GRAPH-THEORETIC ARGUMENTS IN LOW-LEVEL COMPLEXITY

Leslie G. Valiant
Computer Science Department
University of Edinburgh
Edinburgh, Scotland.

1. Introduction

A major goal of complexity theory is to offer an understanding of why some specific problems are inherently more difficult to compute than others. The pursuit of this goal has two complementary facets, the positive one of finding fast algorithms, and the negative one of proving lower bounds on the inherent complexity of problems. Finding a proof of such a lower bound is equivalent to giving a property of the class of all algorithms for the problem. Because of the sheer richness of such classes, even for relatively simple problems, very little is yet understood about them and consequently the search for lower bound proofs has met with only isolated successes.

The poverty of our current knowledge can be illustrated by stating some major current research goals for three distinct models of computation. In each case complexity is measured in terms of n, the sum of the number of inputs and outputs:
(A) Discrete problems: For some natural problem known to be computable in polynomial time on a multi-tape Turing machine (TM) prove that no TM exists that computes it in time $O(n)$. This problem is open even when TMs are restricted to be oblivious [12].
(B) Discrete finite problems: For some problem computable in polynomial time on a TM show that no combinational circuit over a complete basis exists that is of size $O(n)$.
(C) Algebraic problems: For some natural sets of multinomials of constant degree over a ring show that no straight-line program consisting of the operations $+, -$, and $\times$, exists of size $O(n)$.

Known results on lower bounds are excluded by the above specifications either because they assume other restrictions on the models, or for the following reasons: For TMs lower bounds for natural problems have only been found for those of apparent or provable exponential complexity or worse [11,6,7]. For unrestricted combinational circuits all arguments involve counting. The only problems that have been proved of nonlinear complexity are those that can encode a counting process and are of exponential complexity or more [4,20]. For algebraic problems "degree arguments" have been successfully applied to natural problems, but only when the degrees grow with n [21].

Algebraic independence arguments have been applied only to problems which we would not regard here as natural. (Various linear lower bounds do exist [9,14,19] but we are not concerned with these here).

This paper focusses on one particular approach to attempting to understand computations for the above models. The approach consists of analysing the global flow of information in an algorithm by reducing this to a combination of graph-theoretic, algebraic and combinatorial problems. We shall restrict ourselves to lower bound arguments and shall omit some related results that exploit the same approach but are better regarded as positive applications of it [7,13,24]. The hope of finding positive byproducts, in particular new surprising algorithms, remains, however, a major incentive in our pursuit of negative results.

Though organized as a survey article, the main purpose of this paper is to present some previously unpublished results. Among other things they show, apparently for the first time, that a significant computational property (the non-achievability of size $O(n)$ and depth $O(\log n)$ simultaneously) of unrestricted straight-line arithmetic programs for certain problems can be reduced to non-computational questions (see §6). The grounds on which we claim that a "meaningful reduction" has been demonstrated are perhaps the weakest that can be allowed. Nevertheless, in the absence of alternative approaches to understanding these problems, we believe that these grounds are sufficient to make the related questions raised worthy of serious investigation.

2. Preliminaries

In the main we follow [23] and [25] for definitions: A straight-line program is a sequence of assignments each of the form $x := f(y,z)$ where f belongs to a set of binary functions and x,y,z belong to a set of variables that can take values in some domain. The only restriction is that any variable x occurring on the left-hand side of some assignment cannot occur in any assignment earlier in the sequence. The variables that never occur on the left-hand side of any instruction are called input variables. The graph of a straight-line program is an acyclic directed graph that has a node, denoted by $\bar{u}$, for each variable u in the program, and directed edges $(\bar{y},\bar{x})$ and $(\bar{z},\bar{x})$ for each instruction $x := f(y,z)$.

A linear form in indeterminates $x_1,\ldots,x_n$ over a field F is any expression of the form $\Sigma\lambda_i x_i$ where each $\lambda_i \varepsilon F$. A linear program over F on inputs $x_1,\ldots,x_n$ is a straight-line program with $\{x_1,\ldots,x_n\}$ as input variables and function set $\{f_{\lambda,\mu} \mid \lambda,\mu \varepsilon F\}$ where $f_{\lambda,\mu}(u,v) = \lambda u + \mu v$. The importance of linear programs is that, for certain fields F, for computing the values of sets of linear forms in $x_1,\ldots,x_n$ with each x_i ranging over F, linear programs are optimal to within a constant factor as compared with straight-line programs in which unrestricted use of all the operations $\{+,-,*,\div\}$ is allowed [26,22,3] . Examples of such fields are the real and complex numbers. Hence the results in §6 apply to the unrestricted model in the case of arithmetic problems over these fields. (Note that there is a similar correspondence between bilinear forms and bilinear programs, and this can be exploited in the same way.)

