Local Reductions

Hamid Jahanjou[1], Eric Miles[2(✉)], and Emanuele Viola[1]

[1] Northeastern University, Boston, MA, USA
{hamid,viola}@ccs.neu.edu
[2] UCLA, Los Angeles, CA, USA
enmiles@cs.ucla.edu

Abstract. We reduce non-deterministic time $T \geq 2^n$ to a 3SAT instance ϕ of quasilinear size $|\phi| = T \cdot \log^{O(1)} T$ such that there is an explicit circuit C that on input an index i of $\log |\phi|$ bits outputs the ith clause, and each output bit of C depends on $O(1)$ input bits. The previous best result was C in NC^1. Even in the simpler setting of polynomial size $|\phi| = \mathrm{poly}(T)$ the previous best result was C in AC^0.

More generally, for any time $T \geq n$ and parameter $r \leq n$ we obtain $\log_2 |\phi| = \max(\log T, n/r) + O(\log n) + O(\log \log T)$ and each output bit of C is a decision tree of depth $O(\log r)$.

As an application, we tighten Williams' connection between satisfiability algorithms and circuit lower bounds (STOC 2010; SIAM J. Comput. 2013).

1 Introduction

The efficient reduction of arbitrary non-deterministic computation to 3SAT is a fundamental result with widespread applications. For many of these, two aspects of the efficiency of the reduction are at a premium. The first is the length of the 3SAT instance. A sequence of works shows how to reduce non-deterministic time-T computation to a 3SAT instance ϕ of quasilinear size $|\phi| = \tilde{O}(T) := T \log^{O(1)} T$ [HS66, Sch78, PF79, Coo88, GS89, Rob91]. This has been extended to PCP reductions [BGH+05, Mie09, BCGT13, BCGT12].

The second aspect is the computational complexity of producing the 3SAT instance ϕ given a machine M, an input $x \in \{0,1\}^n$, and a time bound $T = T(n) \geq n$. It is well-known and easy to verify that a ϕ of size $\mathrm{poly}(T)$ is computable even by circuits from the restricted class NC^0. More generally, Agrawal, Allender, Impagliazzo, Pitassi, and Rudich show [AAI+01] that such NC^0 reductions exist whenever AC^0 reductions do.

A stronger requirement on the complexity of producing ϕ is critical for many applications. The requirement may be called *clause-explicitness*. It demands that the ith clause of ϕ be computable, given $i \leq |\phi|$ and $x \in \{0,1\}^n$, with resources $\mathrm{poly}(|i|) = \mathrm{poly} \log |\phi| = \mathrm{poly} \log T$. In the case $|\phi| = \mathrm{poly}(T)$, this is known to be possible by an unrestricted circuit D of size $\mathrm{poly}(|i|)$. (The circuit has either

Supported by NSF grants CCF-0845003, CCF-1319206.

M.M. Halldórsson et al. (Eds.): ICALP 2015, Part I, LNCS 9134, pp. 749–760, 2015.
DOI: 10.1007/978-3-662-47672-7_61

random access to x, or, if $T \geq 2^n$, it may have x hardwired.) As a corollary, so-called succinct versions of NP-complete problems are complete for NEXP. Arora, Steurer, and Wigderson [ASW09] note that the circuit D may be taken from the restricted class AC^0. They use this to argue that, unless EXP = NEXP, standard NP-complete graph problems cannot be solved in time $\mathrm{poly}(2^n)$ on graphs of size 2^n that are described by AC^0 circuits of size $\mathrm{poly}(n)$.

Interestingly, applications to unconditional complexity lower bounds rely on reductions that are clause-explicit and simultaneously optimize the length of the 3SAT instance ϕ and the complexity of the circuit D computing clauses. For example, the time-space tradeoffs for SAT need to reduce non-deterministic time T to a 3SAT instance ϕ of quasilinear size $\tilde{O}(T)$ such that the ith clause is computable in time $\mathrm{poly}(|i|) = \mathrm{poly}\log|\phi|$ and space $O(\log|\phi|)$, see e.g. [FLvMV05] or Van Melkebeek's survey [vM06]. More recently, the importance of optimizing both aspects of the reduction is brought to the forefront by Williams' approach to obtain lower bounds by satisfiability algorithms that improve over brute-force search by a super-polynomial factor [Wil13a, Wil11b, Wil11a, SW12, Wil13b]. To obtain lower bounds against a circuit class C using this technique, one needs a reduction of non-deterministic time $T = 2^n$ to a 3SAT instance of size $\tilde{O}(T)$ whose clauses are computable by a circuit D of size $\mathrm{poly}(n)$ that belongs to the class C. For example, for the ACC^0 lower bounds [Wil11b, Wil13b] one needs to compute them in ACC^0. However it has seemed "hard (perhaps impossible)" [Wil11b] to compute the clauses with such restricted resources.

