class?*

Given that there's evidence of hardness for some natural properties, we could try to find ways around the "complexity barrier" to explicit constructions expressed by Theorems 16 and 17. One possible route is through the notion of *combinability*. If given a list of objects such that at least one of them satisfies a given property, we could produce a different object that is guaranteed to satisfy the property, then Theorem 15 would be useful in actually getting explicit constructions (under a hardness hypothesis).

Definition 24. *A property S is combinable if there is an $\epsilon > 0$ and a polynomial-time algorithm A which when given $y_1 \dots y_k$, $|y_i| = n$ for each $1 \leqslant i \leqslant k$, and $k = poly(n)$, produces in polynomial time a string $y \in S$ such that $|y| \geqslant n^\epsilon$.*

Combiners have been studied in the cryptographic context [HKN$^+$05]. In the combinatorial context, a weak version of combinability is shown for the Rigidity property for large fields by Klivans and van Melkebeek [KvM02].

Corollary 25. *If S is combinable and there is a $\delta > 0$ such that E does not have S-oracle circuits of size $2^{\delta n}$ infinitely often, then S has explicit i.o.constructions.*

Question 26. *Is the Ramsey property combinable?*

Another way to get around the complexity barrier is by making a different but still plausible complexity assumption under which explicit constructions for most "natural" combinatorial properties could be shown to exist, without worrying about the somewhat artificial counterexamples of Theorems 16 and 17. We present a candidate assumption.

Definition 27. *Condition H: Fix a universal log-space Turing machine U. There is a polynomial-time algorithm which on input 1^n outputs a string x_n such that $U(p) = x_n$ implies $|p| \geqslant n - \Omega(\log(n))$.*

Condition H is robust in the sense that it does not depend on the choice of the universal machine U.

Theorem 28. *Under Condition H, Ramsey(k, l) has explicit constructions when $k, l = \omega(\log(n))$.*

Proof. We provide a sketch. The key idea is that non-Ramsey graphs can be compressed so that the graph can be recovered from its compressed version by a logarithmic-space computation. The compressed version saves on bits by explicitly listing vertices in a large clique or independent set rather than listing edges. This gives savings when there is guaranteed to be a clique or independent set of size $\omega(\log(n))$. Condition H gives a polynomial-time algorithm to list strings which cannot be compressed in this manner, hence any such string is a representation of a Ramsey graph. □

Question 29. *How does Condition H relate to traditional complexity assumptions?*

Question 30. *Which other natural properties have explicit constructions under Condition H?*

The strongest argument for the relevance of complexity theory to explicit constructions would be to give previously unknown *unconditional* explicit constructions for some natural combinatorial property which follow from complexity-theoretic considerations. Suppose we wish to use the theory of pseudo-randomness to give explicit constructions for some property S. Even if S is testable in polynomial time, in general we would need circuit lower bound assumptions to achieve this, as in Theorem 12. There are a few ways one could imagine getting around this. First, by examining the proof that a random object satisfies a given property, i.e., perhaps one could design a randomized algorithm for generation which uses only $O(\log(n))$ random bits and can therefore be unconditionally derandomized as long as the property is efficiently testable. Second, if S is very easily testable, say in a complexity class C for which circuit lower bounds are already known, then by applying the idea in the proof of Theorem 12, we could hope to get unconditional explicit constructions. Third, we could try to exploit the fact that we only need generators fooling a *specific test*, i.e., S, rather than a class of tests. Namely, we could try and tailor the PRG we use to the specific property for which we seek explicit constructions.

As far as the author is aware, there are no interesting instantiations of the third approach yet. We apply the first and second approaches to the problem of explicit constructions of Ramsey graphs. The first approach has been used before in the context of various combinatorial properties [AS92, Spe94] while the second approach has been heavily used within complexity theory for unconditional derandomization in various contexts.

Theorem 31. *Ramsey has* $\mathsf{DTIME}(n^{O(\log(n))^2})$*-explicit constructions.*

Proof. We start with the observation that a graph chosen from a $3(\log(n))^2$-wise independent sample space on n^2 bits has a high probability of being Ramsey. This is because Erdos' application of the probabilistic method for this case only requires $O(\log(n))^2$-wise independence. Using known constructions of k-wise independent sample spaces for this value of k [AS92], we can generate in time $O(f(n))$ where $f(n) = n^{O(\log(n))^2}$ $f(n)$ candidate graphs, and check for each one in time $n^{O(\log(n))}$ whether it is Ramsey or not using exhaustive search. Most such graphs are Ramsey, hence sooner or later we will find a Ramsey graph using this method. The time required is $O(f(n))$. □

To apply the second approach, notice that *Ramsey* has constant-depth circuits of size $n^{O(\log(n))}$. Now, using the unconditional pseudo-random generator of Nisan for constant-depth circuits and applying the proof idea of Theorem 31, we get quasipolynomial time-explicit constructions. However, the exponent in the quasi-polynomial time bound is worse than that in Theorem 31.

Question 32. *Are there unconditional explicit poly*(n)*-list constructions of Ramsey graphs?*

Next, we address the question of reductions between explicit construction problems. The theory of algorithms has achieved its current richness mostly because of the wealth of reductions that are known between problems with regard to algorithmic solvability. We first formulate a natural notion of reduction between properties.