Straight-line programs over GF(2) define just the class of combinational circuits over the complete basis <and, exclusive-or> . Also, the combinational complexity of a function bounds its oblivious TM complexity from below by a constant factor. Unfortunately the optimality of linear programs for evaluating sets of linear forms over GF(2) is at present unknown. Hence the results in §6 may be relevant only for the restricted class of circuits corresponding to linear programs.

A "graph-theoretic argument" for a lower bound on the complexity of a problem P consists of two parts:
(i) For some graph theoretic property X a proof that the graph of any program for P must have property X.
(ii) A proof that any graph with property X must be of size superlinear in n .

We note that the graph of any algorithm has indegree two, and hence the number of edges is bounded by twice the number of nodes. Conversely, isolated nodes are clearly redundant. Hence, by defining the _size_ of a graph to be the number of edges, we will be measuring, to within a constant factor, both the number of nodes and the number of instructions in any corresponding algorithm. In this paper graphs will always be assumed to be _directed_ and _acyclic_. The fixed indegree property will not be assumed, except where so indicated. Note that by replacing each node by a binary fanin tree a graph can be made to have fanin two without more than doubling its size or destroying any flow properties relevant here.

A _labelling_ of a directed acyclic graph is a mapping of the nodes into the integers such that for each edge $(\bar{u},\bar{v})$ the label of $\bar{v}$ is strictly greater than the label of $\bar{u}$. If the total number of nodes on the longest directed path in the graph is d then d is the _depth_ of the graph. It is easily verified that if each node is labelled by the total number of nodes on the longest directed path that terminates at it, then this constitutes a consistent labelling using only the integers $1,2,\ldots,d$.

3. Shifting Graphs

Connection networks are graphs in which certain sets of specified input-output connections can be realised. For simplicity we consider the canonical case of a directed acyclic graph G with n input nodes $a_0, a_1, \ldots, a_{n-1}$ and n output nodes $b_0, b_1, \ldots, b_{n-1}$. If σ is a permutation mapping of the integers $\{1,\ldots,n\}$ then G implements σ iff there are n mutually node disjoint paths joining the n pairs $\{a_i, b_{\sigma(i)} \mid 0 \leqslant i < n\}$. It is well-known that any graph that implements all $n!$ different permutations has to be of size at least $n\log_2 n \approx \log_2(n!)$ simply because there are $n!$ different sets of paths to be realised. Furthermore this order of size (in fact $6n\log_3 n + O(n)$ [2,18]) is achievable. It is perhaps remarkable that even to implement just the n distinct circular shifts $\{\sigma_i \mid \sigma_i(j) = j+i \mod n ;\ 0 \leqslant i \leqslant n-1\}$ a graph of size $3n\log_3 n$ is necessary. This follows from the following special case of a result proved in [18] :

Theorem 3.1 If $\sigma_1, \ldots, \sigma_s$ are any permutations such that for all $i,j,k (i \neq j)$ $\sigma_i(k) \neq \sigma_j(k)$ then any graph that implements all the s permutations has to have size at least $3n\log_3 s$. □

In fact two distinct constructions of size $3n\log_3 n + O(n)$ are known for such shifting graphs [18,23].

The above theorem has been used to prove superlinear lower bounds on the complexity of problems for various restricted models of computation. The restriction necessary is that the algorithm be conservative or be treatable as such. Conservatism as defined in [18,23] means that the input elements of the algorithm are atomic unchangeable elements that can be compared or copied in the course of the algorithm, but not used to synthesize new elements or transmuted in any way. This notion is a generic one that has to be made precise for each model of computation.

Applications of shifting graphs to proving lower bounds for various merging, shifting and pattern matching problems can be found in [18]. In each case the lower bound is closely matched by an $O(n\log n)$ upper bound and is either new or related to results proved elsewhere by more specialized arguments.

Unfortunately it appears that connection networks cannot be applied to unrestricted models (interpreted here to mean models (A), (B) and (C)). The presence of negation or subtraction allows for pairs of equivalent algorithms of the following genre:

(i) $b_1 := a_1 ;\ b_2 := a_2 ;$

(ii) $x := a_1 + a_2 ;\ b_1 := x - a_2 ;\ b_2 := x - a_1 ;$

In the graph of the second algorithm the identity permutation is not implemented, contrary to its semantics.

4. Superconcentrators

Concentration networks are graphs in which specified sets of input nodes have to be connected to specified sets of output nodes, but it is immaterial which particular pairs of nodes in these sets are connected. Various kinds of concentration networks have been studied [16]. Superconcentrators were defined in [23] to have the most restrictive property of this kind.

Definition A directed acyclic graph with distinguished input nodes $a_1, \ldots a_n$, and output nodes $b_1, \ldots, b_n$ is an n-superconcentrator iff for all r $(1 \leq r \leq n)$ for all sets A of r distinct input nodes and all sets B of r distinct output nodes, there are r mutually node-disjoint paths going from nodes in A to nodes in B.