Two workarounds have been devised [Wil11b, SW12]. Both exploit the fact that, under an assumption such as $\mathrm{P} \subseteq \mathrm{ACC}^0$, non-constructively there does exist such an efficient circuit computing clauses; the only problem is constructing it. They accomplish the latter using either nondeterminism [Wil11b] or brute-force [SW12] (cf. [AK10]). The overhead in these arguments limits the consequences of satisfiability algorithms: before this work, for a number of well-studied circuit classes C (discussed later) a lower bound against C did not follow from a satisfiability algorithm for circuits in C.

2 Our Results

We show that, in fact, it is possible to reduce non-deterministic computation of time $T \geq 2^n$ to a 3SAT formula ϕ of quasilinear size $|\phi| = \tilde{O}(T)$ such that given an index of $\ell = \log|\phi|$ bits to a clause, one can compute (each bit of) the clause by looking at a constant number of bits of the index. Such maps are also known as local, NC^0, or junta. More generally our results give a trade-off between decision-tree depth and $|\phi|$. The results apply to any time bound T, paying an inevitable loss in $|x| = n$ for T close to n.

Theorem 1 (Local reductions). *Let M be an algorithm running in time $T = T(n) \geq n$ on inputs of the form (x, y) where $|x| = n$. Given $x \in \{0,1\}^n$ one can output a circuit $D : \{0,1\}^\ell \to \{0,1\}^{3v+3}$ in time $\mathrm{poly}(n, \log T)$ mapping an index to a clause of a 3CNF ϕ in v-bit variables, for $v = \Theta(\ell)$, such that*

1. *ϕ is satisfiable iff there is $y \in \{0,1\}^T$ such that $M(x,y) = 1$, and*
2. *for any $r \leq n$ we can have $\ell = \max(\log T, n/r) + O(\log n) + O(\log\log T)$ and each output bit of D is a decision tree of depth $O(\log r)$.*

Note that for $T = 2^{\Omega(n)}$ we get that D is in NC^0 and ϕ has size $2^\ell = T \cdot \log^{O(1)} T$, by setting $r := n/\log T$. We also point out that the only place where locality $O(\log r)$ (as opposed to $O(1)$) is needed in D is to index bits of the string x.

The previous best result was D in NC^1 [BGH+05]. Even in the simpler setting of $|\phi| = \mathrm{poly}(T)$ the previous best result was D in AC^0 [ASW09].

Tighter connections between satisfiability and lower bounds. The quest for non-trivial satisfiability algorithms has seen significant progress recently, see e.g. [Wil11b, Her11, IMP12, BIS12, IPS13, CKS13]. Our results lower the bar for obtaining new circuit lower bounds from such algorithms. Previously, a lower bound for circuits of depth d and size s was implied by a satisfiability algorithm for depth $c \cdot d$ and size s^c for a constant $c > 1$ (for typical settings of s and d). With our proof it suffices to have a satisfiability algorithm for depth $d+c$ and size $c \cdot s$ for a constant c. This can be extended and optimized for several well-studied circuit classes. In particular we obtain the following new connections.

Corollary 1. *For each of the following classes C, if the satisfiability of circuits in C can be solved in time $2^n/n^{\omega(1)}$ then there is a problem $f \in \mathrm{E}^{\mathrm{NP}}$ that is not solvable by circuits in C:*

(1) linear-size circuits,
(2) linear-size series-parallel circuits,
(3) linear-size log-depth circuits,
(4) quasi-polynomial-size SYM-AND circuits.

Recall that available size lower bounds for unrestricted circuits are between $3n - o(n)$ and $5n - o(n)$, depending on the basis [Blu84, LR01, IM02]. Although Corollary 1 and Corollary 2 below are stated in terms of linear-size circuits, the proofs provide a close correspondence between the running time for satisfiability and the parameters of the circuit class. In particular, the constant hidden by the circuit size in class (1) can be optimized, as discussed in the paragraph "Subsequent work" below. At the moment this approach does not match known lower bounds, due to the (in)efficiency of known satisfiability algorithms.

In 1977 Valiant [Val77] focused attention on classes (2) and (3). (Some missing details about series-parallel graphs are provided in [Cal08].) The class (4) contains ACC [Yao90, BT94], and can be simulated by number-on-forehead protocols with a polylogarithmic number of players and communication [HG91]. Williams [Wil11b] gives a quasilinear-time algorithm to evaluate a SYM-AND circuit on all inputs.