Definition 33. *Let S and T be properties. The explicit construction problem for T poly-time reduces to the explicit construction problem for S if there is a polynomial-time oracle Turing machine M with the following properties. On input 1^n, M only makes oracle queries of the form 1^m for some integer m. On such an oracle query, the oracle returns a string in $S \cap \{0,1\}^m$ if such a string exists, and an arbitrary string otherwise. At the end of its computation, M outputs a string in $T \cap \{0,1\}^n$, if such a string exists.*

Proposition 12 can be re-interpreted as a one-query polynomial-time reduction of the explicit construction problem for S to the explicit construction problem for $MIN - CKT$ for any $S \in \mathsf{P}$. However, $MIN - CKT$ is not known to be in P and indeed the natural proofs framework of Razborov and Rudich [RR97] indicates it is unlikely that $MIN - CKT \in \mathsf{P}$, as this would break any one-way function. It's natural to ask the following question.

Question 34. *Is there an explicit construction problem for a property $S \in \mathsf{P}$ such that every explicit construction problem for a property $T \in \mathsf{P}$ poly-time reduces to it? Is there such an explicit construction problem for a natural combinatorial property?*

The notion of reduction in Definition 33 can easily be extended to parameterized properties and a similar question about completeness could be asked in this setting. The surveys of Trevisan [Tre06] and Vadhan [Vad07] detail many connections between various combinatorial objects such as expanders, extractors and error-correcting codes. It would be interesting to formulate some of these connections in terms of the notion of reduction between explicit construction problems.

4 Acknowledgements

I am grateful to Lance Fortnow and Srikanth Srinivasan for several productive discussions. Lance kindly allowed me to include Theorem 16 in this paper. Thanks also to Tony Tan for pointing me to the paper by Spencer [Spe94].

References

[AB09] S. Arora and B. Barak. *Complexity Theory: A Modern Approach*. Cambridge University Press, Cambridge, 2009.

[AKS02] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. Report, Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, 2002.

[AS92] Noga Alon and Joel H. Spencer. *The Probabilistic Method, with an appendix on open problems by Paul Erdös*. John Wiley & Sons, 1992.

[BKV+81] Manuel Blum, Richard Karp, Oliver Vornberger, Christos Papadimitriou, and Mihalis Yannakakis. The complexity of testing whether a graph is a superconcentrator. *Information Processing Letters*, 13(4–5):164–167, 1981.

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequence of pseudo-random bits. *SIAM Journal on Computing*, 13:850–864, 1984.

[BRSW06] Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2-source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction. In *Proceedings of 38th Symposium on Theory of Computing*, pages 671–680, 2006.

[FW81] Peter Frankl and Robin Wilson. Intersection theorems with geometric consequences. *Combinatorica*, 1:357–368, 1981.

[HKN+05] Danny Harnik, Joe Kilian, Moni Naor, Omer Reingold, and Alon Rosen. Robust combiners for oblivious transfer and other cryptographic primitives. In *Proceedings of 24th International Conference on the Theory and Application of Cryptographic Techniques (CRYPTO '95)*, pages 96–113, 2005.

[IW97] Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pages 220–229, 1997.

[Jer92] Mark Jerrum. Large cliques elude the metropolis process. *Random Structures and Algorithms*, 3(4):347–359, 1992.

[KC00] Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In ACM, editor, *Proceedings of the thirty second annual ACM Symposium on Theory of Computing: Portland, Oregon, May 21–23, [2000]*, pages 73–79, New York, NY, USA, 2000. ACM Press.

[Kuc95] Ludek Kucera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2–3):193–212, 1995.

[KvM02] Adam Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial hierarchy collapses. *SIAM Journal of Computing*, 31(5):1501–1526, 2002.

[Lok09] Satyanarayana Lokam. Complexity lower bounds using linear algebra. *Foundations and Trends in Theoretical Computer Science*, 4(1–2):1–155, 2009.

[LV93] Ming Li and Paul Vitanyi. *Introduction to Kolmogorov Complexity and its Applications.* Springer-Verlag, August 1993.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

[RR97] Alexander Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

[Spe94] Joel Spencer. From erdos to algorithms. *Discrete Mathematics*, 136(1–3):295–307, 1994.

[Tre06] Luca Trevisan. Pseudorandomness and combinatorial constructions. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(13), 2006.

[Vad07] Salil Vadhan. The unified theory of pseudorandomness: Guest column. volume 38, pages 39–54, 2007.

[Val77] L. G. Valiant. Graph-theoretic arguments in low-level complexity. In J. Gruska, editor, *Proceedings of the 6th Symposium on Mathematical Foundations of Computer Science*, volume 53 of *LNCS*, pages 162–176, Tatranská Lomnica, Czechoslovakia, September 1977. Springer.

[VV85] Leslie Valiant and Vijay Vazirani. NP is as easy as detecting unique solutions. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, pages 458–463, 1985.

[Yao82] Andrew Yao. Theory and application of trapdoor functions. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 80–91, 1982.