It has been shown for many computational problems that the graph of any algorithm for computing it must be a superconcentrator, or have some weaker property of a similar nature. For example for convolution a superconcentrator is necessary, for the discrete Fourier transform a hyperconcentrator, and for matrix multiplication in a ring, or for $(\wedge, \vee)$-Boolean matrix multiplication, a matrix concentrator (see [23] for definitions and proofs.) Furthermore, for at least one restricted model of computation, the BRAM [23] , it can be shown that the graphs associated with these properties have to be of size $kn\log n$ and hence the algorithms must have this complexity. (A BRAM is a random access machine in which unit cost is assigned to communication between locations whose addresses differ by a power of two, and inputs are in consecutive locations.)

Contrary to expectation, however, it has been also shown [23] that superconcentrators do not account for superlinear complexity in unrestricted algorithms:

Theorem 4.1 $\exists k\ \forall n$ there is an n-superconcentrator of size kn. □

An improvement on the original construction found by Pippenger [17] has size $39n$, constant indegree and outdegree, and depth $O(\log n)$.

Although this is a negative result for lower bounds, it is also a positive result about the surprising richness of connections possible in small graphs. As hoped for, this has led to a surprising result due to V.Strassen, about the existence of new fast algorithms, and has refuted a previously plausible conjecture:

Theorem 4.2 $\exists k\ \forall n$ there is an $n \times n$ integer matrix A in which all minors of all sizes are nonsingular, but such that the n linear forms $A\underline{x}$ (where $\underline{x}$ is the column vector $(x_1, \ldots, x_n)$) can be computed together in kn time.

Proof Consider an n-superconcentrator of linear size with fanin two. Give the nodes unique labels in some consistent way. Construct a linear program by identifying the n inputs with $x_1, \ldots, x_n$ respectively, and defining the linear combination $f_{\lambda,\mu}(u,v)$ at each node in the order of the labels as follows: Choose λ and μ to have the property that "$\forall r$ $(1 \leq r \leq n)$, for all sets $\{w_1, \ldots, w_{r-1}\}$ of functions computed at smaller labels, for all sets X of r components of $\{x_1, \ldots, x_n\}$, if

$\{u,w_1,\ldots, w_{r-1}\}$ and $\{v,w_1,\ldots, w_{r-1}\}$ when restricted to X are both linearly independent then so is $\{\lambda u + \mu v, w_1,\ldots, w_{r-1}\}$ over the same set of components". Clearly for each combination of r, $\{w_1,\ldots,w_{r-1}\}$ and X at most one ratio $\lambda:\mu$ will be forbidden. Hence we can always find integral values of λ and μ at each node.

For any $r \times r$ minor B of A consider a set of r node disjoint paths from the r inputs X corresponding to the columns of B to the outputs corresponding to the rows of B. It is easily verified by induction that the $r \times r$ matrix corresponding to the restriction to X of the r linear forms computed at "parallel" nodes on the r disjoint paths as these are traced in order of their labels, is always nonsingular.□

We note that much yet remains to be understood about superconcentrators: Both of the known constructions [23,17] use as building blocks certain bipartite graphs, called "partial concentrators" in [16], for which no completely constructive construction is known [10,16]. Little is known about what restrictions have to be imposed on graphs to ensure that superconcentrators be of superlinear size. The one such restriction known is the one corresponding to BRAMs [23]. In the other direction the two restrictions considered in the next chapter (of $O(\log n)$ depth, and the "series-parallel" property), the linear construction in [17] has both. Yet another relevant restriction is the one corresponding to oblivous TM computations, called TM-graphs in [15]. W. Paul and R. Tarjan have raised the question as to whether there exist linear size TM-graphs that are superconcentrators.

5. Graphs that are Dense in Long Paths

We come to a different graph property that has been suspected of accounting for the complexity of algorithms. The first concrete evidence that it does so in at least a limited sense will be explained in the next section. The property has been studied previously by Erdos, Graham and Szemeredi [5] but only for parameters other than the ones we require. Here we shall prove the sought after nonlinear bounds for the relevant parameters for two distinct restricted classes of graphs: (i) shallow graphs (i.e. depth $O(\log n)$), and (ii) series-parallel graphs, defined later.

Definition A directed acyclic graph G has the $R(n,m)$ property iff whichever set of n edges are removed from G, some directed path of m edges remains in G. Let $S(n,m,d)$ be the size of the smallest graph of depth at most d with the $R(n,m)$ property.

The following generalizes a corresponding result in [5] and simplifies the proof. (An intermediate form was stated in [24].)

Theorem 5.1 $S(n,m,d) > (n\log_2 d)/(\log_2(d/m))$ assuming for simplicity that m and d are exact powers of 2.