For class (4) one can in fact obtain $f \in \mathrm{NE}$ using the seminal work by Impagliazzo, Kabanets, and Wigderson [IKW01] and its extension by Williams [Wil13a, Wil11b]. But to do so for classes (1)-(3), one would need a strengthening of [IKW01] to linear-size circuits, which we raise as an open problem.

It has long been known that the satisfiability of classes (1)-(3) in Corollary 1 can be linked to kSAT. Using Corollary 1, we can link kSAT to circuit lower bounds. (In the following, a kSAT instance has n variables and $O(n)^k$ clauses.)

Corollary 2.

(1) Assume that the exponential time hypothesis (ETH) is false [IP01]; i.e., for every $\epsilon > 0$, 3SAT is in time $2^{\epsilon n}$. Then there is a problem $f \in \mathrm{E}^{\mathrm{NP}}$ that is not solvable by linear-size circuits.

(2) Assume that the strong exponential time hypothesis (SETH) is false [IP01]; i.e., there is $\epsilon < 1$ such that for every k, kSAT is in time $2^{\epsilon n}$. Then there is a problem $f \in \mathrm{E}^{\mathrm{NP}}$ that is not solvable by linear-size series-parallel circuits.

(3) Assume that there is $\alpha > 0$ such that n^{α}-SAT is in time $2^{n-\omega(n/\log\log n)}$. Then there is a problem $f \in \mathrm{E}^{\mathrm{NP}}$ that is not solvable by linear-size log-depth circuits.

In Corollary 2, only (1) was known [Wil13a, Theorem 6.1]. Our proof is different: we obtain it immediately from (1) in Corollary 1 by the Cook-Levin theorem.

For context, the best algorithms for kSAT run in time $2^{n(1-O(1/k))}$ [DGH+02, PPSZ05].

Finally, we consider the class of polynomial-size depth-d circuits of threshold gates, which may have unbounded or bounded weights. (The latter case corresponds to Majority.) Recall that anything computed by a poly-size depth-d circuit with unbounded weights can be computed by a depth $d+1$ circuit with bounded weights [HMP+93, GHR92], and that it is not known if $\mathrm{EXP}^{\mathrm{NP}}$ has poly-size unbounded-weight circuits of depth $d = 2$. For these classes (and others) we show that a lower bound for depth d follows from a satisfiability algorithm for depth $d+2$.

Corollary 3. *Consider unbounded fan-in circuits consisting of threshold gates (either bounded- or unbounded-weight). Let d be an integer.*

Suppose that for every c, given a circuit of depth $d+2$ and size n^c on n input bits one can decide its satisfiability in time $2^n/n^{\omega(1)}$.

Then NE does not have circuits of polynomial size and depth d.

A diagram of some of the classes mentioned above, and their relative power, can be found in [Vio13].

Our results have a few other consequences. For example they imply that the so-called succinct version of various NP-complete problems remain NEXP-complete even if described by an NC^0 circuit. In particular we obtain this for 3SAT and 3Coloring. Our techniques are also relevant to the notion of circuit uniformity. A standard notion of uniformity is log-space uniformity, requiring that the circuit is computable in logarithmic space or, equivalently, that given an index to a gate in the circuit one can compute its type and its children in linear space. Equivalences with various other uniformity conditions are given by Ruzzo [Ruz81], see also [Vol99]. We consider another uniformity condition which is stronger than previously considered ones in some respects. Specifically,

we describe the circuit by showing how to compute children by an NC^0 circuit, i.e. a function with constant locality.

Theorem 2 (L-uniform $\Leftrightarrow$ local-uniform). *Let $f : \{0,1\}^* \to \{0,1\}$ be a function computable by a family of log-space uniform polynomial-size circuits. Then f is computable by a family of polynomial-size circuits $C = \{C_n : \{0,1\}^n \to \{0,1\}\}_n$ such that there is a Turing machine that on input n (in binary) runs in time $O(\mathrm{poly}\log n)$ and outputs a circuit $D : \{0,1\}^{O(\log n)} \to \{0,1\}^{O(\log n)}$ such that (i) D has constant locality: every output bit depends on $O(1)$ input bits, and (ii) on input a label g of a gate in C_n, D outputs the type of g and labels for each child.*

Does this paper simplify the proof that NEXP is not in ACC?. Recall that the proof [Wil11b] that NEXP is not in ACC uses as a black-box a result like Theorem 1 but with the requirement on the efficiency of D relaxed to polynomial-size circuits. If one instead uses as a black-box Theorem 1, one obtains a simpler proof, reported for completeness in the full version of this paper.