Proof Consider any graph with q edges and depth d and consider a labelling of it with $\{0,1,\ldots,d-1\}$. Let X_i $(i = 1,2,\ldots,\log_2 d)$ be the set of edges between pairs of labels x and y such that the most significant bit in which their binary representations differ is the i^{th} (from the left). If X_i is removed from the graph then we can validly relabel the nodes by $0,1,\ldots,(d/2)-1$, by simply deleting the i^{th} bits in all the old labels. Consequently if any $s \leq \log_2 d$ of the X_i's are removed a graph of depth $d/2^s$ remains.

The union of the s smallest of the classes $\{X_1,\ldots,X_{\log_2 d}\}$ contains at most $qs/\log_2 d$ edges. Hence we conclude that

$$S(qs/\log_2 d\ ,\ d/2^s\ ,\ d) > q$$

or $$S(n,m,d) > (n\log_2 d)/\log_2(d/m). \qquad \square$$

Corollary 5.2 For any $k > 0$ the depth of any graph with $q \leq (n\log_2 d)/k$ can be reduced to $d/2^k$ by removing some set of n edges. $\square$
(Theorems 2 and 3 in [5] correspond to the cases $d \leq n\log_2 n$, $k = \log\log_2 n$ and $d = n$, $k =$ constant.)

That Corollary 5.2 is optimal to within constant factors for all d, provided that k is a constant, follows from Theorem 1 in [5]', which states that for some constants $c_1,c_2 > 0$, $S(c_1p,\ c_1p,\ p) \leq c_2 p\log_2 p$. Placing in parallel $n/c_2 d$ such bad graphs for $p = d$ gives the result for all d.

When k is not a constant optimality is unknown. In the extreme case of $k = \varepsilon\log_2 d$ the corollary says only that if n edges are removed from a graph of size n/ε then the depth can be reduced to at most $d^{1-\varepsilon}$.

5.1 Shallow Graphs

The application of Corollary 5.2 in §6 is the case $m < \log_2 n$, to which the following instance of it is applicable directly:

Corollary 5.3 The depth of any graph with $d = c(\log_2 n)^{c'}$ can be reduced to $d/\log\log n$ by removing some set of n edges, if $q < (n\log\log n)/\log\log\log n$. $\square$
Typical applications are $d = O(\log n)$ and $d = O((\log n)\log\log\log n))$. Note that the practical significance of depth $O(\log n)$, besides its obvious optimality, is that for numerous problems the most efficient algorithms known achieve this depth (e.g. discrete Fourier transform, Strassen's matrix multiplication algorithm [1].)

5.2 Series-Parallel Graphs

This is roughly the class of graphs that can be constructed recursively from subgraphs placed in series or parallel. Nearly all known efficient constructions of circuits have this property, as is also the case for relevant graph constructions (e.g. superconcentrators [23,17], universal graphs [24], and graphs dense in long paths as given in [15], though not in [5].)

Definition A graph with designated sets of input nodes and of output nodes is an sp-graph iff there is a labelling of it such that all inputs have one label, all outputs another label, and for all pairs of edges (i,j) and (k,m) it is the case that $(i - k)(m - j) \geqslant 0$.

Definition An sp-graph has the $R'(n,m)$ property iff whichever set of n edges are removed some directed path of at least m edges remains from an input to an output. $S_{sp}(n,m)$ is the size of the smallest sp-graph with the $R'(n,m)$ property.

Theorem 5.4 For some constant $c > 0$

$$S_{sp}(n,m) \geqslant cn\log\log_2 m.$$

Proof We perform "induction on edges" in the manner of [13,7]. We assume sp-graphs with designated input arcs (directed out of nodes of indegree zero and outdegree one) and output arcs (directed into nodes of indegree one and outdegree zero). Only paths that go from an input arc to an output arc will be counted. In the induction the input arcs and output arcs are not counted in the size of the graph or of the paths.

Consider a graph G with the $R'(n,m)$ property. Consider a labelling of it satisfying the sp-condition and find the smallest label i such that the following has the $R'(n/2,(m-2)/2)$ property: the graph G_1 consisting of all the nodes labelled less than i and all connections between them, with the original input arcs to this subgraph as input arcs, and all arcs directed out of these nodes to the outside as output arcs. By the choice of i if a certain set of $n/2$ arcs are removed from G_1 then no path longer than $(m-2)/2$ will remain. Clearly the complementary graph G_2 on all the nodes labelled greater than i must also have the $R'(n/2,(m-2)/2)$ property, for otherwise by removing some $n/2$ edges from each of G_1 and G_2 we would have no path longer than $(m - 2)/2 + 2 + (m - 2)/2 - 1 = m - 1$.

The sum of the sizes of G_1 and G_2 will be the size of G minus r , the total number of edges between some node with label i and some internal node of G_1 or G_2 and between some internal node of G_1 and one of G_2. Hence

$$S_{sp}(n,m) \geqslant 2S_{sp}(n/2,(m-2)/2) + r . \qquad (1)$$

The special property of sp-graphs that we exploit is that at least one of the following must hold in G: (i) there are no input arcs directed into nodes with label greater than i , (ii) there are no output arcs directed out of nodes with label less than i . Without loss of generality we shall assume the former. Then if the r connections are removed then no remaining input-output path in G involves any node in G_2. Hence if $r < n/4$ we have that G_1 has the $R'(3n/4,m)$ property. Since it is clear that $S_{sp}(3n/4,m) \geqslant S_{sp}(n/2,m) + n/4$ it follows that

$$S_{sp}(n,m) \geqslant 2S_{sp}(n/2,(m-2)/2) + n/4.$$

In the alternative case of $r \geqslant n/4$ the same inequality is immediate from (1). Solving this recurrence gives the claimed bound. □

Problem 1 Can Corollary 5.2 be improved? The particularly relevant question is to settle whether $S(n,\log_2 n,\infty)$ is linear in n or not. [N.B. We have shown that no $o(n\log\log_2 n)$ construction can be sp.]

Problem 2 How can deep graphs, and graphs without the sp-property be exploited in algorithms and circuits to obtain substantial reductions in total complexity?

6. Grates and Rigidity

We finally discuss a pair of notions introduced in [25], which offer a proof that nontrivial complexity measures for unrestricted arithmetic programs can be related to natural non-computational properties of the function to be computed. We emphasize that the results are weak in two senses: (i) the lower bounds we prove are on the combination of size and depth achievable simultaneously by any algorithm (i.e. that simultaneous size $O(n)$ and depth $O(\log n)$ is impossible,) and (ii) while we can prove for our non-computational property that "most" sets of linear forms possess them, we have not been able to prove it for any specific natural problem. We believe, however, that further progress on both issues is possible. In particular it appears plausible to conjecture that these properties, (which are more severe than the $R(n,\log_2 n)$ property) do guarantee superlinear size.

We shall assume now that all matrices are $n \times n$ and have elements drawn from a field F.

Definition The density of a matrix A is the number of nonzero entries in A (and is denoted by $\mathrm{dens}(A)$).

Definition The rigidity of a matrix A is the function

$$R_A(r) : \{1,\ldots,n\} \to \{0,1,\ldots,n^2\}$$

defined by

$$R_A(r) = \min\{i \mid \exists B \text{ with } \mathrm{dens}(B) = i \text{ and } \mathrm{rank}(A + B) \leqslant r\}.$$

From elementary matrix properties it is easy to verify that for any F and any matrix A, $R_A(r) \leqslant (n - r)^2$ for each r. As we shall see later (Theorem 6.4) this maximal rigidity is indeed achieved by "most" matrices.

The significance of the notion of rigidity comes from the fact that it can be related intimately to the following graph-theoretic property.

Definition A directed acyclic graph G is an f(r)-grate iff for some subsets $\{a_1,\ldots,a_s\}$ and $\{b_1,\ldots,b_t\}$ of its nodes it has the property that "if any r nodes and adjacent edges are removed from G then for at least $f(r)$ of the st distinct pairs (a_i,b_j) there remains a directed path from a_i to b_j in G."

The function $f(r)$ will be specified on a subset of the integers and will be assumed to be zero for all other values of r. The slightly weaker restriction corresponding to specific chosen values of s and t will be called an $(f(r),s,t)$-grate.
The next theorem shows that a typical case of interest for linear forms is the $((n-r)^2,n,n)$-grate. The smallest graphs known with such properties are shifting networks which are of size $\sim 3n\log_3 n$ and are in fact $(n(n-r),n,n)$-grates

Theorem 6.1 (i) The graph of any linear program for computing a set of linear forms $A\underline{x}$ is an $R_A(r)$-grate. (ii) Conversely, if for some r $f(r) > R_A(r)$ then there exists a linear program P for computing $A\underline{x}$ whose graph is not an $f(r)$-grate (w.r.t. the natural inputs and outputs).

Proof (i) Let $s = t = n$ and identify $a_1,\ldots,a_n$ with the inputs $x_1,\ldots,x_n$ and $b_1,\ldots,b_n$ with the nodes at which the outputs are computed. We assume for the sake of contradiction that for some r $(1 \leq r \leq n)$ if a certain set of r nodes are removed then fewer than $R_A(r)$ input-output pairs remain connected. This implies that if the multipliers λ and μ at these r nodes are changed to zero then the matrix B of the linear forms computed by the modified program has density less than $R_A(r)$. However, the rows of B differ from the corresponding ones of A only by linear combinations of the forms computed by the original program at the removed nodes. (To verify this, for each output expand the sub-program that computes it into a tree structure. Let N be the set of nodes in the tree corresponding to the (possibly repeated) occurrences of the removed nodes. Consider the contribution to the output of all the nodes in N that are not separated from the root by other nodes in N.) It follows that $A = B + X$ for some $n \times n$ matrix X of rank r and hence, by the definition of rigidity, that $R_A(r) \leq \text{dens}(B) < R_A(r)$, a contradiction.