In fact, to obtain the separation of NEXP from ACC it suffices to prove a weaker version of Theorem 1 where D is, say, in AC^0. This weaker version has a simpler proof, as explained in §3. Independently of our work, Kowalski and Van Melkebeek proved this AC^0 result (personal communication).

Subsequent work. The announcement of our results as (ECCC Technical Report 13-099, July 2013) contained the same results as above except it did not mention Corollary 2 and items (2) and (4) in Corollary 1. After that announcement several related works have appeared. Oliveira's survey [Oli13] contains an alternative connection between satisfiability and circuit lower bounds, which yields a different proof of our Corollary 3 establishing a depth-2 overhead in that connection. Williams [Wil14] shows that the ability to count the number of satisfying assignments to circuits faster than brute-force search yields lower bounds against related circuits. His connection preserves the type of the gates in the input layer, a feature which is used to obtain some new lower bounds.

The work [BV14] builds on our results and is concurrent with [Wil14]. It gives a connection between derandomization and lower bounds that also preserves the type of the gates in the input layer. Thus, derandomization (or satisfiability), as opposed to counting, is sufficient for the lower bounds in [Wil14]. [BV14] also improves the depth loss of 2 in Corollary 3 to 1. Finally, they make a step in the direction we suggested of optimizing the constants in Item (1) of Corollary 1. In combination with the standard Cook-Levin reduction to 3SAT, they obtain that if 3SAT is in deterministic time c^n for any $c < 2^{1/10} = 1.07\ldots$ then E^{NP} does not have circuits of size $3n$ over the standard, full basis. Note that such a lower bound does not easily follow from diagonalization because the description length of a circuit of size $3n$ is superlinear. (Also recall the available lower bounds have the form $3n - o(n)$). The current record for solving 3SAT deterministically has $c = 1.33\ldots$ [MTY11], cf. [Her11].

As a corollary to [BV14], in this revision we show that even a somewhat more modest improvement to 3SAT algorithms would imply new lower bounds

for non-boolean functions with range $m = 2$ bits. Such lower bounds do not seem known for any $m = o(n)$, cf. [KMM12].

Corollary 4 (Corollary to [BV14]). *If 3SAT is in time c^n for any $c < 2^{1/7} = 1.10\ldots$, then there exists a (non-Boolean) function $f : \{0,1\}^n \to \{0,1\}^2$ in E^{NP} such that any circuit over the full basis computing it requires at least $3n$ (non-input) gates.*

3 Techniques

Proofs of the theorems and corollaries above are omitted due to space constraints, but they can be found in the full version of this paper at the authors' websites. We now give an overview of the techniques used.

Background: Reducing non-deterministic time T to size-$\tilde{O}(T)$ 3SAT. Our starting point is the reduction of non-deterministic time-T computation to 3SAT instances of quasilinear size $T' = \tilde{O}(T)$. The classical proof of this result [HS66,Sch78,PF79,Coo88,GS89,Rob91] hinges on the oblivious Turing machine simulation by Pippenger and Fischer [PF79]. However computing connections in the circuit induced by the oblivious TM is a somewhat complicated recursive procedure, and we have not been able to use this construction for our results.

Instead, we use a proof by Van Melkebeek [vM06, §2.3.1] which replaces this simulation by coupling an argument due to Gurevich and Shelah [GS89] with sorting circuits. We note that the idea of using sorting is already in [GS89], but if one follows their paper one ends up using again the oblivious simulation. Van Melkebeek's observation is that essentially all that needs to be done obliviously is sorting, and so one can use a sorting network, a more familiar construction than the oblivious simulation. Specifically, Van Melkebeek uses Batcher's odd-even mergesort networks [Bat68]. This proof was rediscovered by a superset of the authors as a class project [VN12]. We now recall it in more detail.

Consider any general model of (non-deterministic) computation, such as RAM or random-access Turing machines. (One nice feature of this proof is that it directly handles models with random-access, aka direct-access, capabilities.) The proof reduces computation to the satisfiability of a circuit C. The latter is then reduced to 3SAT via the textbook reduction. Only the first reduction to circuit satisfiability is problematic and we will focus on that one here. Consider a non-deterministic time-T computation. The proof constructs a circuit of size $\tilde{O}(T)$ whose inputs are (non-deterministic guesses of) T configurations of the machine. Each configuration has size $O(\log T)$ and contains the state of the machine, all registers, and the content of the memory locations indexed by the registers. This computation is then verified in two steps. First, one verifies that every configuration C_i yields configuration C_{i+1} assuming that all bits read from memory are correct. This is a simple check of adjacent configurations. Then to verify correctness of read/write operations in memory, one sorts the configurations by memory indices, and within memory indices by timestamp. Now

verification is again a simple check of adjacent configurations. The resulting circuit is outlined in Figure 1 (for a $2k$-tape random-access Turing machine). Using a sorting network of quasilinear size $\tilde{O}(T)$ results in a circuit of size $\tilde{O}(T)$.