(ii) Suppose that for some given r $f(r) > R_A(r)$. Consider a matrix C of rank r such that $\text{dens}(A-C) = R_A(r)$. Let P first compute the following $n + r$ forms in the obvious way as $n + r$ separate computations: (a) a set X of r linearly independent forms from $C\underline{x}$, and (b) $(A - C)\underline{x}$. The n outputs $A\underline{x}$ are then computed as linear combinations of the above. Clearly if the r nodes corresponding to X are removed then the remaining graph contains n disjoint trees, with the outputs as roots and with $R_A(r) < f(r)$ input-output connections. □

The above theorem motivates two complexes of problems, one to do with the size of graphs and the other with the rigidity of natural functions. Positive solutions to Problems 3 and 4 below would give the desired superlinear lower bounds on the complexity of natural sets of linear forms. An alternative result to aim for would be bilinear forms (e.g. matrix multiplication) which would require solutions to problems 3 and 5.

The main evidence we have that the above theorem does provide a reduction of a nontrivial computational property to a noncomputational problem is the conjunction of Corollary 6.3 and Theorem 6.4 below.

Proposition 6.2 $\forall\varepsilon > 0$, $\forall c > 0$, $\forall k > 0$ and for all sufficiently large n, any $f(r)$-grate of indegree two and depth $k\log_2 n$ with $f(n) > cn^{1+\varepsilon}$ has size at least $(n\log\log n)/\log\log\log n$.

Proof Assume the contrary. By corollary 5.3 some set of n nodes can be removed from any graph of size $(n\log\log n)/\log\log\log n$ and depth $k\log n$ so as to leave no path longer than $(k\log n)/\log\log n$. Hence each output will be connected to at most $n^{k/\log\log n} = o(n^{\varepsilon})$ inputs after the deletions. This implies that the graph is not an $f(r)$-grate for any sufficiently large n, which is a contradiction. □

Corollary 6.3 Let $A_1, A_2, \ldots$ be an infinite family where A_n is an $n \times n$ real matrix and for some $c, \varepsilon > 0$, $R_{A_n}(n/2) \geq cn^{1+\varepsilon}$. Then there does not exist a family of straight-line programs for the corresponding sets of linear forms that for some $c_1, c_2 > 0$, (i) achieve size $c_1 n$ and depth $c_2 \log n$ simultaneously for all n, or (ii) are series-parallel and of size $c_1 n$ for all n.

Proof (i) Immediate from Theorem 6.1(i), Proposition 6.2, and the fact that the standard translation from straight-line programs to linear programs changes both the size and depth by only a constant factor. (ii) Follows similarly from Theorem 5.4.

Theorem 6.4 (i) For F infinite, $\forall n \exists\, n \times n$ matrix A such that $R_A(r) = (n-r)^2$.

(ii) For F finite with c elements, $\forall n \exists\ n \times n$ matrix A such that for all $r < n - \sqrt{(2n.\log_c 2 + \log_2 n)}$,

$$R_A(r) \geq ((n-r)^2 - 2n \log_c 2 - \log_2 n)/(2 \log_c n + 1).$$

Proof Define a mask σ to be any subset of s pairs from the set of pairs $\{(i,j) \mid 1 \leq i,j \leq n\}$. A minor τ is any pair of subsets of $\{i \mid 1 \leq i \leq n\}$, both of size t. Define $M(\sigma,\tau)$ to be the set of all $n \times n$ matrices A with the property that "$\exists$B such that (i) all the non-zero entries of B are indexed by σ, (ii) rank (A+B) = t, and (iii) τ specifies one of the minors of C = A+B of maximal rank t in C."

Without loss of generality we shall assume that τ is in the top left corner. We shall denote an $n \times n$ matrix X generically by

$$\begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix}$$

where X_{11} is $t \times t$.

Consider the set of all matrices of rank t that have a minor of maximal rank in the top left corner. Clearly there is a fixed set $\{f_k'\}$ of $(n-t)^2$ rational functions such that for any C in this set of matrices the entries of C_{22} are given by these functions in terms of the entries of C_{11}, C_{12} and C_{21}. But each element of $M(\sigma,\tau)$ differs from some element of this class by only an additive B. It follows that there is a fixed set $\{f_k\}$ of n^2 rational functions such that the entries of any $A \in M(\sigma,\tau)$ are given by these functions in terms of $(n^2 - (n-t)^2 + s)$ arguments (i.e. the entries of C_{11}, C_{12}, C_{21} and the non-zero entries of B). Hence each element of $\{M(\sigma,\tau) \mid \sigma,\tau\}$ is the image of F^{2tn-t^2+s} under some rational mapping into F^{n^2}

(i) Hence for any r all the matrices that can be reduced to rank r by adding a matrix of density $(n-r)^2-1$ belong to the union of the images in F^{n^2} of a finite number of rational mappings from F^{n^2-1}. But if F is infinite the result follows since for any u the finite union of the images of F^u under rational mappings into F^{u+1} is properly contained in F^{u+1}. (This last fact can be established by first showing that if $f_1,\ldots,f_{u+1}$ are rational functions of $x_1,\ldots,x_u$ then the f's are algebraically dependent. (A counting argument in the style of [3] p.442 suffices if applied to the numerators of these functions when put over a common denominator). It then follows that the points in any finite union of such images are the roots of a non-trivial polynomial, and therefore cannot fill F^{n^2}).

(ii) If F has $c < \infty$ elements then the number of elements in $M(\sigma,\tau)$ is bounded by the size of F^{2tn-t^2+s}, i.e. c^{2tn-t^2+s}. For fixed s and t the number of possible choices of σ is

$$n^2C_s \leqslant 2^{2s \log_2 n} ,$$

and of τ is

$$(nC_t)^2 < 2^{2n}.$$

Hence for fixed s and t the number of matrices in the union of $M(\sigma,\tau)$ over all σ, τ of these sizes is bounded by

$$c^{2tn-t^2 + s + 2s \log_c n + 2n \log_c 2} .$$

It follows that for any $t < n - \surd(2n \log_c 2 + \log_2 n)$, if

$$0 \leqslant s < ((n-t)^2 -2n \log_c 2 - \log_c n)/(1 + 2 \log_c n)$$

then the number of such matrices is less than

$$c^{n^2 - \log_c n} = c^{n^2}/n.$$

Hence the union of all these matrices over all values of t will not fill F^{n^2}. □

Unfortunately we do not know of any explicit characterization of matrices of high rigidity. Indeed we have the following matrix-theoretic result that there are integer matrices in which all minors of all sizes are nonsingular but whose rank can be reduced to $o(n)$ by changing a mere $o(n^{1+\varepsilon})$ elements:

Proposition 6.5 For each n there is an $n \times n$ matrix A in which all minors of all sizes are nonsingular but

$$R_A((n\log\log\log n)/\log\log n) \leqslant n^{1 + O(1/\log\log n)} .$$

Proof Let A be the matrix of Theorem 4.2 constructed from a superconcentrator of size $O(n)$ and depth $O(\log n)$. Applying Corollary 5.3 to the graph of this algorithm in the manner of Proposition 6.2 gives that for $r = (n\log\log\log n)/\log\log n$ $f(r) \leqslant n^{1 + O(1/\log\log n)}$. The result then follows from Theorem 6.1(i). □

We note that although grates seem more restrictive than the corresponding $R(n, O(\log n))$ graphs, Proposition 6.2 exploits them only via this weakening correspondence. There therefore remains a hope that much better bounds are provable for them.

Problem 3 Prove a lower bound superlinear in n on the size of $f(r)$-grates for appropriately "nonlinear" $f(r)$. One candidate is: $f(r) = (n-r)^2$ for $r = 1,\ldots,n$. A weaker candidate is: $f(r) = kn^2$ when $r = n$ and $f(r) = 0$ when $r \neq n$. (Alternatively prove a linear upper bound noting that no such construction can be "series-parallel" or "shallow".)

Problem 4 For some natural $n \times n$ matrix A prove that $R_A(r)$ is large. A bound of $k(n - r)^2$ is one aim. A weaker aim would be one on the value of $R_A(n/2)$ alone, of $kn^2, kn^{1+\varepsilon}$, or some other superlinear function in n. Natural candidates for A are: (i) for the integers some Vandermonde matrix (i.e. $A_{ij} = z_i^{j-1}$ for distinct $z_1, z_2, \ldots z_n$), (ii) for the complex numbers the discrete Fourier transform matrix (i.e. $A_{ij} = w^{(i-1)(j-1)}$ where w is an n^{th} primitive root of unity), and (iii) for GF(2) the 0-1 matrix associated with a finite projective plane.

Problem 5 It is known that for computing sets of bilinear forms (e.g. matrix multiplication, convolution) bilinear programs are optimal to within a constant factor [3,26]. Prove that the graph of any bilinear program for a natural set of bilinear forms is an $f(r)$-grate for such values of $f(r)$ as in Problem 3.