Making low-space computation local. We employ a general technique that we call *spreading computation*. This shows that any circuit C whose connections can be computed in space linear in the description of a gate (i.e., space $\log |C|$) has an equivalent circuit C' of size $|C'| = \text{poly}|C|$ whose connections can be computed with constant locality.

The main idea in the proof is simply to let the gates of C' represent configurations of the low-space algorithm computing children in C. Then computing a child amounts to performing one step of the low-space algorithm, (each bit of) which can be done with constant locality in a standard Turing machine model.

We note that the technique of labeling gates by configurations goes back at least to the work of Ruzzo [Ruz81] who uses it to show the equivalence of some uniformity conditions involving alternating Turing machines that are simultaneously time and space restricted. However, [Ruz81] does not show how to compute gate connections with small locality, which is our aim here. We note that this task is non-trivial. For example, with constant locality one cannot even check the validity of a configuration. This means that the circuit C' has many invalid gates, i.e., gates that do not correspond to the computation of the low-space algorithm on a label of C. These gates could induce loops that do not correspond to computation, and make the final 3SAT instance always unsatisfiable. We avoid cycles by augmenting the low-space algorithm with a preliminary check for the validity of the configuration, and by including a clock in the configurations. These allow us to ensure that each invalid gate leads to a sink.

We apply spreading computation to the various sub-circuits checking consistency of configurations, corresponding to the triangles in Figure 1. These sub-circuits operate on configurations of size $O(\log T)$ and have size $\text{poly} \log T$. Hence, we can tolerate the polynomial increase in their complexity given by the spreading computation technique.

There remain however tasks for which we cannot use spreading computation. One is the sorting sub-circuit. Since it has size $> T$ we cannot afford a polynomial increase. Another task is indexing adjacent configurations. We now discuss these two in turn.

Sorting. We first mention a natural approach that gets us close but not quite to our main theorem. The approach is to define an appropriate labeling of the sorting network so that its connections can be computed very efficiently. We are able to define a labeling of bit-length $t + O(\log t) = \log \tilde{O}(T)$ for comparators in the odd-even mergesort network of size $\tilde{O}(2^t)$ (and depth t^2) that sorts $T = 2^t$ elements such that given a label one can compute the labels of its children by a decision tree of depth logarithmic in the length of the label, i.e. depth $\log \log \tilde{O}(T)$. With a similar labeling we can get linear size circuits. Or we can get constant locality at the price of making the 3SAT instance of size $T^{1+\epsilon}$. The details appear in the separate work [JMV14].