7. Conclusion

We have surveyed one approach to understanding complexity issues for certain easily computable natural functions. Shifting graphs have been seen to account accurately and in a unified way for the superlinear complexity of several problems for various restricted models of computation. To attack "unrestricted" models (in the present context combinational circuits or straight-line arithmetic programs) a first attempt, through superconcentrators, fails to provide any lower bounds although it does give counter-examples to alternative approaches. The notion of rigidity, however, does offer for the first time a reduction of relevant computational questions to noncomputional properties. The "reduction" consists of the conjunction of Corollary 6.3 and Theorem 6.4 which show that "for most sets of linear forms over the reals the stated algebraic and combinatorial reasons account for the fact that they cannot be computed in linear time and depth $O(\log n)$ simultaneously." We have outlined some problem areas which our preliminary results raise, and feel that further progress on most of these is humanly feasible. We would be interested in alternative approaches also.

Problem 6 Propose reductions of relevant complexity issues to noncomputational properties, that are more promising or tractable than the ones above.

References

1. Aho, A.V., Hopcroft, J.E. and Ullman, J.D., The Design and Analysis of Computer Algorithms, Addison Wesley, 1974.

2. Benes, V.E., *Mathematical Theory of Connecting Networks and Telephone Traffic.* Academic Press, New York, 1965.

3. Borodin, A.B. and Munro, I. *The Complexity of Algebraic and Numeric Problems,* American Elsevier, 1975.

4. Ehrenfeucht, A. Practical decidability. Report CU-CS-008-72, Univ. of Colorado (1972).

5. Erdős, P., Graham and Szemerédi, Ě. On sparse graphs with dense long paths. *Comp. and Maths. with Appls.,* *1*, (1975) 365-369.

6. Fischer, M.J. and Rabin, M.O. Super-exponential complexity of Presburger arithmetic. MACTR43, Project MAC,MIT,(1974).

7. Hopcroft, J.E., Paul, W.J. and Valiant, L.G. Time versus space and related problems. *Proc. 16th Symp. on Foundations of Computer Science,* Berkeley, (1975) 57-64.

8. Hartmanis, J., Lewis, P.M. and Stearns, R.E. Classification of Computations by time and memory requirements. *Proc. IFIP Congress 1965*, Spartan, N.Y., 31-35.

9. Hyafil, L. and Kung, H.T. The complexity of parallel evaluation of linear recurrence. *Proc. 7th ACM Symp. on Theory of Computing* (1975) 12-22.

10. Margulis, G.A. Explicit constructions of Concentrators, *Problemy Peredachi Informatsii,* *9* :4(1973) 71-80.

11. Meyer, A.R. and Stockmeyer, L.J. The word problem for regular expressions with squaring requires exponential space. *Proc. 13th IEEE Symp. on Switching and Automata Theory,* (1972)125-129.

12. Paterson, M.S., Fischer, M.J. and Meyer A.R. An improved overlap argument for on-line multiplication. *SIAM-AMS Proceedings* Vol *7*, (1974) 97-111

13. Paterson, M.S. and Valiant, L.G. Circuit size is nonlinear in depth. *Theoretical Computer Science* *2* (1976) 397-400.

14. Paul, W.J. A 2.5N Lower bound for the combinational complexity of boolean functions. *Proc. 7th ACM Symp. on Theory of Computing,* (1975) 27-36.

15. Paul, W.J., Tarjan, R.E. and Celoni, J.R. Space bounds for a game on graphs. *Proc. 8th ACM Symp. on Theory of Computing,* (1976) 149-160.

16. Pippenger, N. The complexity theory of switching networks. Ph.D. Thesis, Dept. of Elect. Eng., MIT, (1973).

17. Pippenger, N. Superconcentrators. RC5937. IBM Yorktown Heights (1976).

18. Pippenger, N. and Valiant, L.G. Shifting graphs and their applications. JACM 23 (1976) 423-432.

19. Schnorr, C.P. Zwei lineare Schranken fur die Komplexität Boolischer Funktionen, Computing, 13 (1974) 155-171.

20. Stockmeyer, L.J. and Meyer, A.R. Inherent computational complexity of decision problems in logic and automata theory. Lecture Notes in Computer Science (to appear), Springer

21. Strassen, V. Die Berechnungkomplexität von elementar symmetrichen Funktionen und von Interpolationskoeffizienten. Numer. Math 20 (1973) 238-251.

22. Strassen, V. Vermeidung von Divisionen, J.Reine Angew.Math., 264,(1973), 184-202.

23. Valiant, L.G. On non-linear lower bounds in computational complexity. Proc. 7th ACM Symp. on Theory of Computing, (1975) 45-53.

24. Valiant, L.G. Universal circuits. Proc. 8th ACM Symp. on Theory of Computing, (1976) 196-203.

25. Valiant, L.G. Some conjectures relating to superlinear lower bounds. TR85, Dept. of Comp. Sci., Univ. of Leeds (1976).

26. Winograd, S. On the number of multiplications necessary to compute certain functions. Comm. on Pure and App. Math. 23 (1970) 165-179.