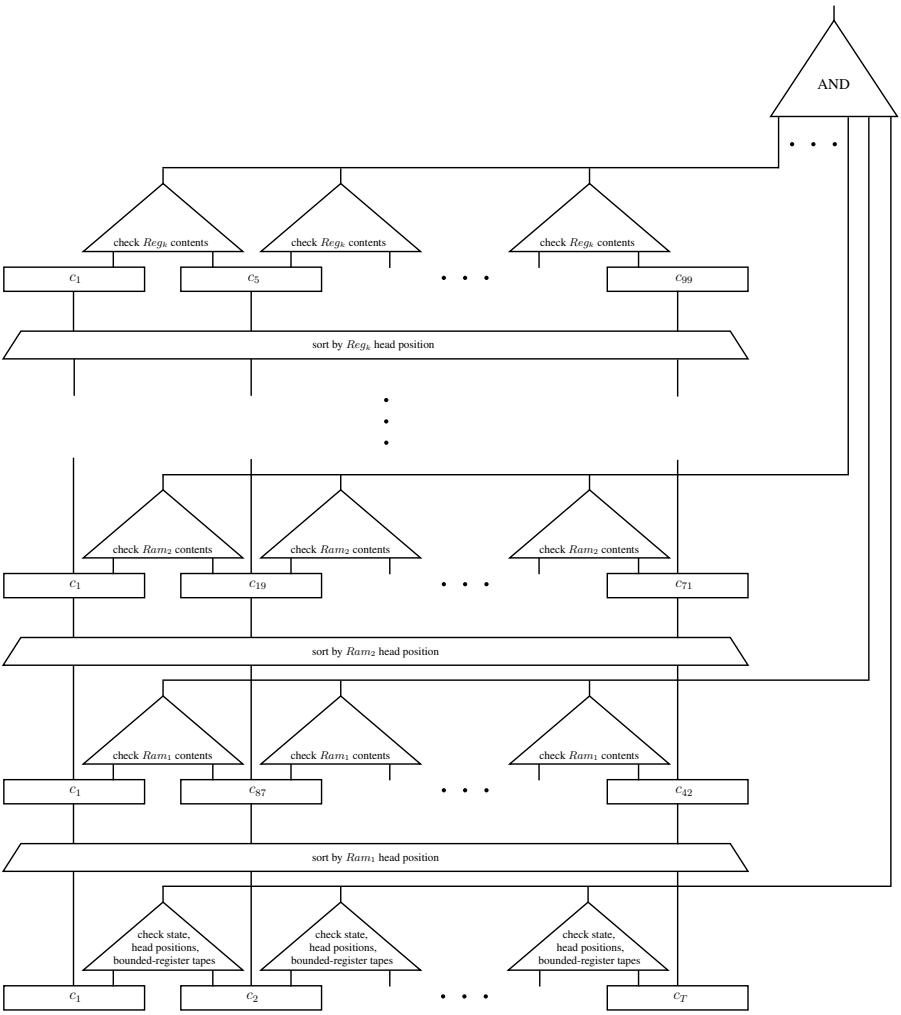

Fig. 1. Each of the T configurations has size $O(\log T)$. The checking circuits have size $\operatorname{poly}\log T$. The sorting circuits have size $\tilde{O}(T)$. k is a constant. Hence overall circuit has size $\tilde{O}(T)$.

To obtain constant locality we use a variant by Ben-Sasson, Chiesa, Genkin, and Tromer [BCGT13]. They replace sorting networks with routing networks based on De Bruijn graphs. We note that routing networks have been used extensively in the PCP literature starting, to our knowledge, with the work of Polishchuk and Spielman [PS94]. They have been used mostly for their algebraic properties, whereas we exploit the small locality of these networks. Specifically, the connections of these networks involve computing bit-shift, bit-xor, and addition by 1. The first two operations can easily be computed with constant locality, but the latter cannot in the standard binary representation. However, this addition by 1 is only on $O(\log \log T)$ bits. Hence we can afford an alternative, redundant representation which gives us an equivalent network where all the operations can be computed with constant locality. This representation again introduces invalid labels; those are handled in a manner similar to our spreading computation technique.

Plus one. Regardless of whether we are using sorting or routing networks, another issue that comes up in all previous proofs is addition by 1 on strings of $> \log T$ bits. This is needed to index adjacent configurations C_i and C_{i+1} for the pairwise checks in Figure 1. As mentioned before, this operation cannot be performed with constant locality in the standard representation. Also, we cannot afford a redundant representation (since strings of length $c \log T$ would correspond to an overall circuit of size $> T^c$).

For context, we point out an alternative approach to compute addition by 1 with constant locality which however cannot be used because it requires an inefficient pre-processing. The approach is to use primitive polynomials over $\mathrm{GF}(2)^{\log T}$. These are polynomials modulo which x has order $2^{\log T} - 1$. Addition by 1 can then be replaced by multiplication by x, which can be shown to be local. This is similar to *linear feedback registers*. However, it is not known how to construct such polynomials efficiently w.r.t. their degrees, see [Sho92].

To solve this problem we use routing networks in a different way from previous works. Instead of letting the network output an array $C_1, C_2, \ldots$ representing the sorted configurations, we use the network to represent the "next configuration" map $C_i \to C_{i+1}$. Viewing the network as a matrix whose first column is the input and the last column is the output, we then perform the pairwise checks on every pair of input and output configurations that are in the same row. The bits of these configurations will be in the same positions in the final label, thus circumventing addition by one.

As we mentioned earlier, for a result such as NEXP not in ACC [Wil11b] it suffices to prove a weaker version of our Theorem 1 where the reduction is computed by, say, an AC^0 circuit. For the latter, it essentially suffices to show that either the sorting or the routing network's connections are in that class.

Acknowledgments. We are very grateful to Eli Ben-Sasson for a discussion on routing networks which led us to improving our main result, cf. §3. We also thank Ryan Williams for feedback on the write-up.

References

[AAI+01] Agrawal, M., Allender, E., Impagliazzo, R., Pitassi, T., Rudich, S.: Reducing the complexity of reductions. Computational Complexity **10**(2), 117–138 (2001)

[AK10] Allender, E., Koucký, M.: Amplifying lower bounds by means of self-reducibility. J. of the ACM, **57**(3) (2010)

[ASW09] Arora, S., Steurer, D., Wigderson, A.: Towards a study of low-complexity graphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part I. LNCS, vol. 5555, pp. 119–131. Springer, Heidelberg (2009)

[Bat68] Batcher, K.E.: Sorting networks and their applications. AFIPS Spring Joint Computing Conference **32**, 307–314 (1968)

[BCGT12] Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E.: On the concrete-efficiency threshold of probabilistically-checkable proofs. Electronic Colloquium on Computational Complexity (ECCC) **19**, 45 (2012)

[BCGT13] Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E.: Fast reductions from RAMs to delegatable succinct constraint satisfaction problems. In: ACM Innovations in Theoretical Computer Science Conf. (ITCS), pp. 401–414 (2013)

[BGH+05] Ben-Sasson, E., Goldreich, O., Harsha, P., Sudan, M., Vadhan, S.P.: Short PCPs verifiable in polylogarithmic time. In: IEEE Conf. on Computational Complexity (CCC), pp. 120–134 (2005)

[BIS12] Beame, P., Impagliazzo, R., Srinivasan, S.: Approximating AC^0 by small height decision trees and a deterministic algorithm for $\#AC^0$sat. In: IEEE Conf. on Computational Complexity (CCC), pp. 117–125 (2012)

[Blu84] Blum, N.: A boolean function requiring 3n network size. Theoretical Computer Science **28**, 337–345 (1984)

[BT94] Beigel, R., Tarui, J.: On ACC. Computational Complexity **4**(4), 350–366 (1994)

[BV14] Ben-Sasson, E., Viola, E.: Short PCPs with projection queries (2014). http://www.ccs.neu.edu/home/viola/

[Cal08] Calabro, C.: A lower bound on the size of series-parallel graphs dense in long paths. Electronic Colloquium on Computational Complexity (ECCC), **15**(110) (2008)

[CKS13] Chen, R., Kabanets, V., Saurabh, N.: An improved deterministic #SAT algorithm for small De Morgan formulas. Technical Report TR13-150, Electronic Colloquium on Computational Complexity (2013). http://www.eccc.uni-trier.de/

[Coo88] Cook, S.A.: Short propositional formulas represent nondeterministic computations. Information Processing Letters **26**(5), 269–270 (1988)

[DGH+02] Dantsin, E., Goerdt, A., Hirsch, E.A., Kannan, R., Kleinberg, J., Papadimitriou, C., Raghavan, P., Schöning, U.: A deterministic $(2 - 2/(k + 1))n$ algorithm for k-SAT based on local search. Theoretical Computer Science **289**(1), 69–83 (2002)

[FLvMV05] Fortnow, L., Lipton, R., van Melkebeek, D., Viglas, A.: Time-space lower bounds for satisfiability. J. of the ACM **52**(6), 835–865 (2005)

[GHR92] Goldmann, M., Håstad, J., Razborov, A.A.: Majority gates vs. general weighted threshold gates. Computational Complexity **2**, 277–300 (1992)

[GS89] Gurevich, Y., Shelah, S.: Nearly linear time. In: Logic at Botik, Symposium on Logical Foundations of Computer Science, pp. 108–118 (1989)
[Her11] Hertli, T.: 3-SAT faster and simpler - unique-SAT bounds for PPSZ hold in general. In: IEEE Symp. on Foundations of Computer Science (FOCS), pp. 277–284 (2011)
[HG91] Håstad, J., Goldmann, M.: On the power of small-depth threshold circuits. Comput. Complexity **1**(2), 113–129 (1991)
[HMP+93] Hajnal, A., Maass, W., Pudlák, P., Szegedy, M., Turán, G.: Threshold circuits of bounded depth. J. of Computer and System Sciences **46**(2), 129–154 (1993)
[HS66] Hennie, F., Stearns, R.: Two-tape simulation of multitape turing machines. J. of the ACM **13**, 533–546 (1966)
[IKW01] Impagliazzo, R., Kabanets, V., Wigderson, A.: In search of an easy witness: Exponential time vs. probabilistic polynomial time. In: IEEE Conf. on Computational Complexity (CCC) (2001)
[IM02] Iwama, K., Morizumi, H.: An explicit lower bound of $5n - o(n)$ for boolean circuits. In: Symp. on Math. Foundations of Computer Science (MFCS), pp. 353–364 (2002)
[IMP12] Impagliazzo, R., Matthews, W., Paturi, R.: A satisfiability algorithm for AC^0. In: ACM-SIAM Symp. on Discrete Algorithms (SODA), pp. 961–972 (2012)
[IP01] Impagliazzo, R., Paturi, R.: On the complexity of k-SAT. J. of Computer and System Sciences **62**(2), 367–375 (2001)
[IPS13] Impagliazzo, R., Paturi, R., Schneider, S.: A satisfiability algorithm for sparse depth-2 threshold circuits. IEEE Symp. on Foundations of Computer Science (FOCS) (2013)
[JMV14] Jahanjou, H., Miles, E., Viola, E.: Succinct and explicit circuits for sorting and connectivity (2014). http://www.ccs.neu.edu/home/viola/
[KMM12] Kulikov, A.S., Melanich, O., Mihajlin, I.: A $5n - o(n)$ lower bound on the circuit size over U_2 of a linear boolean function. In: Cooper, S.B., Dawar, A., Löwe, B. (eds.) CiE 2012. LNCS, vol. 7318, pp. 432–439. Springer, Heidelberg (2012)
[LR01] Lachish, O., Raz, R.: Explicit lower bound of 4.5n - o(n) for boolena circuits. In: ACM Symp. on the Theory of Computing (STOC), pp. 399–408 (2001)
[Mie09] Mie, T.: Short pcpps verifiable in polylogarithmic time with o(1) queries. Ann. Math. Artif. Intell. **56**(3–4), 313–338 (2009)
[MTY11] Makino, K., Tamaki, S., Yamamoto, M.: Derandomizing HSSW algorithm for 3-SAT (2011). CoRR, abs/1102.3766
[Oli13] Oliveira, I.C.: Algorithms versus circuit lower bounds (2013). CoRR, abs/1309.0249
[PF79] Pippenger, N., Fischer, M.J.: Relations among complexity measures. J. of the ACM **26**(2), 361–381 (1979)
[PPSZ05] Paturi, R., Pudlák, P., Saks, M.E., Zane, F.: An improved exponential-time algorithm for k-sat. J. of the ACM **52**(3), 337–364 (2005)
[PS94] Polishchuk, A., Spielman, D.A.: Nearly-linear size holographic proofs. In: ACM Symp. on the Theory of Computing (STOC), pp. 194–203 (1994)
[Rob91] Robson, J.M.: An O(T log T) reduction from RAM computations to satisfiability. Theoretical Computer Science **82**(1), 141–149 (1991)
[Ruz81] Ruzzo, W.L.: On uniform circuit complexity. J. of Computer and System Sciences **22**(3), 365–383 (1981)

[Sch78] Schnorr, C.-P.: Satisfiability is quasilinear complete in NQL. J. of the ACM **25**(1), 136–145 (1978)
[Sho92] Shoup, V.: Searching for primitive roots in finite fields. Math. Comp. **58**, 369–380 (1992)
[SW12] Santhanam, R., Williams, R.: Uniform circuits, lower bounds, and qbf algorithms. Electronic Colloquium on Computational Complexity (ECCC) **19**, 59 (2012)
[Val77] Valiant, L.G.: Graph-theoretic arguments in low-level complexity. In: Gruska, J. (ed.) MFCS 1977. LNCS, vol. 53, pp. 162–176. Springer, Heidelberg (1977)
[Vio13] Viola, E.: Challenges in computational lower bounds (2013). http://www.ccs.neu.edu/home/viola/
[vM06] van Melkebeek, D.: A survey of lower bounds for satisfiability and related problems. Foundations and Trends in Theoretical Computer Science **2**(3), 197–303 (2006)
[VN12] Viola, E., NEU. From RAM to SAT (2012). http://www.ccs.neu.edu/home/viola/
[Vol99] Vollmer, H.: Introduction to circuit complexity. Springer-Verlag, Berlin (1999)
[Wil11a] Williams, R.: Guest column: a casual tour around a circuit complexity bound. SIGACT News **42**(3), 54–76 (2011)
[Wil11b] Williams, R.: Non-uniform ACC circuit lower bounds. In: IEEE Conf. on Computational Complexity (CCC), pp. 115–125 (2011)
[Wil13a] Williams, R.: Improving exhaustive search implies superpolynomial lower bounds. SIAM J. on Computing **42**(3), 1218–1244 (2013)
[Wil13b] Williams, R.: Natural proofs versus derandomization. In: ACM Symp. on the Theory of Computing (STOC) (2013)
[Wil14] Williams, R.: New algorithms and lower bounds for circuits with linear threshold gates (2014)
[Yao90] Yao, A.C.-C.: On ACC and threshold circuits. In: IEEE Symp. on Foundations of Computer Science (FOCS), pp. 619–627 (1990)